National
Semiconductor
Corporation

400066 Rev. 1

Microcommunication Elements Databook

- *UARTs*
- *NSC800 Family*

A Corporate Dedication to Quality and Reliability

National Semiconductor is an industry leader in the manufacture of high quality, high reliability integrated circuits. We have been the leading proponent of driving down IC defects and extending product lifetimes. From raw material through product design, manufacturing and shipping, our quality and reliability is second to none.

We are proud of our success . . . it sets a standard for others to achieve. Yet, our quest for perfection is ongoing so that you, our customer, can continue to rely on National Semiconductor Corporation to produce high quality products for your design systems.

Charlie Sporck

Charles E. Sporck
President, Chief Executive Officer
National Semiconductor Corporation

Wir fühlen uns zu Qualität und Zuverlässigkeit verpflichtet

National Semiconductor Corporation ist führend bei der Herstellung von integrierten Schaltungen hoher Qualität und hoher Zuverlässigkeit. National Semiconductor war schon immer Vorreiter, wenn es galt, die Zahl von IC Ausfällen zu verringern und die Lebensdauern von Produkten zu verbessern. Vom Rohmaterial über Entwurf und Herstellung bis zur Auslieferung, die Qualität und die Zuverlässigkeit der Produkte von National Semiconductor sind unübertroffen.

Wir sind stolz auf unseren Erfolg, der Standards setzt, die für andere erstrebenswert sind. Auch ihre Ansprüche steigen ständig. Sie als unser Kunde können sich auch weiterhin auf National Semiconductor verlassen.

La Qualité et La Fiabilité:
Une Vocation Commune Chez National Semiconductor Corporation

National Semiconductor Corporation est un des leaders industriels qui fabrique des circuits intégrés d'une très grande qualité et d'une fiabilité exceptionelle. National a été le premier à vouloir faire chuter le nombre de circuits intégrés défectueux et a augmenter la durée de vie des produits. Depuis les matières premières, en passant par la conception du produit sa fabrication et son expédition, partout la qualité et la fiabilité chez National sont sans équivalents.

Nous sommes fiers de notre succès et le standard ainsi défini devrait devenir l'objectif à atteindre par les autres sociétés. Et nous continuons à vouloir faire progresser notre recherche de la perfection; il en résulte que vous, qui êtes notre client, pouvez toujours faire confiance à National Semiconductor Corporation, en produisànt des systèmes d'une très grande qualité standard.

Un Impegno Societario di Qualità e Affidabilità

National Semiconductor Corporation è un'industria al vertice nella costruzione di circuiti integrati di altà qualità ed affidabilità. National è stata il principale promotore per l'abbattimento della difettosità dei circuiti integrati e per l'allungamento della vita dei prodotti. Dal materiale grezzo attraverso tutte le fasi di progettazione, costruzione e spedizione, la qualità e affidabilità National non è seconda a nessuno.

Noi siamo orgogliosi del nostro successo che fissa per gli altri un traguardo da raggiungere. Il nostro desiderio di perfezione è d'altra parte illimitato e pertanto tu, nostro cliente, puoi continuare ad affidarti a National Semiconductor Corporation per la produzione dei tuoi sistemi con elevati livelli di qualità.

Charlie Sporck

Charles E. Sporck
President, Chief Executive Officer
National Semiconductor Corporation

MICROCOMMUNICATION ELEMENTS

DATABOOK

1987 Edition

TRADEMARKS

Following is the most current list of National Semiconductor Corporation's trademarks and registered trademarks.

Abuseable™
Anadig™
ANS-R-TRAN™
APPS™
Auto-Chem Deflasher™
BI-FET™
BI-FET II™
BI-LINE™
BIPLAN™
BLC™
BLX™
Brite-Lite™
BTL™
CheckTrack™
CIM™
CIMBUS™
Clock✓Chek™
COMBO™
COMBO I™
COMBO II™
COPS™ microcontrollers
Datachecker®
DENSPAK™
DIB™
Digitalker®
DISCERN™
DISTILL™
DNR™
DPVM™
ELSTAR™
E-Z-LINK™
5-Star Service™
GENIX™
GNX™
HEX 3000™
HPC™
ICM™
INFOCHEX™
Integral ISE™
Intelisplay™
ISE™
ISE/06™
ISE/08™
ISE/16™
ISE32™
KeyScan™
LMCMOS™
M^{2}CMOS™
Macrobus™
Macrocomponent™
Meat✓Chek™
Microbus™ data bus
MICRO-DAC™
μtalker™
Microtalker™
MICROWIRE™
MICROWIRE/PLUS™
MOLE™
MST™
Naked-8™
National®
NAX 800™
Nitride Plus™
Nitride Plus Oxide™
NML™
NOBUS™
NSC800™
NSX-16™
NS-XC-16™
NURAM™
OXISS™
P^{2}CMOS™
Perfect Watch™
Pharma✓Chek™
PLAN™
Polycraft™
POSitalker™
Power & Control™
QUAD3000™
QUIKLOOK™
RAT™
RTX16™
SABR™
Script✓Chek™
SCX™
SERIES/800™
Series 3000™
Series 32000®
Shelf✓Chek™
SPIRE™
STAR™
Starlink™
STARPLEX™
STARPLEX II™
SuperChip™
SuperScript™
SYS32™
TapePak™
TDS™
TeleGate™
The National Anthem®
Time✓Chek™
TINA™
TLC™
Trapezoidal™
TRI-CODE™
TRI-POLY™
TRI-SAFE™
TRI-STATE®
TURBOTRANSCEIVER™
VIP™
VR32™
XMOS™
XPU™
Z STAR™
883B/RETS™
883S/RETS™

CP/M™ is a trademark of Digital Research Corporation
Z80® is a registered trademark of Zilog Corporation
IBM® is a registered trademark of International Business Machine Corporation
VAX™ is a trademark of Digital Equipment Corporation

LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF NATIONAL SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

National Semiconductor Corporation 2900 Semiconductor Drive, P.O. Box 58090, Santa Clara, California 95052-8090 (408) 721-5000
TWX (910) 339-9240

National does not assume any responsibility for use of any circuitry described, no circuit patent licenses are implied, and National reserves the right, at any time without notice, to change said circuitry or specifications.

Introduction

The Microcommunication Elements and NSC800 databook updates previous handbooks and datasheets and introduces new UARTs. It also contains a section on National's NSC800 Microprocessor Family. The databook provides the system designer with detailed technical descriptions of National's devices.

National Semiconductor Corporation is the leading supplier of Universal Asynchronous Receiver Transmitters (UART). National offers the most complete list of UARTs, covering all of the personal computers compatible with IBM type software.

Introduced in this catalog is the new National NS16550A. The NS16550A, an enhanced version of the NS16450, offers customers two onboard FIFOs that relieve the CPU of overhead allowing the CPU to take on more significant tasks.

National continues to be the innovative developer of new IBM type personal computer UARTs. As a company, National is committed to providing customers with unsurpassed quality and service.

Table of Contents

Section 1 CPU

Section 2 Peripherals

Section 3 Evaluation Board

Section 4 Logic Devices

Section 5 Appendices/Physical Dimensions

Alpha-Numeric Index

Section 1
CPU

Section 1 Contents

National Semiconductor Corporation

microCMOS

NSC800™ High-Performance Low-Power CMOS Microprocessor

General Description

The NSC800 is an 8-bit CMOS microprocessor that functions as the central processing unit (CPU) in National Semiconductor's NSC800 microcomputer family. National's microCMOS technology used to fabricate this device provides system designers with performance equivalent to comparable NMOS products, but with the low power advantage of CMOS. Some of the many system functions incorporated on the device, are vectored priority interrupts, refresh control, power-save feature and interrupt acknowledge. The NSC800 is available in dual-in-line and surface mounted chip carrier packages.

The system designer can choose not only from the dedicated CMOS peripherals that allow direct interfacing to the NSC800 but from the full line of National's CMOS products to allow a low-power system solution. The dedicated peripherals include NSC810A RAM I/O Timer, NSC858 UART, and NSC831 I/O.

All devices are available in commercial, industrial and military temperature ranges along with two added reliability flows. The first is an extended burn in test and the second is the military class C screening in accordance with Method 5004 of MIL-STD-883.

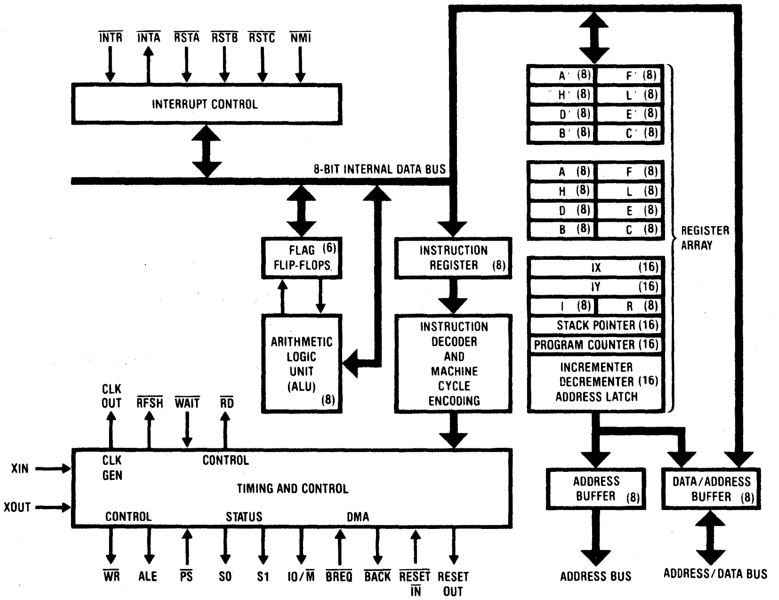
Features

- Fully compatible with Z80® instruction set:
  Powerful set of 158 instructions
  10 addressing modes
  22 internal registers
- Low power: 50 mW at 5V V_{CC}
- Unique power-save feature
- Multiplexed bus structure
- Schmitt trigger input on reset
- On-chip bus controller and clock generator
- Variable power supply 2.4V−6.0V
- On-chip 8-bit dynamic RAM refresh circuitry
- Speed: 1.0 μs instruction cycle at 4.0 MHz

NSC800-4	4.0 MHz
NSC800-3	2.5 MHz
NSC800-1	1.0 MHz

- Capable of addressing 64k bytes of memory and 256 I/O devices
- Five interrupt request lines on-chip

Block Diagram



TL/C/5171-73

Table of Contents

1.0 Absolute Maximum Ratings (Note 1)

If Military/Aerospace specified devices are required, contact the National Semiconductor Sales Office/ Distributors for availability and specifications.

Storage Temperature	−65°C to +150°C
Voltage on Any Pin with Respect to Ground	−0.3V to V_{CC} +0.3V
Maximum V_{CC}	7V
Power Dissipation	1W
Lead Temp. (Soldering, 10 seconds)	300°C

2.0 Operating Conditions

NSC800-1 →	T_A = 0°C to +70°C
	T_A = −40°C to +85°C
NSC800-3 →	T_A = 0°C to +70°C
	T_A = −40°C to +85°C
	T_A = −55°C to +125°C
NSC800-4 →	T_A = 0°C to +70°C
	T_A = −40°C to +85°C
	T_A = −55°C to +125°C

3.0 DC Electrical Characteristics V_{CC} = 5V ±10%, GND = 0V, unless otherwise specified.

Symbol	Parameter	Conditions	Min	Typ	Max	Units
V_{IH}	Logical 1 Input Voltage		0.8 V_{CC}		V_{CC}	V
V_{IL}	Logical 0 Input Voltage		0		0.2 V_{CC}	V
V_{HY}	Hysteresis at RESET IN input	V_{CC} = 5V	0.25	0.5		V
V_{OH1}	Logical 1 Output Voltage	I_{OUT} = −1.0 mA	2.4			V
V_{OH2}	Logical 1 Output Voltage	I_{OUT} = −10 μA	V_{CC} −0.5			V
V_{OL1}	Logical 0 Output Voltage	I_{OUT} = 2 mA	0		0.4	V
V_{OL2}	Logical 0 Output Voltage	I_{OUT} = 10 μA	0		0.1	V
I_{IL}	Input Leakage Current	$0 \leq V_{IN} \leq V_{CC}$	−10.0		10.0	μA
I_{OL}	Output Leakage Current	$0 \leq V_{IN} \leq V_{CC}$	−10.0		10.0	μA
I_{CC}	Active Supply Current	I_{OUT} = 0, $f_{(XIN)}$ = 2 MHz, T_A = 25°C		8	11	mA
I_{CC}	Active Supply Current	I_{OUT} = 0, $f_{(XIN)}$ = 5 MHz, T_A = 25°C		10	15	mA
I_{CC}	Active Supply Current	I_{OUT} = 0, $f_{(XIN)}$ = 8 MHz, T_A = 25°C		15	21	mA
I_Q	Quiescent Current	I_{OUT} = 0, $\overline{PS}$ = 0, V_{IN} = 0 or V_{IN} = V_{CC} $f_{(XIN)}$ = 0 MHz, T_A = 25°C, X_{IN} = 0, CLK = 1		2	5	mA
I_{PS}	Power-Save Current	I_{OUT} = 0, $\overline{PS}$ = 0, V_{IN} = 0 or V_{IN} = V_{CC} $f_{(XIN)}$ = 5.0 MHz , T_A = 25°		5	7	mA
C_{IN}	Input Capacitance			6	10	pF
C_{OUT}	Output Capacitance			8	12	pF
V_{CC}	Power Supply Voltage	(Note 2)	2.4	5	6	V

Note 1: Absolute Maximum Ratings indicate limits beyond which permanent damage may occur. Continuous operation at these limits is not intended and should be limited to those conditions specified under DC Electrical Characteristics.

Note 2: CPU operation at lower voltages will reduce the maximum operating speed. Operation at voltages other than 5V ±10% is guaranteed by design, not tested.

4.0 AC Electrical Characteristics V_{CC} = 5V ±10%, GND = 0V, unless otherwise specified

Symbol	Parameter	NSC800-1		NSC800		NSC800-4		Units	Notes
		Min	Max	Min	Max	Min	Max		
t_X	Period at XIN and XOUT Pins	500	3333	200	3333	125	3333	ns	
T	Period at Clock Output (=2 t_X)	1000	6667	400	6667	250	6667	ns	
t_R	Clock Rise Time		110		110		80	ns	Measured from 10%–90% of signal
t_F	Clock Fall Time		70		60		50	ns	Measured from 10%–90% of signal
t_L	Clock Low Time	435		150		85		ns	50% duty cycle, square wave input on XIN
t_H	Clock High Time	450		145		75		ns	50% duty cycle, square wave input on XIN
$t_{ACC(OP)}$	ALE to Valid Data		1340		490		300	ns	Add t for each $\overline{\text{WAIT}}$ STATE
$t_{ACC(MR)}$	ALE to Valid Data		1875		620		375	ns	Add t for each $\overline{\text{WAIT}}$ STATE
t_{AFR}	AD(0–7) Float after $\overline{\text{RD}}$ Falling		0		0		0	ns	
t_{BABE}	$\overline{\text{BACK}}$ Rising to Bus Enable		1000		400		250	ns	
t_{BABF}	$\overline{\text{BACK}}$ Falling to Bus Float		50		50		50	ns	
t_{BACL}	$\overline{\text{BACK}}$ Fall to CLK Falling	425		125		55		ns	
t_{BRH}	$\overline{\text{BREQ}}$ Hold Time	0		0		0		ns	
t_{BRS}	$\overline{\text{BREQ}}$ Set-Up Time	100		50		45		ns	
t_{CAF}	Clock Falling ALE Falling	0	70	0	65	0	55	ns	
t_{CAR}	Clock Rising to ALE Rising	0	100	0	100	0	80	ns	
t_{CRD}	Clock Rising to Read Rising		100		90		80	ns	
t_{CRF}	Clock Rising to Refresh Falling		80		70		60	ns	
t_{DAI}	ALE Falling to $\overline{\text{INTA}}$ Falling	445		160		85		ns	
t_{DAR}	ALE Falling to $\overline{\text{RD}}$ Falling	400	575	160	250	90	160	ns	
t_{DAW}	ALE Falling to $\overline{\text{WR}}$ Falling	900	1010	350	420	200	255	ns	
$t_{D(BACK)1}$	ALE Falling to $\overline{\text{BACK}}$ Falling	2460		975		600		ns	Add t for each $\overline{\text{WAIT}}$ state Add t for opcode fetch cycles
$t_{D(BACK)2}$	$\overline{\text{BREQ}}$ Rising to $\overline{\text{BACK}}$ Rising	500	1610	200	700	125	475	ns	
$t_{D(I)}$	ALE Falling to $\overline{\text{INTR}}$, $\overline{\text{NMI}}$, $\overline{\text{RSTA-C}}$, $\overline{\text{PS}}$, $\overline{\text{BREQ}}$, Inputs Valid		1360		475		250	ns	Add t for each $\overline{\text{WAIT}}$ state Add t for opcode fetch cycles
t_{DPA}	Rising $\overline{\text{PS}}$ to Falling ALE	500	1685	200	760	125	500	ns	See *Figure 14* also
$t_{D(WAIT)}$	ALE Falling to $\overline{\text{WAIT}}$ Input Valid		550		250		125	ns	

4.0 AC Electrical Characteristics V_{CC} = 5V ±10%, GND = 0V, unless otherwise specified (Continued)

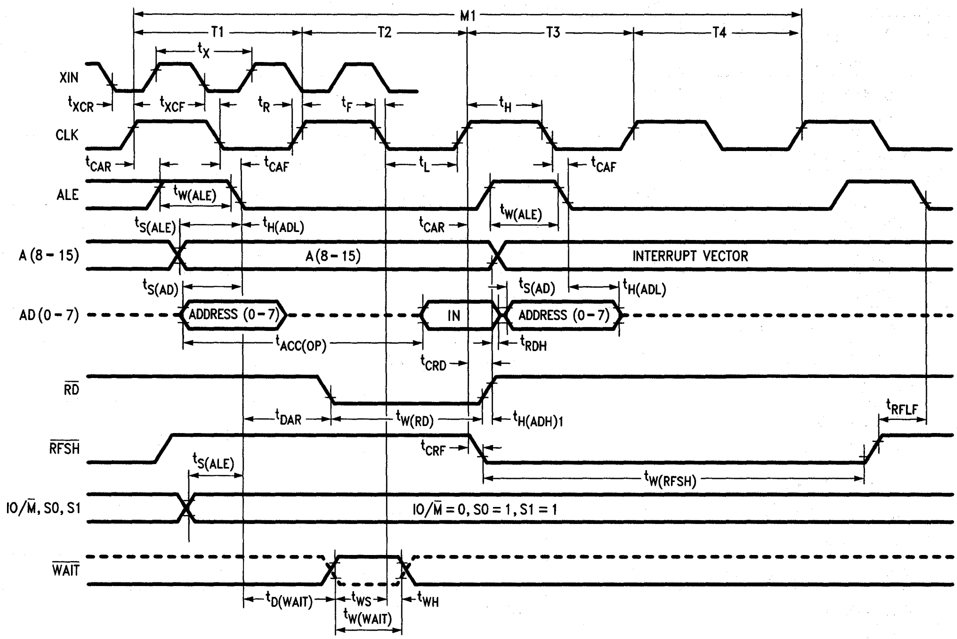
Symbol	Parameter	NSC800-1		NSC800		NSC800-4		Units	Notes
		Min	Max	Min	Max	Min	Max		
$T_{H(ADH)1}$	A(8–15) Hold Time During Opcode Fetch	0		0		0		ns	
$T_{H(ADH)2}$	A(8–15) Hold Time During Memory or IO, $\overline{RD}$ and $\overline{WR}$	400		100		60		ns	
$T_{H(ADL)}$	AD(0–7) Hold Time	100		60		30		ns	
$T_{H(WD)}$	Write Data Hold Time	400		100		75		ns	
t_{INH}	Interrupt Hold Time	0		0		0		ns	
t_{INS}	Interrupt Set-Up Time	100		50		45		ns	
t_{NMI}	Width of NMI Input	50		30		20		ns	
t_{RDH}	Data Hold after Read	0		0		0		ns	
t_{RFLF}	$\overline{RFSH}$ Rising to ALE Falling	60		50		40		ns	
$t_{RL(MR)}$	$\overline{RD}$ Rising to ALE Rising (Memory Read)	390		100		45		ns	
$t_{S(AD)}$	AD(0–7) Set-Up Time	300		45		40		ns	
$t_{S(ALE)}$	A(8–15), SO, SI, IO/$\overline{M}$ Set-Up Time	350		70		50		ns	
$t_{S(WD)}$	Write Data Set-Up Time	385		75		30		ns	
$t_{W(ALE)}$	ALE Width	430		130		100		ns	
t_{WH}	$\overline{WAIT}$ Hold Time	0		0		0		ns	
$t_{W(I)}$	Width of $\overline{INTR}$, $\overline{RSTA\text{-}C}$, $\overline{PS}$, $\overline{BREQ}$	500		200		125		ns	
$t_{W(INTA)}$	$\overline{INTA}$ Strobe Width	1000		400		200		ns	Add two t states for first $\overline{INTA}$ of each interrupt response string Add t for each $\overline{WAIT}$ state
t_{WL}	$\overline{WR}$ Rising to ALE Rising	450		130		70		ns	
$t_{W(RD)}$	Read Strobe Width During Opcode Fetch	960		360		185		ns	Add t for each $\overline{WAIT}$ State Add t/2 for Memory Read Cycles
$t_{W(RFSH)}$	Refresh Strobe Width	1925		725		395		ns	
t_{WS}	$\overline{WAIT}$ Set-Up Time	100		70		55		ns	
$t_{W(WAIT)}$	$\overline{WAIT}$ Input Width	550		250		175		ns	
$t_{W(WR)}$	Write Strobe Width	985		390		220		ns	Add t for each $\overline{WAIT}$ state
t_{XCF}	XIN to Clock Falling	25	100	20	95	5	80	ns	
t_{XCR}	XIN to Clock Rising	25	85	20	85	5	80	ns	

Note 1: Test conditions: t = 1000 ns for NSC800-1, 400 ns for NSC800, 250 ns for NSC800-4.

Note 2: Output timings are measured with a purely capacitive load of 100 pF.

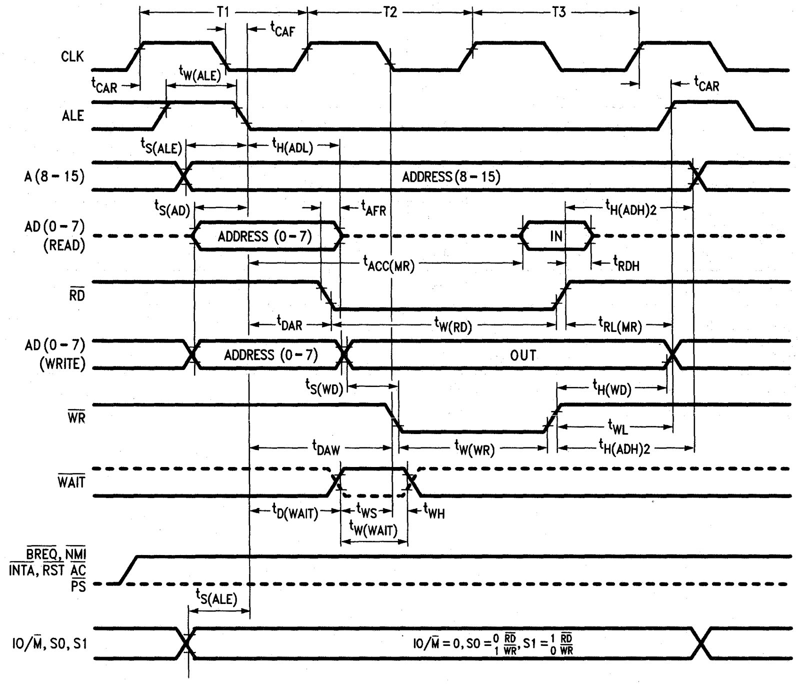
5.0 Timing Waveforms

Opcode Fetch Cycle



TL/C/5171–3

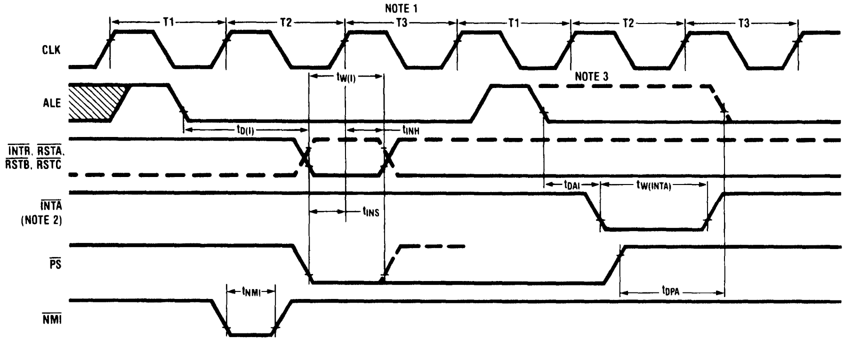
Memory Read and Write Cycle



TL/C/5171–4

5.0 Timing Waveforms (Continued)

Interrupt—Power-Save Cycle



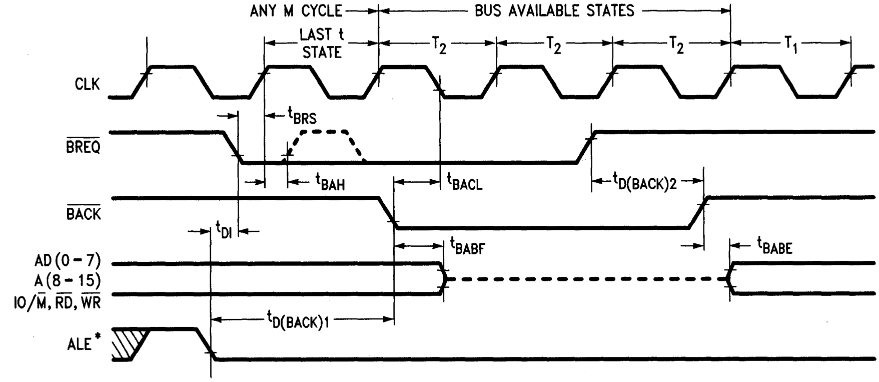
TL/C/5171-5

Note 1: This t state is the last t state of the last M cycle of any instruction.

Note 2: Response to INTR input.

Note 3: Response to PS input.

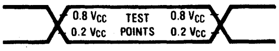
Bus Acknowledge Cycle



TL/C/5171-6

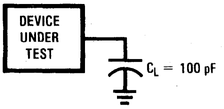
*Waveform not drawn to proportion. Use only for specifying test points.

AC Testing Input/Output Waveform



TL/C/5171-7

AC Testing Load Circuit



TL/C/5171-8

NSC800 HARDWARE

6.0 Pin Descriptions

6.1 INPUT SIGNALS

Reset Input ($\overline{\text{RESET IN}}$): Active low. Sets A (8–15) and AD (0–7) to TRI-STATE® (high impedance). Clears the contents of PC, I and R registers, disables interrupts, and activates reset out.

Bus Request ($\overline{\text{BREQ}}$): Active low. Used when another device requests the system bus. The NSC800 recognizes $\overline{\text{BREQ}}$ at the end of the current machine cycle, and sets A(8–15), AD(0–7), IO/$\overline{\text{M}}$, $\overline{\text{RD}}$, and $\overline{\text{WR}}$ to the high impedance state. $\overline{\text{RFSH}}$ is high during a bus request cycle. The CPU acknowledges the bus request via the $\overline{\text{BACK}}$ output signal.

Non-Maskable Interrupt ($\overline{\text{NMI}}$): Active low. The non-maskable interrupt, generated by the peripheral device(s), is the highest priority interrupt. The edge sensitive interrupt requires only a pulse to set an internal flip-flop which generates the internal interrupt request. The $\overline{\text{NMI}}$ flip-flop is monitored on the same clock edge as the other interrupts. It must also meet the minimum set-up time spec for the interrupt to be accepted in the current machine instruction. When the processor accepts the interrupt the flip-flop resets automatically. Interrupt execution is independent of the interrupt enable flip-flop. $\overline{\text{NMI}}$ execution results in saving the PC on the stack and automatic branching to restart address X'0066 in memory.

Restart Interrupts, A, B, C ($\overline{\text{RSTA}}$, $\overline{\text{RSTB}}$, $\overline{\text{RSTC}}$): Active low level sensitive. The CPU recognizes restarts generated by the peripherals at the end of the current instruction, if their respective interrupt enable and master enable bits are set. Execution is identical to $\overline{\text{NMI}}$ except the interrupts vector to the following restart addresses:

Name	Restart Address (X')
$\overline{\text{NMI}}$	0066
$\overline{\text{RSTA}}$	003C
$\overline{\text{RSTB}}$	0034
$\overline{\text{RSTC}}$	002C
$\overline{\text{INTR}}$ (Mode 1)	0038

The order of priority is fixed. The list above starts with the highest priority.

Interrupt Request ($\overline{\text{INTR}}$): Active low, level sensitive. The CPU recognizes an interrupt request at the end of the current instruction provided that the interrupt enable and master interrupt enable bits are set. $\overline{\text{INTR}}$ is the lowest priority interrupt. Program control selects one of three response modes which determines the method of servicing $\overline{\text{INTR}}$ in conjunction with $\overline{\text{INTA}}$. See Interrupt Control.

Wait (WAIT): Active low. When set low during $\overline{\text{RD}}$, $\overline{\text{WR}}$ or $\overline{\text{INTA}}$ machine cycles (during the $\overline{\text{WR}}$ machine cycle, wait must be valid prior to write going active) the CPU extends its machine cycle in increments of t (wait) states. The wait machine cycle continues until the $\overline{\text{WAIT}}$ input returns high.

The wait strobe input will be accepted only during machine cycles that have $\overline{\text{RD}}$, $\overline{\text{WR}}$ or $\overline{\text{INTA}}$ strobes and during the machine cycle immediately after an interrupt has been accepted by the CPU. The later cycle has its RD strobe suppressed but it will still accept the wait.

Power-Save ($\overline{\text{PS}}$): Active low. $\overline{\text{PS}}$ is sampled during the last t state of the current instruction cycle. When $\overline{\text{PS}}$ is low, the CPU stops executing at the end of current instruction and keeps itself in the low-power mode. Normal operation resumes when $\overline{\text{PS}}$ returns high (see Power Save Feature description).

CRYSTAL (X_{IN}, X_{OUT}): X_{IN} can be used as an external clock input. A crystal can be connected across X_{IN} and X_{OUT} to provide a source for the system clock.

6.2 OUTPUT SIGNALS

Bus Acknowledge ($\overline{\text{BACK}}$): Active low. $\overline{\text{BACK}}$ indicates to the bus requesting device that the CPU bus and its control signals are in the TRI-STATE mode. The requesting device then commands the bus and its control signals.

Address Bits 8–15 [A(8–15)]: Active high. These are the most significant 8 bits of the memory address during a memory instruction. During an I/O instruction, the port address on the lower 8 address bits gets duplicated onto A(8–15). During a BREQ/BACK cycle, the A(8–15) bus is in the TRI-STATE mode.

Reset Out (RESET OUT): Active high. When RESET OUT is high, it indicates the CPU is being reset. This signal is normally used to reset the peripheral devices.

Input/Output/Memory (IO/$\overline{\text{M}}$): An active high on the IO/$\overline{\text{M}}$ output signifies that the current machine cycle is an input/output cycle. An active low on the IO/$\overline{\text{M}}$ output signifies that the current machine cycle is a memory cycle. It is TRI-STATE during $\overline{\text{BREQ}}$/$\overline{\text{BACK}}$ cycles.

Refresh ($\overline{\text{RFSH}}$): Active low. The refresh output indicates that the dynamic RAM refresh cycle is in progress. $\overline{\text{RFSH}}$ goes low during T3 and T4 states of all M1 cycles. During the refresh cycle, AD(0–7) has the refresh address and A(8–15) indicates the interrupt vector register data. $\overline{\text{RFSH}}$ is high during $\overline{\text{BREQ}}$/$\overline{\text{BACK}}$ cycles.

Address Latch Enable (ALE): Active high. ALE is active only during the T1 state of any M cycle and also T3 state of the M1 cycle. The high to low transition of ALE indicates that a valid memory, I/O or refresh address is available on the AD(0–7) lines.

Read Strobe ($\overline{\text{RD}}$): Active low. The CPU receives data via the AD(0–7) lines on the trailing edge of the $\overline{\text{RD}}$ strobe. The $\overline{\text{RD}}$ line is in the TRI-STATE mode during $\overline{\text{BREQ}}$/$\overline{\text{BACK}}$ cycles.

Write Strobe ($\overline{\text{WR}}$): Active low. The CPU sends data via the AD(0–7) lines while the $\overline{\text{WR}}$ strobe is low. The $\overline{\text{WR}}$ line is in the TRI-STATE mode during $\overline{\text{BREQ}}$/$\overline{\text{BACK}}$ cycles.

Clock (CLK): CLK is the output provided for use as a system clock. The CLK output is a square wave at one half the input frequency.

Interrupt Acknowledge ($\overline{\text{INTA}}$): Active low. This signal strobes the interrupt response vector from the interrupting peripheral devices onto the AD(0–7) lines. $\overline{\text{INTA}}$ is active during the M1 cycle immediately following the t state where the CPU recognized the $\overline{\text{INTR}}$ interrupt request.

Two of the three interrupt request modes use $\overline{\text{INTA}}$. In mode 0 one to four $\overline{\text{INTA}}$ signals strobe a one to four byte instruction onto the AD(0–7) lines. In mode 2 one $\overline{\text{INTA}}$ signal strobes the lower byte of an interrupt response vector onto the bus. In mode 1, $\overline{\text{INTA}}$ is inactive and the CPU response to $\overline{\text{INTR}}$ is the same as for an NMI or restart interrupt.

6.0 Pin Descriptions (Continued)

Status (SO, S1): Bus status outputs provide encoded information regarding the current M cycle as follows:

Machine Cycle	Status			Control	
	S0	S1	IO/$\overline{M}$	$\overline{RD}$	$\overline{WR}$
Opcode Fetch	1	1	0	0	1
Memory Read	0	1	0	0	1
Memory Write	1	0	0	1	0
I/O Read	0	1	1	0	1
I/O Write	1	0	1	1	0
Halt*	0	0	0	0	1
Internal Operation*	0	1	0	1	1
Acknowledge of Int**	1	1	0	1	1

*ALE is not suppressed in this cycle.

**This is the cycle that occurs immediately after the CPU accepts an interrupt ($\overline{RSTA}$, $\overline{RSTB}$, $\overline{RSTC}$, $\overline{INTR}$, $\overline{NMI}$).

Note 1: During halt, CPU continues to do dummy opcode fetch from location following the halt instruction with a halt status. This is so CPU can continue to do its dynamic RAM refresh.

Note 2: No early status is provided for interrupt or hardware restarts.

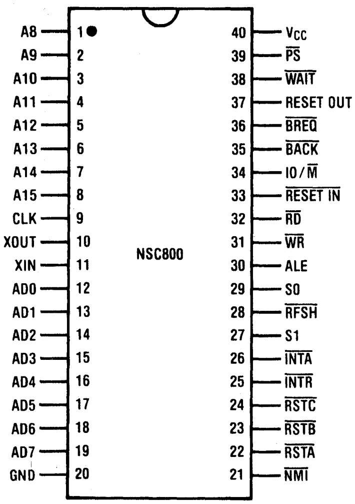
6.3 INPUT/OUTPUT SIGNALS

Multiplexed Address/Data [AD(0–7)]: Active high

At $\overline{RD}$ Time:	Input data to CPU.
At $\overline{WR}$ Time:	Output data from CPU.
At Falling Edge of ALE Time:	Least significant byte of address during memory reference cycle. 8-bit port address during I/O reference cycle.
During $\overline{BREQ}$/$\overline{BACK}$ Cycle:	High impedance.

7.0 Connection Diagrams

Dual-In-Line Package

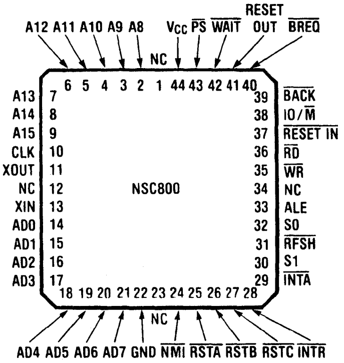


Top View

TL/C/5171–10

Order Number NSC800D or N
See NS Package D40C or N40A

Chip Carrier Package



Top View

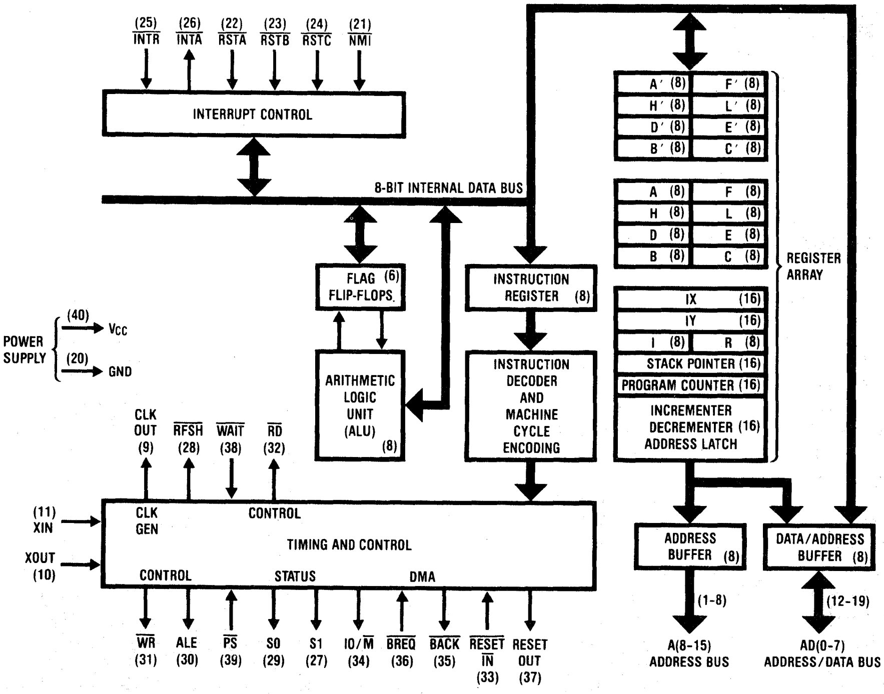
TL/C/5171–11

Order Number NSC800E or V
See NS Package E44B or V44A

8.0 Functional Description

This section reviews the CPU architecture shown below, focusing on the functional aspects from a hardware perspective, including timing details.

As illustrated in *Figure 1*, the NSC800 is an 8-bit parallel device. The major functional blocks are: the ALU, register array, interrupt control, timing and control logic. These areas are connected via the 8-bit internal data bus. Detailed descriptions of these blocks ae provided in the following sections.



TL/C/5171-9

Note: Applicable pinout for 40-pin dual-in-line package within parentheses

FIGURE 1. NSC800 CPU Functional Block Diagram

8.0 Functional Description (Continued)

8.1 REGISTER ARRAY

The NSC800 register array is divided into two parts: the dedicated registers and the working registers, as shown in *Figure 2*.

Main Reg. Set		Alternate Reg. Set		
Accumulator	**Flags**	**Accumulator**	**Flags**	
A	F	A′	F′	Working Registers
B	C	B′	C′	
D	E	D′	E′	
H	L	H′	L′	

Interrupt Vector I	Memory Refresh R	Dedicated Registers
Index Register IX		
Index Register IV		
Stack Pointer SP		
Program Counter PC		

FIGURE 2. NSC800 Register Array

8.2 DEDICATED REGISTERS

There are 6 dedicated registers in the NSC800: two 8-bit and four 16-bit registers (see *Figure 3*).

Although their contents are under program control, the program has no control over their operational functions, unlike the CPU working registers. The function of each dedicated register is described as follows:

CPU Dedicated Registers	
Program Counter PC	(16)
Stack Pointer SP	(16)
Index Register IX	(16)
Index Register IY	(16)
Interrupt Vector Register I	(8)
Memory Refresh Register R	(8)

FIGURE 3. Dedicated Registers

8.2.1 Program Counter (PC)

The program counter contains the 16-bit address of the current instruction being fetched from memory. The PC increments after its contents have been transferred to the address lines. When a program jump occurs, the PC receives the new address which overrides the incrementer.

There are many conditional and unconditional jumps, calls, and return instructions in the NSC800's instruction repertoire that allow easy manipulation of this register in controlling the program execution (i.e. JP NZ nn, JR Zd2, CALL NC, nn).

8.2.2 Stack Pointer (SP)

The 16-bit stack pointer contains the address of the current top of stack that is located in external system RAM. The stack is organized in a last-in, first-out (LIFO) structure. The pointer decrements before data is pushed onto the stack, and increments after data is popped from the stack.

Various operations store or retrieve, data on the stack. This, along with the usage of subroutine calls and interrupts, allows simple implementation of subroutine and interrupt nesting as well as alleviating many problems of data manipulation.

8.2.3 Index Register (IX and IY)

The NSC800 contains two index registers to hold independent, 16-bit base addresses used in the indexed addressing mode. In this mode, an index register, either IX or IY, contains a base address of an area in memory making it a pointer for data tables.

In all instructions employing indexed modes of operation, another byte acts as a signed two's complement displacement. This addressing mode enables easy data table manipulations.

8.2.4 Interrupt Register (I)

When the NSC800 provides a Mode 2 response to $\overline{\text{INTR}}$, the action taken is an indirect call to the memory location containing the service routine address. The pointer to the address of the service routine is formed by two bytes, the high-byte is from the I Register and the low-byte is from the interrupting peripheral. The peripheral always provides an even address for the lower byte (LSB=0). When the processor receives the lower byte from the peripheral it concatenates it in the following manner:

I Register	External byte	
8 bits		0

↑ The LSB of the external byte must be zero.

FIGURE 4a. Interrupt Register

The even memory location contains the low-order byte, the next consecutive location contains the high-order byte of the pointer to the beginning address of the interrupt service routine.

8.2.5 Refresh Register (R)

For systems that use dynamic memories rather than static RAM's, the NSC800 provides an integral 8-bit memory refresh counter. The contents of the register are incremented after each opcode fetch and are sent out on the lower portion of the address bus, along with a refresh control signal. This provides a totally transparent refresh cycle and does not slow down CPU operation.

The program can read and write to the R register, although this is usually done only for test purposes.

8.0 Functional Description (Continued)

8.3 CPU WORKING AND ALTERNATE REGISTER SETS

8.3.1 CPU Working Registers

The portion of the register array shown in *Figure 4b* represents the CPU working registers. These sixteen 8-bit registers are general-purpose registers because they perform a multitude of functions, depending on the instruction being executed. They are grouped together also due to the types of instructions that use them, particularly alternate set operations.

The F (flag) register is a special-purpose register because its contents are more a result of machine status rather than program data. The F register is included because of its interaction with the A register, and its manipulations in the alternate register set operations.

8.3.2 Alternate Registers

The NSC800 registers designated as CPU working registers have one common feature: the existence of a duplicate register in an alternate register set. This architectural concept simplifies programming during operations such as interrupt response, when the machine status represented by the contents of the registers must be saved.

The alternate register concept makes one set of registers available to the programmer at any given time. Two instructions (EX AF, A'F' and EXX), exchange the current working set of registers with their alternate set. One exchange between the A and F registers and their respective duplicates (A' and F') saves the primary status information contained in the accumulator and the flag register. The second exchange instruction performs the exchange between the remaining registers, B, C, D, E, H, and L, and their respective alternates B', C', D', E', H', and L'. This essentially saves the contents of the original complement of registers while providing the programmer with a usable alternate set.

CPU Main Working Register Set

Accumulator A	(8)	Flags F	(8)
Register B	(8)	Register C	(8)
Register D	(8)	Register E	(8)
Register H	(8)	Register L	(8)

CPU Alternate Working Register Set

Accumulator A'	(8)	Flags F'	(8)
Register B'	(8)	Register C'	(8)
Register D'	(8)	Register E'	(8)
Register H'	(8)	Register L'	(8)

FIGURE 4b. CPU Working and Alternate Registers

8.4 REGISTER FUNCTIONS

8.4.1 Accumulator (A Register)

The A register serves as a source or destination register for data manipulation instructions. In addition, it serves as the accumulator for the results of 8-bit arithmetic and logic operations.

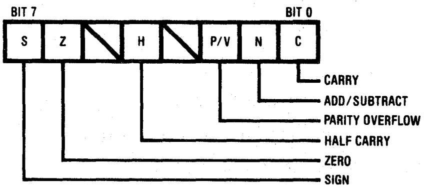
The A register also has a special status in some types of operations; that is, certain addressing modes are reserved for the A register only, although the function is available for all the other registers. For example, any register can be loaded by immediate, register indirect, or indexed addressing modes. The A register, however, can also be loaded via an additional register indirect addressing.

Another special feature of the A register is that it produces more efficient memory coding than equivalent instruction functions directed to other registers. Any register can be rotated; however, while it requires a two-byte instruction to normally rotate any register, a single-byte instruction is available for rotating the contents of the accumulator (A register).

8.4.2 F Register - Flags

The NSC800 flag register consists of six status bits that contain information regarding the results of previous CPU operations. The register can be read by pushing the contents onto the stack and then reading it, however, it cannot be written to. It is classified as a register because of its affiliation with the accumulator and the existence of a duplicate register for use in exchange instructions with the accumulator.

Of the six flags shown in *Figure 5*, only four can be directly tested by the programmer via conditional jump, call, and return instructions. They are the Sign (S), Zero (Z), Parity/Overflow (P/V), and Carry (C) flags. The Half Carry (H) and Add/Subtract (N) flags are used for internal operations related to BCD arithmetic.



TL/C/5171-23

FIGURE 5. Flag Register

8.0 Functional Description (Continued)

8.4.3 Carry (C)

A carry from the highest order bit of the accumulator during an add instruction, or a borrow generated during a subtraction instruction sets the carry flag. Specific shift and rotate instructions also affect this bit.

Two specific instructions in the NSC800 instruction repertoire set (SCF) or complement (CCF) the carry flag.

Other operations that affect the C flag are as follows:

- Adds
- Subtracts
- Logic Operations (always resets C flag)
- Rotate Accumulator
- Rotate and Shifts
- Decimal Adjust
- Negation of Accumulator

Other operations do not affect the C flag.

8.4.4 Adds/Subtract (N)

This flag is used in conjunction with the H flag to ensure that the proper BCD correction algorithm is used during the decimal adjust instruction (DAA). The correction algorithm depends on whether an add or subtract was previously done with BCD operands.

The operations that set the N flag are:

- Subtractions
- Decrements (8-bit)
- Complementing of the Accumulator
- Block I/O
- Block Searches
- Negation of the Accumulator

The operations that reset the N flag are:

- Adds
- Increments
- Logic Operations
- Rotates
- Set and Complement Carry
- Input Register Indirect
- Block Transfers
- Load of the I or R Registers
- Bit Tests

Other operations do not affect the N flag.

8.4.5 Parity/Overflow (P/V)

The Parity/Overflow flag is a dual-purpose flag that indicates results of logic and arithmetic operations. In logic operations, the P/V flag indicates the parity of the result; the flag is set (high) if the result is even, reset (low) if the result is odd. In arithmetic operations, it represents an overflow condition when the result, interpreted as signed two's complement arithmetic, is out of range for the eight-bit accumulator (i.e. −128 to +127).

The following operations affect the P/V flag according to the parity of the result of the operation:

- Logic Operations
- Rotate and Shift
- Rotate Digits
- Decimal Adjust
- Input Register Indirect

The following operations affect the P/V flag according to the overflow result of the operation.

- Adds (16 bit with carry, 8-bit with/without carry)
- Subtracts (16 bit with carry, 8-bit with/without carry)
- Increments and Decrements
- Negation of Accumulator

The P/V flag has no significance immediately after the following operations.

- Block I/O
- Bit Tests

In block transfers and compares, the P/V flag indicates the status of the BC register, always ending in the reset state after an auto repeat of a block move. Other operations do not affect the P/V flag.

8.4.6 Half Carry (H)

This flag indicates a BCD carry, or borrow, result from the low-order four bits of operation. It can be used to correct the results of a previously packed decimal add, or subtract, operation by use of the Decimal Adjust Instruction (DAA).

The following operations affect the H flag:

- Adds (8-bit)
- Subtracts (8-bit)
- Increments and Decrements
- Decimal Adjust
- Negation of Accumulator
- Always Set by: Logic AND
  Complement Accumulator
  Bit Testing
- Always Reset By: Logic OR's and XOR's
  Rotates and Shifts
  Set Carry
  Input Register Indirect
  Block Transfers
  Loads of I and R Registers

The H flag has no significance immediately after the following operations.

- 16-bit Adds with/without carry
- 16-Bit Subtracts with carry
- Complement of the carry
- Block I/O
- Block Searches

Other operations do not affect the H flag.

8.0 Functional Description (Continued)

8.4.7 Zero Flag (Z)

Loading a zero in the accumulator or when a zero results from an operation sets the zero flag.

The following operations affect the zero flag.

- Adds (16-bit with carry, 8-bit with/without carry)
- Subtracts (16-bit with carry, 8-bit with/without carry)
- Logic Operations
- Increments and Decrements
- Rotate and Shifts
- Rotate Digits
- Decimal Adjust
- Input Register Indirect
- Block I/O (always set after auto repeat block I/O)
- Block Searches
- Load of I and R Registers
- Bit Tests
- Negation of Accumulator

The Z flag has no signficance immediately after the following operations:

- Block Transfers

Other operations do not affect the zero flag.

8.4.8 Sign Flag (S)

The sign flag stores the state of bit 7 (the most-significant bit and sign bit) of the accumulator following an arithmetic operation. This flag is of use when dealing with signed numbers.

The sign flag is affected by the following operation according to the result:

- Adds (16-bit with carry, 8-bit with/without carry)
- Subtracts (16-bit with carry, 8-bit with/without carry)
- Logic Operations
- Increments and Decrements
- Rotate and Shifts
- Rotate Digits
- Decimal Adjust
- Input Register Indirect
- Block Search
- Load of I and R Registers
- Negation of Accumulator

The S flag has no significance immediately after the following operations:

- Block I/O
- Block Transfers
- Bit Tests

Other operations do not affect the sign bit.

8.4.9 Additional General-Purpose Registers

The other general-purpose registers are the B, C, D, E, H and L registers and their alternate register set, B', C', D', E', H' and L'. The general-purpose registers can be used interchangeably.

In addition, the B and C registers perform special functions in the NSC800 expanded I/O capabilities, particularly block I/O operations. In these functions, the C register can address I/O ports; the B register provides a counter function when used in the register indirect address mode.

When used with the special condition jump instruction (DJNZ) the B register again provides the counter function.

8.4.10 Alternate Configurations

The six 8-bit general purpose registers (B,C,D,E,H,L) will combine to form three 16-bit registers. This occurs by concatenating the B and C registers to form the BC register, the D and E registers form the DE register, and the H and L registers form the HL register.

Having these 16-bit registers allows 16-bit data handling, thereby expanding the number of 16-bit registers available for memory addressing modes. The HL register typically provides the pointer address for use in register indirect addressing of the memory.

The DE register provides a second memory pointer register for the NSC800's powerful block transfer operations. The BC register also provides an assist to the block transfer operations by acting as a byte-counter for these operations.

8.5 ARITHMETIC-LOGIC UNIT (ALU)

The arithmetic, logic and rotate instructions are performed by the ALU. The ALU internally communicates with the registers and data buffer on the 8-bit internal data bus.

8.6 INSTRUCTION REGISTER AND DECODER

During an opcode fetch, the first byte of an instruction is transferred from the data buffer (i.e. its on the internal data bus) to the instruction register. The instruction register feeds the instruction decoder, which gated by timing signals, generates the control signals that read or write data from or to the registers, control the ALU and provide all required external control signals.

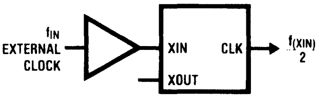
9.0 Timing and Control

9.1 INTERNAL CLOCK GENERATOR

An inverter oscillator contained on the NSC800 chip provides all necessary timing signals. The chip operation frequency is equal to one half of the frequency of this oscillator.

The oscillator frequency can be controlled by one of the following methods:

1. Leaving the X_{OUT} pin unterminated and driving the X_{IN} pin with an externally generated clock as shown in *Figure 6*. When driving X_{IN} with a square wave, the minimum duty cycle is 30% high.

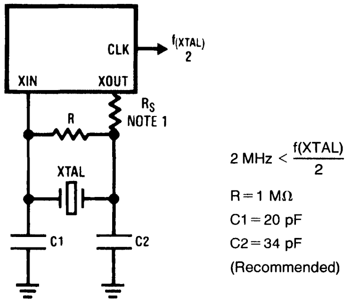


TL/C/5171-13

FIGURE 6. Use of External Clock

2. Connecting a crystal with the proper biasing network between X_{IN} and X_{OUT} as shown in *Figure 7*. Recommended crystal is a parallel resonance AT cut crystal.

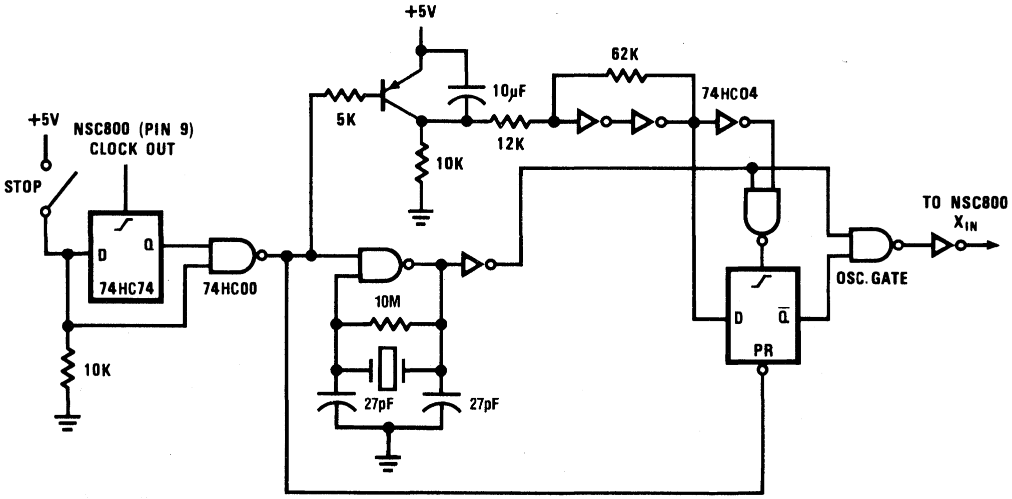
Note 1: If the crystal frequency is 2 MHz or less a series resistor, R_S, (470Ω to 1500Ω) should be connected between X_{OUT} and R, XTAL and C_Z. Additionally, the capacitance of C1 and C2 should be increased by 2 to 3 times the recommended value.



TL/C/5171-14

FIGURE 7. Use Of Crystal

The CPU has a minimum clock frequency input (@ X_{IN}) of 300 kHz, which results in 150 kHz system clock speed. All registers internal to the chip are static, however there is dynamic logic which limits the minimum clock speed. The input clock can be stopped without fear of losing any data or damaging the part. You stop it in the phase of the clock that has X_{IN} low and CLK OUT high. When restarting the CPU, precautions must be taken so that the input clock meets these minimum specification. Once started, the CPU will continue operation from the same location at which it was stopped. During DC operation of the CPU, typical current drain will be 2 mA. This current drain can be reduced by placing the CPU in a wait state during an opcode fetch cycle then stopping the clock. For clock stop circuit, see *Figure 8*.



TL/C/5171-36

FIGURE 8. Clock Stop Circuit

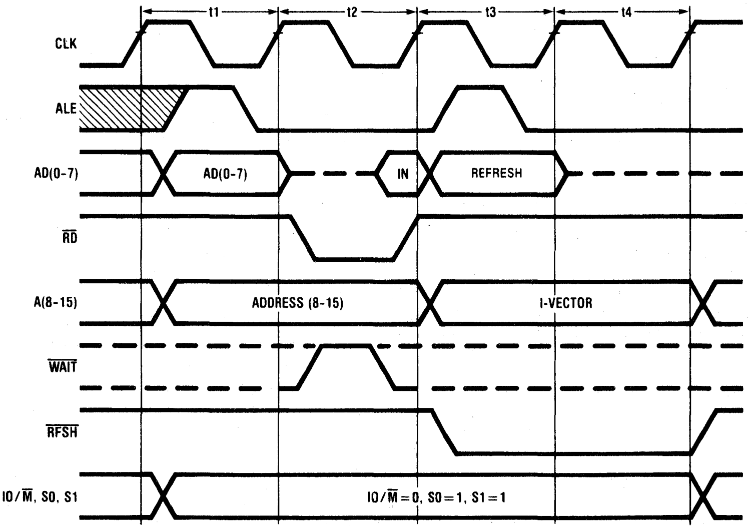
9.0 Timing and Control (Continued)

9.2 CPU TIMING

The NSC800 uses a multiplexed bus for data and addresses. The 16-bit address bus is divided into a high-order 8-bit address bus that handles bits 8–15 of the address, and a low-order 8-bit multiplexed address/data bus that handles bits 0–7 of the address and bits 0–7 of the data. Strobe outputs from the NSC800 (ALE, $\overline{\text{RD}}$ and $\overline{\text{WR}}$) indicate when a valid address or data is present on the bus. IO/$\overline{\text{M}}$ indicates whether the ensuing cycle accesses memory or I/O.

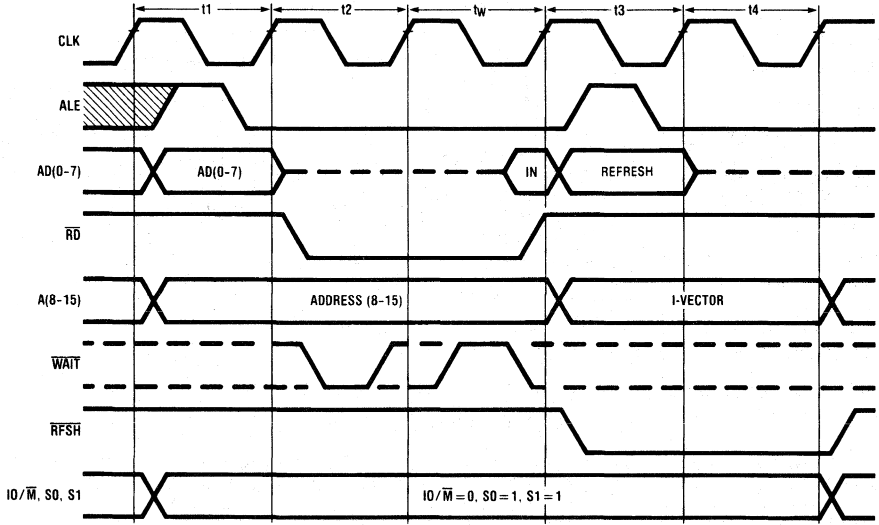
During an input or output instruction, the CPU duplicates the lower half of the address [AD(0–7)] onto the upper address bus [A(8–15)]. The eight bits of address will stay on A(8–15) for the entire machine cycle and can be used for chip selection directly.

Figure 9 illustrates the timing relationship for opcode fetch cycles with and without a wait state.



TL/C/5171-15

FIGURE 9a. Opcode Fetch Cycles without $\overline{\text{WAIT}}$ States

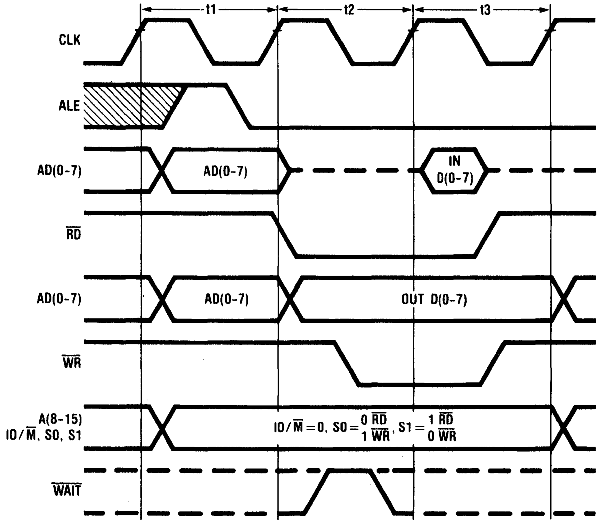


TL/C/5171-16

FIGURE 9b. Opcode Fetch Cycles with $\overline{\text{WAIT}}$ States

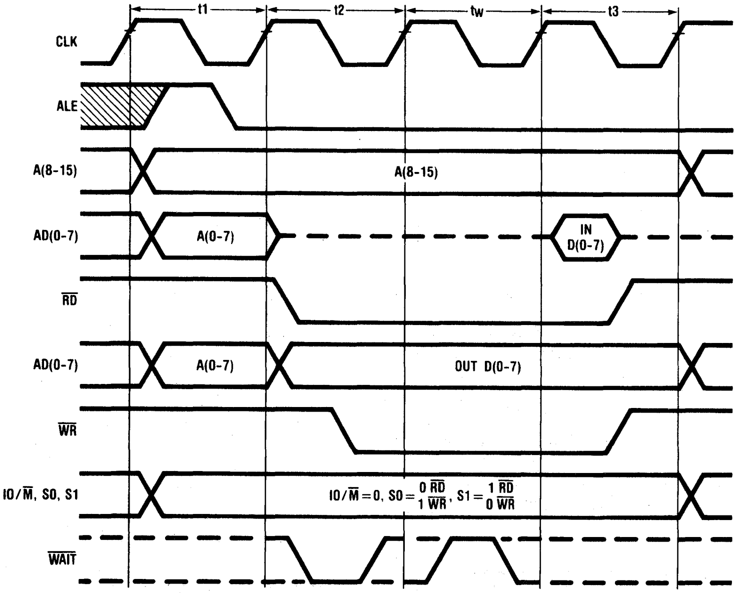
9.0 Timing and Control (Continued)

During the opcode fetch, the CPU places the contents of the PC on the address bus. The falling edge of ALE indicates a valid address on the AD(0–7) lines. The $\overline{\text{WAIT}}$ input is sampled during t_2 and if active causes the NSC800 to insert a wait state (t_w). WAIT is sampled again during t_w so that when it goes inactive, the CPU continues its opcode fetch by latching in the data on the rising edge of $\overline{\text{RD}}$ from the AD(0–7) lines. During t_3, $\overline{\text{RFSH}}$ goes active and AD(0–7) has the dynamic RAM refresh address from register R and A(8–15) the interrupt vector from register I.



TL/C/5171-17

FIGURE 10a. Memory Read/Write Cycles without $\overline{\text{WAIT}}$ States



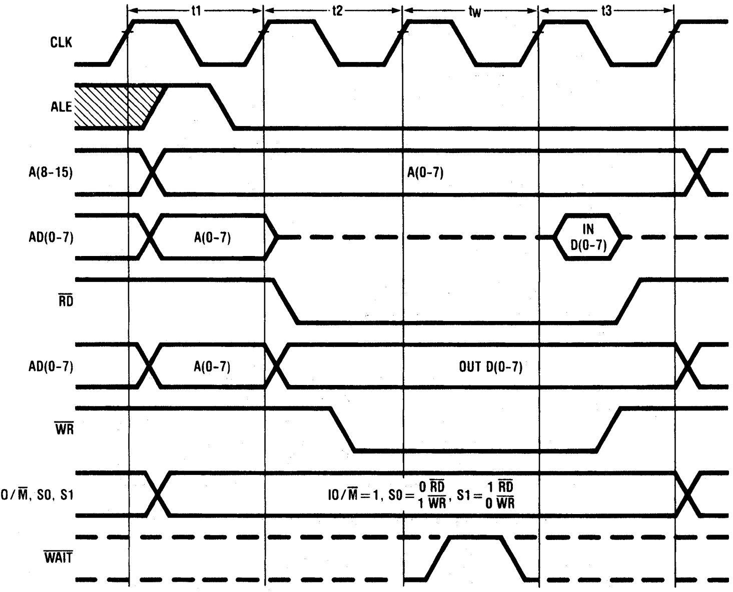
TL/C/5171-18

FIGURE 10b. Memory Read and Write with $\overline{\text{WAIT}}$ States

9.0 Timing and Control (Continued)

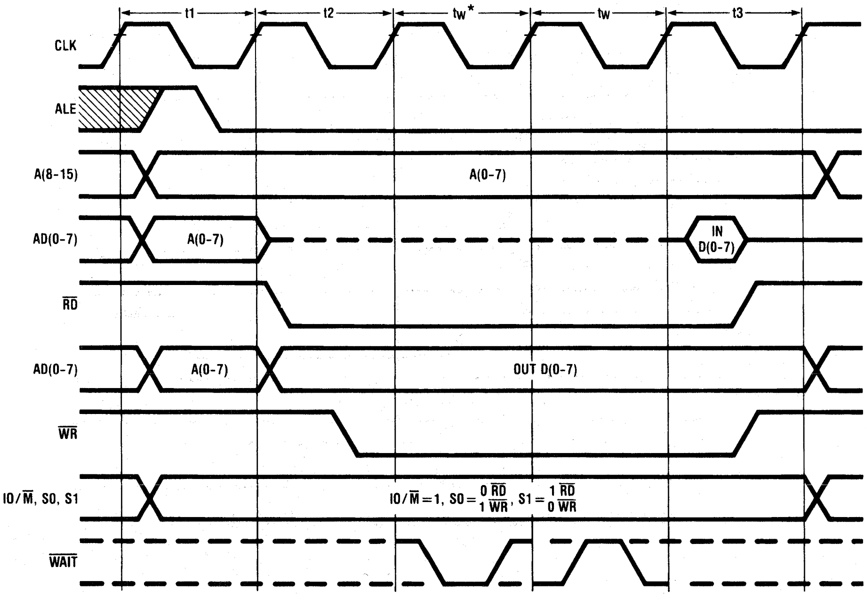
Figure 10 shows the timing for memory read (other than opcode fetchs) and write cycles with and without a wait state. The $\overline{RD}$ stobe is widened by $\frac{t}{2}$ (half the machine state) for memory reads so that the actual latching of the input data occurs later.

Figure 11 shows the timing for input and output cycles with and without wait states. The CPU automatically inserts one wait state into each I/O instruction to allow sufficient time for an I/O port to decode the address.


t1
t2
tw
t3
CLK
ALE
A(8-15)
A(0-7)
AD(0-7)
A(0-7)
IN D(0-7)
$\overline{RD}$
AD(0-7)
A(0-7)
OUT D(0-7)
$\overline{WR}$
IO/$\overline{M}$, S0, S1
IO/$\overline{M}$=1, S0=$\frac{0\ \overline{RD}}{1\ \overline{WR}}$, S1=$\frac{1\ \overline{RD}}{0\ \overline{WR}}$
$\overline{WAIT}$

TL/C/5171-19

FIGURE 11a. Input and Output Cycles without $\overline{WAIT}$ States


t1
t2
tw*
tw
t3
CLK
ALE
A(8-15)
A(0-7)
AD(0-7)
A(0-7)
IN D(0-7)
$\overline{RD}$
AD(0-7)
A(0-7)
OUT D(0-7)
$\overline{WR}$
IO/$\overline{M}$, S0, S1
IO/$\overline{M}$=1, S0=$\frac{0\ \overline{RD}}{1\ \overline{WR}}$, S1=$\frac{1\ \overline{RD}}{0\ \overline{WR}}$
$\overline{WAIT}$

TL/C/5171-20

*$\overline{WAIT}$ state automatically inserted during IO operation.

FIGURE 11b. Input and Output Cycles with $\overline{WAIT}$ States

9.0 Timing and Control (Continued)

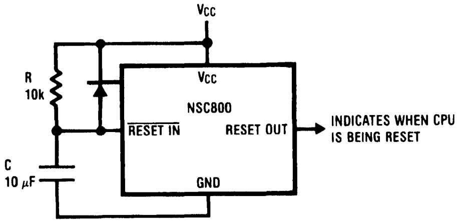
9.3 INITIALIZATION

$\overline{\text{RESET IN}}$ initializes the NSC800; RESET OUT initializes the peripheral components. The Schmitt trigger at the $\overline{\text{RESET IN}}$ input facilitates using an R-C network reset scheme during power up (see *Figure 12*).

To ensure proper power-up conditions for the NSC800, the following power-up and initialization procedure is recommended:

1. Apply power (V_{CC} and GND) and set $\overline{\text{RESET IN}}$ active (low). Allow sufficient time (approximately 30 ms if a crystal is used) for the oscillator and internal clocks to stabilize. $\overline{\text{RESET IN}}$ must remain low for at least 3t state (CLK) times. RESET OUT goes high as soon as the active $\overline{\text{RESET IN}}$ signal is clocked into the first flip-flop after the on-chip Schmitt trigger. RESET OUT signal is available to reset the peripherals.
2. Set $\overline{\text{RESET IN}}$ high. RESET OUT then goes low as the inactive $\overline{\text{RESET IN}}$ signal is clocked into the first flip-flop after the on-chip Schmitt trigger. Following this the CPU initiates the first opcode fetch cycle.

Note: The NSC800 initialization includes: Clear PC to X'0000 (the first opcode fetch, therefore, is from memory location X'0000). Clear registers I (Interrupt Vector Base) and R (Refresh Counter) to X'00. Clear interrupt control register bits IEA, IEB and IEC. The interrupt control bit IEI is set to 1 to maintain INS8080A/Z80A compatibility (see INTERRUPTS for more details). The CPU disables maskable interrupts and enters $\overline{\text{INTR}}$ Mode 0. While $\overline{\text{RESET IN}}$ is active (low), the A(8–15) and AD(0–7) lines go to high impedance (TRI-STATE) and all CPU strobes go to the inactive state (see *Figure 13*).

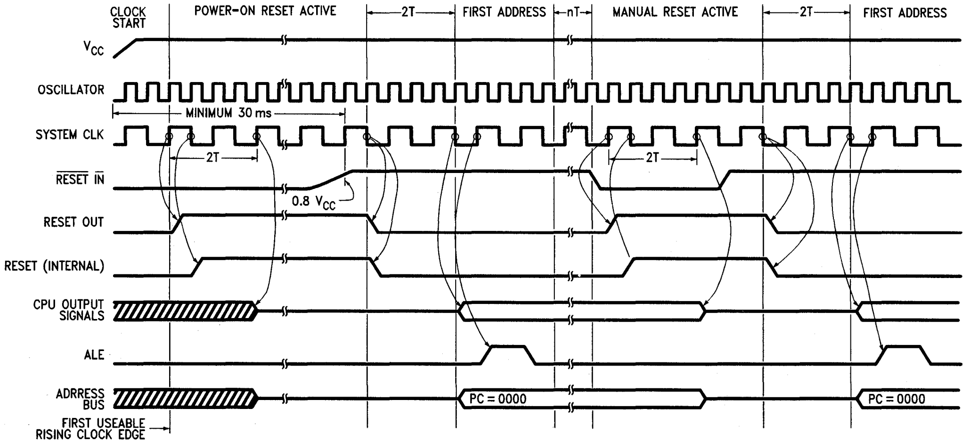


TL/C/5171–21

FIGURE 12. Power-On Reset

9.4 POWER-SAVE FEATURE

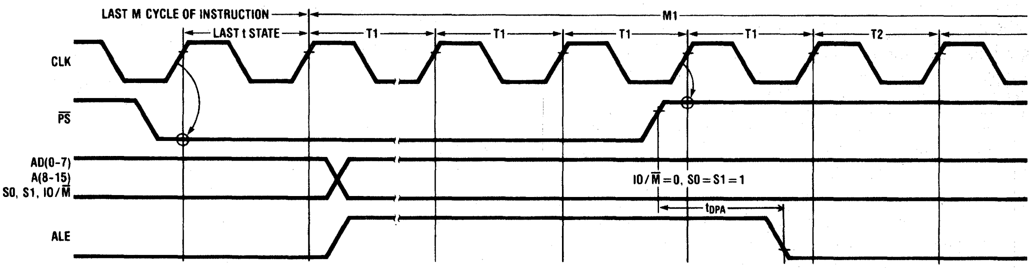
The NSC800 provides a unique power-save mode by the means of the $\overline{\text{PS}}$ pin. $\overline{\text{PS}}$ input is sampled at the last t state of the last M cycle of an instruction. After recognizing an active (low) level on $\overline{\text{PS}}$, The NSC800 stops its internal clocks, thereby reducing its power dissipation to one half of operating power, yet maintaining all register values and internal control status. The NSC800 keeps its oscillator running, and makes the CLK signal available to the system. When in power-save the ALE strobe will be stopped high and the address lines [AD(0–7), A(8–15)] will indicate the next machine address. When $\overline{\text{PS}}$ returns high, the opcode fetch (or M1 cycle) of the CPU begins in a normal manner. Note this M1 cycle could also be an interrupt acknowledge cycle if the NSC800 was interrupted simultaneously with $\overline{\text{PS}}$ (i.e. $\overline{\text{PS}}$ has priority over a simultaneously occurring interrupt). However, interrupts are not accepted during power save. *Figure 14* illustrates the power save timing.



TL/C/5171–74

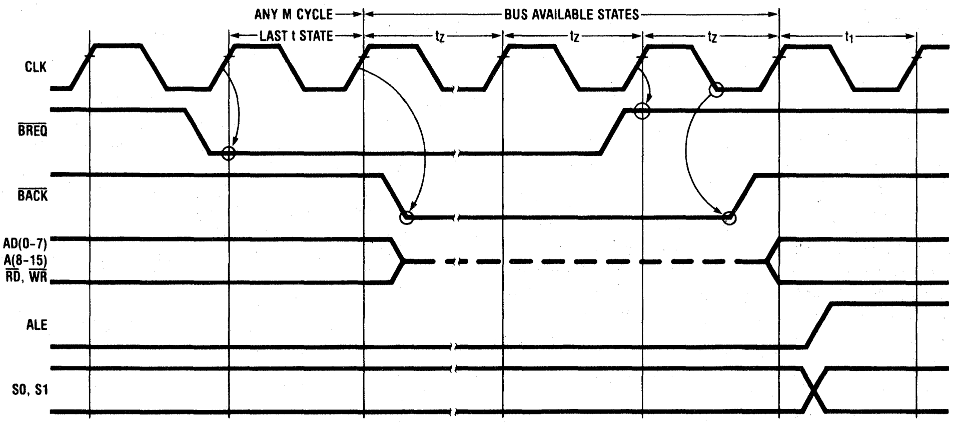
FIGURE 13. NSC800 Signals During Power-On and Manual Reset

9.0 Timing and Control (Continued)



TL/C/5171-28

FIGURE 14. NSC800 Power-Save



TL/C/5171-22

*S0, S1 during BREQ will indicate same machine cycle as during the cycle when BREQ was accepted.

t_Z=time states during which bus and control signals are in high impedance mode.

FIGURE 15. Bus Acknowledge Cycle

In the event BREQ is asserted (low) at the end of an instruction cycle and PS is active simultaneously, the following occurs:

1. The NSC800 will go into BACK cycle.
2. Upon completion of BACK cycle if PS is still active the CPU will go into power-save mode.

9.5 BUS ACCESS CONTROL

Figure 15 illustrates bus access control in the NSC800. The external device controller produces an active BREQ signal that requests the bus. When the CPU responds with BACK then the bus and related control strobes go to high impedance (TRI-STATE) and the RFSH signal remains high. It should be noted that (1) BREQ is sampled at the last t state of any M machine cycle only. (2) The NSC800 will not acknowledge any interrupt/restart requests, and will not peform any dynamic RAM refresh functions until after BREQ input signal is inactive high. (3) BREQ signal has priority over all interrupt request signals, should BREQ and interrupt request become active simultaneously. Therefore, interrupts latched at the end of the instruction cycle will be serviced after a simultaneously occurring BREQ. NMI is latched during an active BREQ.

9.6 INTERRUPT CONTROL

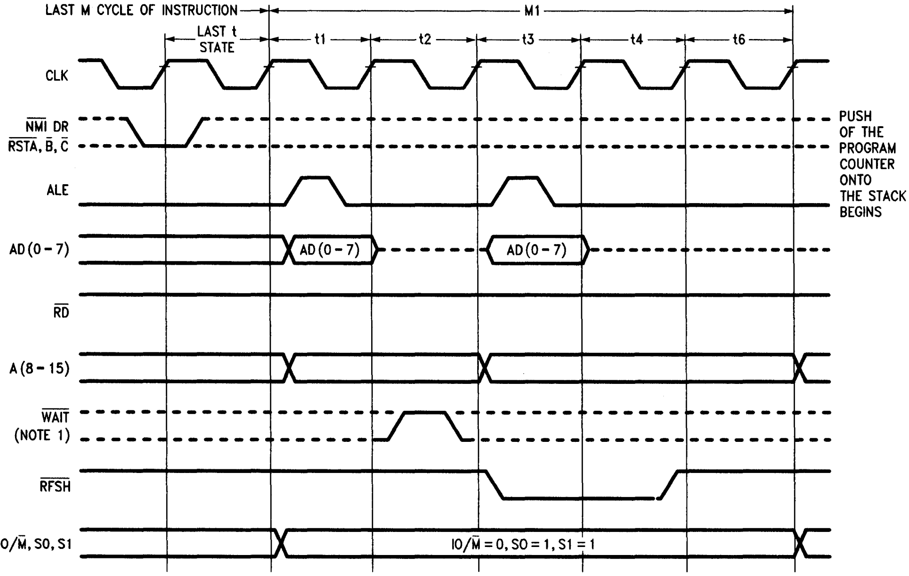
The NSC800 has five interrupt/restart inputs, four are maskable (RSTA, RSTB, RSTC, and INTR) and one is non-maskable (NMI). NMI has the highest priority of all interrupts; the user cannot disable NMI. After recognizing an active input on NMI, the CPU stops before the next instruction, pushes the PC onto the stack, and jumps to address X'0066, where the user's interrupt service routine is located (i.e., restart to memory location X'0066). NMI is intended for interrupts requiring immediate attention, such as power-down, control panel, etc.

RSTA, RSTB and RSTC are restart inputs, which, if enabled, execute a restart to memory location X'003C, X'0034, and X'002C, respectively. Note that the CPU response to the NMI and RST (A, B, C) request input is basically identical, except for the restored memory location. Unlike NMI, however, restart request inputs must be enabled.

Figure 16 illustrates NMI and RST interrupt machine cycles. M1 cycle will be a dummy opcode fetch cycle followed by M2 and M3 which are stack push operations. The following instruction then starts from the interrupts restart location.

Note: RD does *not* go low during this dummy opcode fetch. A unique indication of INTA can be decoded using 2 ALEs and RD.

9.0 Timing and Control (Continued)


LAST M CYCLE OF INSTRUCTION
M1
LAST t STATE
t1
t2
t3
t4
t6
CLK
$\overline{\text{NMI}}$ DR
$\overline{\text{RSTA}}$, $\overline{\text{B}}$, $\overline{\text{C}}$
PUSH OF THE PROGRAM COUNTER ONTO THE STACK BEGINS
ALE
AD (0 – 7)
AD (0 – 7)
AD (0 – 7)
$\overline{\text{RD}}$
A (8 – 15)
$\overline{\text{WAIT}}$ (NOTE 1)
$\overline{\text{RFSH}}$
IO/$\overline{\text{M}}$, S0, S1
IO/$\overline{\text{M}}$ = 0, S0 = 1, S1 = 1

TL/C/5171–24

Note 1: This is the only machine cycle that does not have an $\overline{\text{RD}}$, $\overline{\text{WR}}$, or $\overline{\text{INTA}}$ strobe but will accept a wait strobe.

FIGURE 16. Non-Maskable and Restart Interrupt Machine Cycle

The NSC800 also provides one more general purpose interrupt request input, $\overline{\text{INTR}}$. When enabled, the CPU responds to $\overline{\text{INTR}}$ in one of the three modes defined by instruction IM0, IM1, and IM2 for modes 0, 1, and 2, respectively. Following reset, the CPU automatically enables mode 0.

Interrupt ($\overline{\text{INTR}}$) Mode 0: The CPU responds to an interrupt request by providing an INTA (interrupt acknowledge) strobe, which can be used to gate an instruction from a peripheral onto the data bus. The CPU inserts two wait states during the first $\overline{\text{INTA}}$ cycle to allow the interrupting device (or its controller) ample time to gate the instruction and determine external priorities (*Figure 18*). This can be any instruction from one to four bytes. The most popular instruction is one-byte call (restart instruction) or a three-byte call (CALL NN instruction). If it is a three-byte call, the CPU issues a total of three $\overline{\text{INTA}}$ strobes. The last two (which do not include wait states) read NN.

Note: If the instruction stored in the ICU doesn't require the PC to be pushed onto the stack (eq. JP nn), then the PC will not be pushed.

Interrupt ($\overline{\text{INTR}}$) Mode 1: Similar to restart interrupts except the restart location is X'0038 (*Figure 18*).

Interrupt ($\overline{\text{INTR}}$) Mode 2: With this mode, the programmer maintains a table that contains the 16-bit starting address of every interrupt service routine. This table can be located anywhere in memory. When the CPU accepts a Mode 2 interrupt (*Figure 17*), it forms a 16-bit pointer to obtain the desired interrupt service routine starting address from the table. The upper 8 bits of this pointer are from the contents of the I register. The lower 8 bits of the pointer are supplied by the interrupting device with the LSB forced to zero. The programmer must load the interrupt vector prior to the interrupt occurring. The CPU uses the pointer to get the two adjacent bytes from the interrupt service routine starting address table to complete 16-bit service routine starting address. The first byte of each entry in the table is the least significant (low-order) portion of the address. The programmer must obviously fill this table with the desired addresses before any interrupts are to be accepted.

Note that the programmer can change this table at any time to allow peripherals to be serviced by different service routines. Once the interrupting device supplies the lower portion of the pointer, the CPU automatically pushes the program counter onto the stack, obtains the starting address from the table and does a jump to this address.

The interrupts have fixed priorities built into the NSC800 as:

$\overline{\text{NMI}}$	0066	(Highest Priority)
$\overline{\text{RSTA}}$	003C	
$\overline{\text{RSTB}}$	0034	
$\overline{\text{RSTC}}$	002C	
$\overline{\text{INTR}}$	0038	(Lowest Priority)

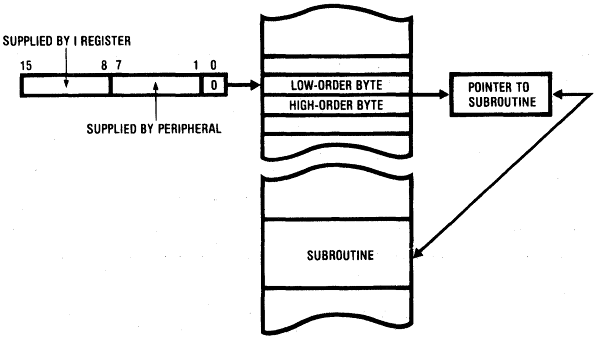
Interrupt Enable, Interrupt Disable. The NSC800 has two types of interrupt inputs, a non-maskable interrupt and four software maskable interrupts. The non-maskable interrupt ($\overline{\text{NMI}}$) cannot be disabled by the programmer and will be accepted whenever a peripheral device requests an interrupt. The $\overline{\text{NMI}}$ is usually reserved for important functions that must be serviced when they occur, such as imminent power failure. The programmer can selectively enable or disable maskable interrupts ($\overline{\text{INT}}$, $\overline{\text{RSTA}}$, $\overline{\text{RSTB}}$ and $\overline{\text{RSTC}}$). This selectivity allows the programmer to disable the maskable interrupts during periods when timing constraints don't allow program interruption.

There are two interrupt enable flip-flops (IFF_1 and IFF_2) on the NSC800. Two instructions control these flip-flops. Enable Interrupt (EI) and Disable Interrupt (DI). The state of IFF_1 determines the enabling or disabling of the maskable interrupts, while IFF_2 is used as a temporary storage location for the state of IFF_1.

9.0 Timing and Control (Continued)

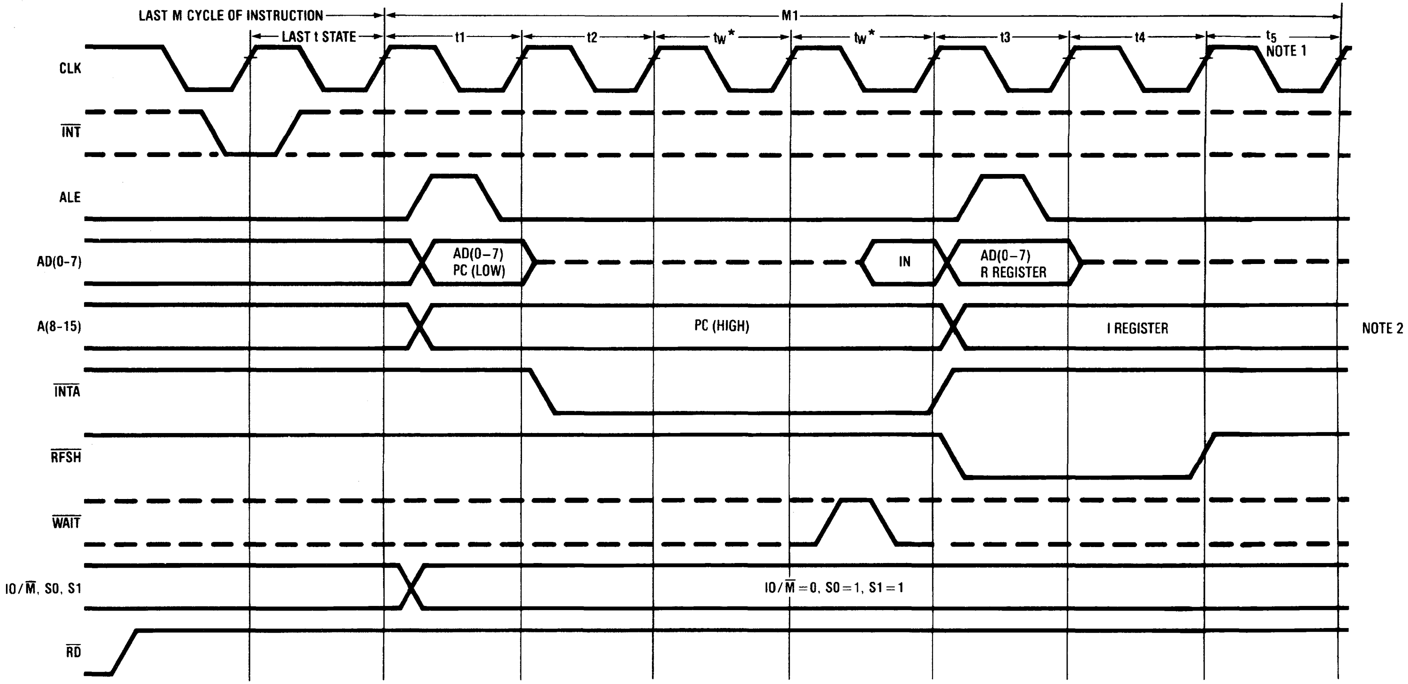
A reset to the CPU will force both IFF_1 and IFF_2 to the reset state disabling maskable interrupts. They can be enabled by an EI instruction at any time by the programmer. When an EI instruction is executed, any pending interrupt requests will not be accepted until after the instruction following EI has been executed. This single instruction delay is necessary in situations where the following instruction is a return instruction and interrupts must not be allowed until the return has been completed. The EI instruction sets both IFF_1 and IFF_2 to the enable state. When the CPU accepts an interrupt, both IFF_1 and IFF_2 are automatically reset, inhibiting further interrupts until the programmer wishes to issue a new EI instruction. Note that for all the previous cases, IFF_1 and IFF_2 are always equal.

The function of IFF_2 is to retain the status of IFF_1 when a non-maskable interrupt occurs. When a non-maskable interrupt is accepted, IFF_1 is reset to prevent further interrupts until reenabled by the programmer. Thus, after a non-maskable interrupt has been accepted, maskable interrupts are disabled but the previous state of IFF_1 is saved by IFF_2



TL/C/5171-27

FIGURE 17. Interrupt Mode 2



TL/C/5171-25

*t_W is the CPU generated WAIT state in response to an interrupt request.

Note 1: t5 will only occur in mode 1 and mode 2. During t5 the stack pointer is decremented.

Note 2: A jump to the appropriate address occurs here in mode 1 and mode 2. The CPU continues gathering data from the interrupting peripheral in mode 0 for a total of 2–4 machine cycles. In mode 0 cycles M2–M4 have only 1 wait state.

FIGURE 18. Interrupt Acknowledge Machine Cycle

9.0 Timing and Control (Continued)

so that the complete state of the CPU just prior to the non-maskable interrupt may be restored. The method of restoring the status of IFF_1 is through the execution of a Return Non-Maskable Interrupt (RETN) instruction. Since this instruction indicates that the non-maskable interrupt service routine is completed, the contents of IFF_2 are now copied back into IFF_1, so that the status of IFF_1 just prior to the acceptance of the non-maskable interrupt will be automatically restored.

Figure 19 depicts the status of the flip flops during a sample series of interrupt instructions.

Interrupt Control Register. The interrupt control register (ICR) is a 4-bit, write only register that provides the programmer with a second level of maskable control over the four maskable interrupt inputs.

The ICR is internal to the NSC800 CPU, but is addressed through the I/O space at I/O address port X'BB. Each bit in the register controls a mask bit dedicated to each maskable interrupt, $\overline{\text{RSTA}}$, $\overline{\text{RSTB}}$, $\overline{\text{RSTC}}$ and $\overline{\text{INTR}}$. For an interrupt request to be accepted on any of these inputs, the corresponding mask bit in the ICR must be set (= 1) and IFF_1 and IFF_2 must be set. This provides the programmer with control over individual interrupt inputs rather than just a system wide enable or disable.

3	2	1	0
IEA	IEB	IEC	IEI

TL/C/5171-26

Bit	Name	Function
0	IEI	Interrupt Enable for $\overline{\text{INTR}}$
1	IEC	Interrupt Enable for $\overline{\text{RSTC}}$
2	IEB	Interrupt Enable for $\overline{\text{RSTB}}$
3	IEA	Interrupt Enable for $\overline{\text{RSTA}}$

For example: In order to enable $\overline{\text{RSTB}}$, CPU interrupts must be enabled and IEB must be set.

At reset, IEI bit is set and other mask bits IEA, IEB, IEC are cleared. This maintains the software compatibility between NSC800 and Z80A.

Execution of an I/O block move instruction will not affect the state of the interrupt control bits. The only two instructions that will modify this write only register are OUT (C), r and OUT (N), A.

Operation	IFF_1	IFF_2	Comment
Initialize	0	0	Interrupt Disabled
•			
•			
•			
EI	1	1	Interrupt Enabled after next instruction
•			
•			
•			
$\overline{\text{INTR}}$	0	0	Interrupt Disable and $\overline{\text{INTR}}$ Being Serviced
•			
•			
•			
EI	1	1	Interrupt Enabled after next instruction
RET	1	1	Interrupt Enabled
•			
•			
•			
$\overline{\text{NMI}}$	0	1	Interrupt Disabled
•			
•			
•			
RETN	1	1	Interrupt Enabled
•			
$\overline{\text{INTR}}$	0	0	Interrupt Disabled
•			
•			
•			
$\overline{\text{NMI}}$	0	0	Interrupt Disabled and $\overline{\text{NMI}}$ Being Serviced
•			
•			
•			
RETN	0	0	Interrupt Disabled and $\overline{\text{INTR}}$ Being Serviced
•			
•			
•			
EI	1	1	Interrupt Enabled after next instruction
RET	1	1	Interrupt Enabled
•			
•			
•			

FIGURE 19. IFF_1 and IFF_2 States Immediately after the Operation has been Completed

NSC800 SOFTWARE

10.0 Introduction

This chapter provides the reader with a detailed description of the NSC800 software. Each NSC800 instruction is described in terms of opcode, function, flags affected, timing, and addressing mode.

11.0 Addressing Modes

The following sections describe the addressing modes supported by the NSC800. Note that particular addressing modes are often restricted to certain types of instructions. Examples of instructions used in the particular addressing modes follow each mode description.

The 10 addressing modes and 158 instructions provide a flexible and powerful instruction set.

11.1 REGISTER

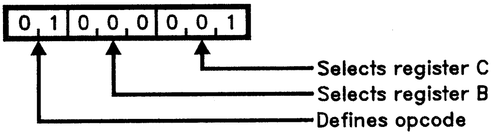
The most basic addressing mode is that which addresses data in the various CPU registers. In these cases, bits in the opcode select specific registers that are to be addressed by the instruction.

Example:

Instruction: Load register B from register C

Mnemonic: LD B,C

Opcode:



TL/C/5171-50

In this instruction, both the B and C registers are addressed by opcode bits.

11.2 IMPLIED

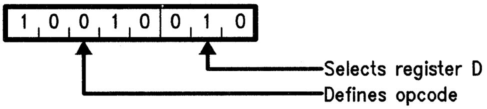
The implied addressing mode is an extension to the register addressing mode. In this mode, a specific register, the accumulator, is used in the execution of the instruction. In particular, arithmetic operations employ implied addressing, since the A register is assumed to be the destination register for the result without being specifically referenced in the opcode.

Example:

Instruction: Subtract the contents of register D from the Accumulator (A register)

Mnemonic: SUB D

Opcode:



TL/C/5171-51

In this instruction, the D register is addressed with register addressing, while the use of the A register is implied by the opcode.

11.3 IMMEDIATE

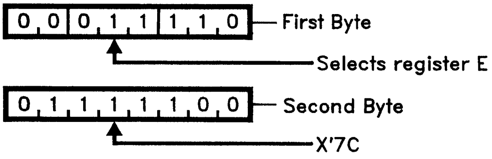
The most straightforward way of introducing data to the CPU registers is via immediate addressing, where the data is contained in an additional byte of multi-byte instructions.

Example:

Instruction: Load the E register with the constant value X'7C.

Mnemonic: LD E,X'7C

Opcode:



TL/C/5171-52

In this instruction, the E register is addressed with register addressing, while the constant X'7C is immediate data in the second byte of the instruction.

11.4 IMMEDIATE EXTENDED

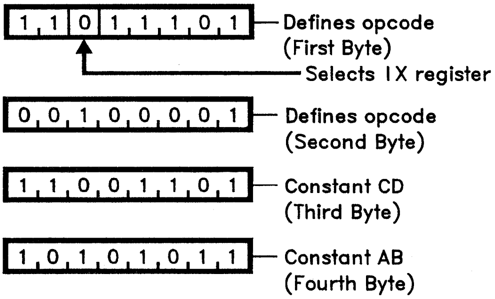
As immediate addressing allows 8 bits of data to be supplied by the operand, immediate extended addressing allows 16 bits of data to be supplied by the operand. These are in two additional bytes of the instruction.

Example:

Instruction: Load the 16-bit IX register with the constant value X'ABCD.

Mnemonic: LD IX,X'ABCD

Opcode:



TL/C/5171-53

In this instruction, register addressing selects the IX register, while the 16-bit quanity X'ABCD is immediate data supplied as immediate extended format.

11.0 Addressing Modes (Continued)

11.5 DIRECT ADDRESSING

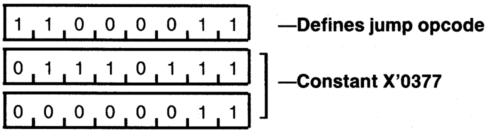
Direct addressing is the most straightforward way of addressing supplies a location in the memory space. Direct addressing, 16-bits of memory address information in two bytes of data as part of the instruction. The memory address could be either data, source of destination, or a location for program execution, as in program control instructions.

Example:

Instruction: Jump to location X'0377

Mnemonic: JP X'0377

Opcode:



This instruction loads the Program Counter (PC) is loaded with the constant in the second and third bytes of the instruction. The program counter contents are transferred via direct addressing.

11.6 REGISTER INDIRECT

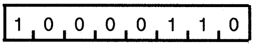
Next to direct addressing, register indirect addressing provides the second most straightforward means of addressing memory. In register indirect addressing, a specified register pair contains the address of the desired memory location. The instruction references the register pair and the register contents define the memory location of the operand.

Example:

Instruction: Add the contents of memory location X'0254 to the A register. The HL register contains X'0254.

Mnemonic: ADD A,(HL)

Opcode



This instruction uses implied addressing of the A and HL registers and register indirect addressing to access the data pointed to by the HL register.

11.7 INDEXED

The most flexible mode of memory addressing is the indexed mode. This is similar to the register indirect mode of addressing because one of the two index registers (IX or IY) contains the base memory address. In addition, a byte of data included in the instruction acts as a displacement to the address in the index register.

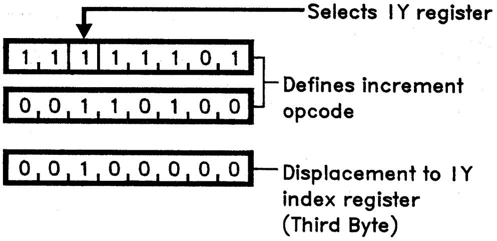
Indexed addressing is particularly useful in dealing with lists of data.

Example:

Instruction: Increment the data in memory location X'1020. The IY register contains X'1000.

Mnemonic: INC (IY+X'20)

Opcode:



TL/C/5171-54

The indexed addressing mode uses the contents of index registers IX or IY along with the displacement to form a pointer to memory.

11.8 RELATIVE

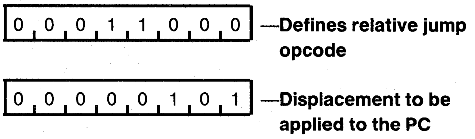
Certain instructions allow memory locations to be addressed as a position relative to the PC register. These instructions allow jumps to memory locations which are offsets around the program counter. The offset, together with the current program location, is determined through a displacement byte included in the instruction. The formation of this displacement byte is explained more fully in the "Instructions Set" section.

Example:

Instruction: Jump to a memory location 7 bytes beyond the current location.

Mnemonic: JR $+7

Opcode:


0 0 0 1 1 0 0 0
—Defines relative jump opcode
0 0 0 0 0 1 0 1
—Displacement to be applied to the PC

The program will continue at a location seven locations past the current PC.

11.0 Addressing Modes (Continued)

11.9 MODIFIED PAGE ZERO

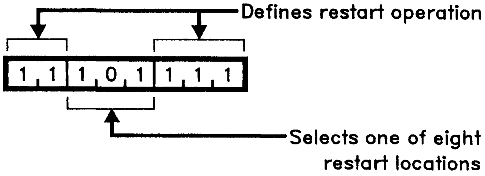
A subset of NSC800 instructions (the Restart instructions) provides a code-efficient single-byte instruction that allows CALLs to be performed to any one of eight dedicated locations in page zero (locations X'0000 to X'00FF). Normally, a CALL is a 3-byte instruction employing direct memory addressing.

Example:

Instruction: Perform a restart call to location X'0028.

Mnemonic: RST X'28

Opcode:


Defines restart operation
1 1 1 0 1 1 1 1
Selects one of eight restart locations

TL/C/5171-55

p	00H	08H	10H	18H	20H	28H	30H	38H
t	000	001	010	011	100	101	110	111

Program execution continues at location X'0028 after execution of a single-byte call employing modified page zero addressing.

11.10 BIT

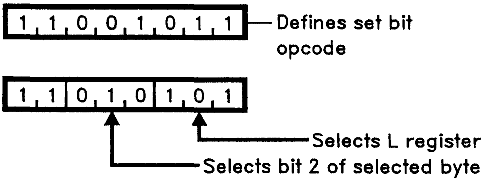
The NSC800 allows setting, resetting, and testing of individual bits in registers and memory data bytes.

Example:

Operation: Set bit 2 in the L register

Mnemonic: SET 2,L

Opcode:



TL/C/5171-56

Bit addressing allows the selection of bit 2 in the L register selected by register addressing.

12.0 Instruction Set

This section details the entire NSC800 instruction set in terms of

- Opcode
- Instruction
- Function
- Timing
- Addressing Mode

The instructions are grouped in order under the following functional headings:

- 8-Bit Loads
- 16-Bit Loads
- 8-Bit Arithmetic
- 16-Bit Arithmetic
- Bit Set, Reset, and Test
- Rotate and Shift
- Exchanges
- Memory Block Moves and Searches
- Input/Output
- CPU Control
- Program Control

12.1 Instruction Set Index

Alphabetical Assembly Mnemonic	Operation	Page
ADC A,m_1	Add, with carry, memory location contents to Accumulator	1-42
ADC A,n	Add, with carry, immediate data n to Accumulator	1-40
ADC A,r	Add, with carry, register r contents to Accumulator	1-38
ADC HL,pp	Add, with carry, register pair pp to HL	1-45
ADD A,m_1	Add memory location contents to Accumulator	1-42
ADD A,n	Add immediate data n to Accumulator	1-40
ADD A,r	Add register r contents to Accumulator	1-38
ADD HL,pp	Add register pair pp to HL	1-45
ADD IX,pp	Add register pair pp to IX	1-45
ADD IY,pp	Add register pair pp to IY	1-45
ADD ss,pp	Add register pair pp to contents of register pair ss	1-45
AND m_1	Logical 'AND' memory contents to Accumulator	1-43
AND n	Logical 'AND' immediate data to Accumulator	1-41
AND r	Logical 'AND' register r contents to Accumulator	1-38
BIT b,m_1	Test bit b of location m_1	1-47
BIT b,r	Test bit b of register r	1-46
CALL cc,nn	Call subroutine at location nn if condition cc is true	1-58
CALL nn	Unconditional call to subroutine at location nn	1-58
CCF	Complement carry flag	1-40
CP m_1	Compare memory contents with Accumulator	1-44
CP n	Compare immediate data n with Accumulator	1-42
CP r	Compare register r to contents with Accumulator	1-39
CPD	Compare location (HL) and Accumulator, decrement HL and BC	1-52
CPDR	Compare location (HL) and Accumulator, decrement HL and BC; repeat until BC = 0	1-53
CPI	Compare location (HL) and Accumulator, increment HL, decrement BC	1-52
CPIR	Compare location (HL) and Accumulator, increment HL, decrement BC; repeat until BC = 0	1-53
CPL	Complement Accumulator (1's complement)	1-39
DAA	Decimai adjust Accumulator	1-40
DEC m_1	Decrement data in memory location m_1	1-44
DEC r	Decrement register r contents	1-39
DEC rr	Decrement register pair rr contents	1-46

12.1 Instruction Set Index (Continued)

Alphabetical Assembly Mnemonic	Operation	Page
DI	Disable interrupts	1-56
DJNZ,d	Decrement B and jump relative B ≠ 0	1-58
EI	Enable interrupts	1-56
EX (SP),ss	Exchange the location (SP) with register ss	1-52
EX AF,A'F'	Exchange the contents of AF and A'F'	1-51
EX DE,HL	Exchange the contents of DE and HL	1-51
EXX	Exchange the contents of BC, DE and HL with the contents of B'C, D'E' and H'L', respectively	1-52
HALT	Halt (wait for interrupt or reset)	1-56
IM 0	Set interrupt mode 0	1-56
IM 1	Set interrupt mode 1	1-57
IM 2	Set interrupt mode 2	1-57
IN A,(n)	Load Accumulator with input from device (n)	1-54
IN r,(C)	Load register r with input from device (C)	1-54
INC m_1	Increment data in memory location m_1	1-44
INC r	Increment register r	1-39
INC rr	Increment contents of register pair rr	1-45
IND	Load location (HL) with input from port (C), decrement HL and B	1-54
INDR	Load location (HL) with input from port (C), decrement HL and B; repeat until B = 0	1-56
INI	Load location (HL) with input from port (C), increment HL, decrement B	1-54
INIR	Load location (HL) with input from port (C), increment HL, decrement B; repeat until B = 0	1-55
JP cc,nn	Jump to location nn, if condition cc is true	1-57
JP nn	Unconditional jump to location nn	1-57
JP (ss)	Unconditional jump to location (ss)	1-57
JR d	Unconditional jump relative to PC + d	1-57
JR kk,d	Jump relative to PC + d, if kk true	1-57
LD A,I	Load Accumulator with register I contents	1-34
LD A,m_2	Load Accumulator from location m_2	1-35
LD A,R	Load Accumulator with register R contents	1-34
LD I,A	Load register I with Accumulator contents	1-34
LD m_1,n	Load memory with immediate data n	1-35
LD m_1,r	Load memory from register r	1-34
LD m_2,A	Load memory from Accumulator	1-35
LD (nn),rr	Load memory location nn with register pair rr	1-36
LD r,m_1	Load register r from memory	1-35
LD r,n	Load register with immediate data n	1-34
LD R,A	Load register R from Accumulator	1-34
LD r_d,r_s	Load destination register r_d from source register r_s	1-34
LD rr,(nn)	Load register pair rr from memory location nn	1-37
LD rr,nn	Load register pair rr with immediate data nn	1-36
LD SP,ss	Load SP from register pair ss	1-36
LDD	Load location (DE) with location (HL), decrement DE, HL and BC	1-52
LDDR	Load location (DE) with location (HL), decrement DE, HL and BC; repeat until BC = 0	1-53
LDI	Load location (DE) with location (HL), increment DE and HL, decrement BC	1-52
LDIR	Load location (DE) with location (HL), increment DE and HL, decrement BC; repeat until BC = 0	1-53
NEG	Negate Accumulator (2's complement)	1-40
NOP	No operation	1-56

12.1 Instruction Set Index (Continued)

Alphabetical Assembly Mnemonic	Operation	Page
OR m_1	Logical 'OR' of memory location contents and accumulator	1-43
OR n	Logical 'OR' of immediate data n and Accumulator	1-41
OR r	Logical 'OR' of register r and Accumulator	1-39
OTDR	Load output port (C) with location (HL), decrement HL and B; repeat until B = 0	1-56
OTIR	Load output port (C) with location (HL), increment HL, decrement B; repeat until B = 0	1-55
OUT (C),r	Load output port (C) with register r	1-54
OUT (n),A	Load output port (n) with Accumulator	1-55
OUTD	Load output port (C) with location (HL), decrement HL and B	1-55
OUTI	Load output port (C) with location (HL), increment HL, decrement B	1-54
POP qq	Load register pair qq with top of stack	1-37
PUSH qq	Load top of stack with register pair qq	1-37
RES b,m_1	Reset bit b of memory location m_1	1-46
RES b,r	Reset bit b of register r	1-46
RET	Unconditional return from subroutine	1-58
RET cc	Return from subroutine, if cc true	1-58
RETI	Unconditional return from interrupt	1-58
RETN	Unconditional return from non-maskable interrupt	1-59
RL m_1	Rotate memory contents left through carry	1-49
RL r	Rotate register r left through carry	1-47
RLA	Rotate Accumulator left through carry	1-47
RLC m_1	Rotate memory contents left circular	1-49
RLC r	Rotate register r left circular	1-47
RLCA	Rotate Accumulator left circular	1-47
RLD	Rotate digit left and right between Accumulator and memory (HL)	1-51
RR m_1	Rotate memory contents right through carry	1-50
RR r	Rotate register r right through carry	1-48
RRA	Rotate Accumulator right through carry	1-50
RRC m_1	Rotate memory contents right circular	1-49
RRC r	Rotate register r right circular	1-47
RRCA	Rotate Accumulator right circular	1-48
RRD	Rotate digit right and left between Accumulator and memory (HL)	1-51
RST P	Restart to location P	1-59
SBC A,m_1	Subtract, with carry, memory contents from Accumulator	1-43
SBC A,n	Subtract, with carry, immediate data n from Accumulator	1-41
SBC A,r	Subtract, with carry, register r from Accumulator	1-38
SBC HL,pp	Subtract, with carry, register pair pp from HL	1-45
SCF	Set carry flag	1-40
SET b,m_1	Set bit b in memory location m_1 contents	1-46
SET b,r	Set bit b in register r	1-46
SLA m_1	Shift memory contents left, arithmetic	1-50
SLA r	Shift register r left, arithmetic	1-48
SRA m_1	Shift memory contents right, arithmetic	1-50
SRA r	Shift register r right, arithmetic	1-48
SRL m_1	Shift memory contents right, logical	1-50
SRL r	Shift register r right, logical	1-48
SUB m_1	Subtract memory contents from Accumulator	1-42
SUB n	Subtract immediate data n from Accumulator	1-41
SUB r	Subtract register r from Accumulator	1-38
XOR m_1	Exclusive 'OR' memory contents and Accumulator	1-44
XOR n	Exclusive 'OR' immediate data n and Accumulator	1-41
XOR r	Exclusive 'OR' register r and Accumulator	1-39

12.0 Instruction Set (Continued)

12.2 INSTRUCTION SET MNEMONIC NOTATION

In the following instruction set listing, the notations used are shown below.

b: Designates one bit in a register or memory location. Bit address mode uses this indicator.

cc: Designates condition codes used in conditional Jumps, Calls, and Return instruction; may be:

NZ = Non-Zero (Z flag=0)
Z = Zero (Z flag=1)
NC = Non-Carry (C flag=0)
C = Carry (C flag=1)
PO = Parity Odd or No Overflow (P/V=0)
PE = Parity Even or Overflow (P/V=1)
P = Positive (S=0)
M = Negative (S=1)

d: Designates an 8-bit signed complement displacement. Relative or indexed address modes use this indicator.

kk: Subset of cc condition codes used in conjunction with conditional relative jumps; may be NZ, Z, NC or C.

m_1: Designates (HL), (IX+d) or (IY+d). Register indirect or indexed address modes use this indicator.

m_2: Designates (BC), (DE) or (nn). Register indirect or direct address modes use this indicator.

n: Any 8-bit binary number.

nn: Any 16-bit binary number.

p: Designates restart vectors and may be the hex values 0, 8, 10, 18, 20, 28, 30 or 38. Restart instructions employing the modified page zero addressing mode use this indicator.

pp: Designates the BC, DE, SP or any 16-bit register used as a destination operand in 16-bit arithmetic operations employing the register address mode.

qq: Designates BC, DE, HL, A, F, IX, or IY during operations employing register address mode.

r: Designates A, B, C, D, E, H or L. Register addressing modes use this indicator.

rr: Designates BC, DE, HL, SP, IX or IY. Register addressing modes use this indicator.

ss: Designates HL, IX or IY. Register addressing modes use this indicator.

X_L: Subscript L indicates the lower-order byte of a 16-bit register.

X_H: Subscript H indicates the high-order byte of a 16-bit register.

( ): parentheses indicate the contents are considered a pointer address to a memory or I/O location.

12.3 ASSEMBLED OBJECT CODE NOTATION

Register Codes:

r	Register	rp	Register	rs	Register
000	B	00	BC	00	BC
001	C	01	DE	01	DE
010	D	10	HL	10	HL
011	E	11	SP	11	AF
		pp	**Register**	**qq**	**Register**
100	H	00	BC	00	BC
101	L	01	DE	01	DE
111	A	10	IX	10	HL
		11	SP	11	AF

Conditions Codes:

cc	Mnemonic	True Flag Condition
000	NZ	Z=0
001	Z	Z=1
010	NC	C=0
011	C	C=1
100	PO	P/V=0
101	PE	P/V=1
110	P	S=0
111	M	S=1
kk	**Mnemonic**	**True Flag Condition**
00	NZ	Z=0
01	Z	Z=1
10	NC	C=0
11	C	C=1

Restart Addresses:

t	T
000	X'00
001	X'08
010	X'10
011	X'18
100	X'20
101	X'28
110	X'30
111	X'38

12.4 8-Bit Loads

REGISTER TO REGISTER

LD r_d, r_s

Load register r_d with r_s:

$r_d \leftarrow r_s$ No flags affected

7	6	5	4	3	2	1	0
0	1	r_d			r_s		

Timing: M cycles — 1
T states — 4

Addressing Mode: Register

LD A, I

Load Accumulator with the contents of the I register.

A ← I

S: Set if negative result
Z: Set if zero result
H: Reset
P/V: Set according to IFF_2 (zero if interrupt occurs during operation)
N: Reset
C: Not affected

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
0	1	0	1	0	1	1	1

Timing: M cycles — 2
T states — 9 (4, 5)

Addressing Mode: Register

LD I, A

Load Interrupt vector register (I) with the contents of A.

I ← A No flags affected

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
0	1	0	0	0	1	1	1

Timing: M cycles — 2
T states — 9 (4, 5)

Addressing Mode: Register

LD A, R

Load Accumulator with contents of R register.

A ← R

S: Set if negative result
Z: Set if zero result
H: Reset
P/V: Set according to IFF_2 (zero if interrupt occurs during operation)
N: Reset
C: Not affected

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
0	1	0	1	1	1	1	1

Timing: M cycles — 2
T states — 9 (4, 5)

Addressing Mode: Register

LD R, A

Load Refresh register (R) with contents of the Accumulator.

R ← A No flags affected

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
0	1	0	0	1	1	1	1

Timing: M cycles — 2
T states — 9 (4, 5)

Addressing Mode: Register

LD r, n

Load register r with immediate data n.

r ← n No flags affected

7	6	5	4	3	2	1	0
0	0	r			1	1	0
n							

Timing: M cycles — 2
T states — 7 (4, 3)

Addressing Mode: Source — Immediate
Destination — Register

REGISTER TO MEMORY

LD m_1, r

Load memory from reigster r.

$m_1 \leftarrow r$ No flags affected

7	6	5	4	3	2	1	0	
0	1	1	1	0	r			LD (HL), r

Timing: M cycles — 2
T states — 7 (4,3)

Addressing Mode: Source — Register
Destination — Register Indirect

7	6	5	4	3	2	1	0	
1	1	N_X	1	1	1	0	1	LD (IX+d), r(for $N_X=0$) LD (IY+d), r(for $N_X=1$)
0	1	1	1	0	r			
d								

Timing: M cycles — 2
T states — 19 (4, 4, 3, 5, 3)

Addressing Mode: Source — Register
Destination — Indexed

12.4 8-Bit Loads (Continued)

LD m$_2$, A

Load memory from the Accumulator.

m$_2$ ← A — No flags affected

7	6	5	4	3	2	1	0	
0	0	0	0	0	0	1	0	LD (BC), A
0	0	0	1	0	0	1	0	LD (DE), A

Timing: M cycles — 2
T states — 7 (4, 3)

Addressing Mode: Source — Register (Implied)
Destination — Register Indirect

7	6	5	4	3	2	1	0	
0	0	2	2	0	0	1	0	LD (nn), A
n (low-order byte)								
n (high-order byte)								

Timing: M cycles — 4
T states — 3 (4, 3, 3, 3)

Addressing Mode: Source — Register (Implied)
Destination — Direct

LD m$_1$, n

Load memory with immediate data.

m$_1$ ← n — No flags affected

7	6	5	4	3	2	1	0	
0	0	1	1	0	1	1	0	LD(HL), n
n								

Timing: M cycles—3
T states—10 (4, 3, 3)

Addressing Mode: Source—Immediate
Destination—Register Indirect

7	6	5	4	3	2	1	0	
1	1	N$_X$	1	1	1	0	1	LD (IX + d), n(for N$_X$ = 0) LD (IY + d), n(for N$_X$ = 1)
0	0	1	1	0	1	1	0	
d								
n								

Timing: M cycles—5
T states—19 (4, 4, 3, 5, 3)

Addressing Mode: Source—Immediate
Destination—Indexed

MEMORY TO REGISTER

LD r, m$_1$

Load register r from memory location m$_1$.

r ← m$_1$ — No flags affected

7	6	5	4	3	2	1	0	
0	1		r		1	1	0	LD R, (HL)

Timing: M cycles—2
T states—7 (4, 3)

Addressing Mode: Source—Register Indirect
Destination—Register

7	6	5	4	3	2	1	0	
1	1	N$_X$	1	1	1	0	1	LD r, (IX + d) (for N$_X$=0) LD r, (IY + d) (for N$_X$=1)
0	1		r		1	1	0	
d								

Timing: M cycles—5
T states—19 (4, 4, 3, 5, 3)

Addressing Mode: Source—Indexed
Destination—Register

LD A, m$_2$

Load the Accumulator from memory location m$_2$.

A ← m$_2$ — No flags affected

7	6	5	4	3	2	1	0	
0	0	0	0	1	0	1	0	LD A, (BC)
0	0	0	1	1	0	1	0	LD A, (DE)

Timing: M cycles—2
T states—7 (4, 3)

Addressing Mode: Source—Register Indirect
Destination—Register (Implied)

7	6	5	4	3	2	1	0	
0	0	1	1	1	0	1	0	LD A, (nn)
n (low-order byte)								
n (high-order byte)								

Timing: M cycles—4
T states—13 (4, 3, 3, 3)

Addressing Mode: Source—Immediate Extended
Destination—Register (Implied)

12.5 16-Bit Loads

REGISTER TO REGISTER

LD rr, nn

Load 16-bit register pair with immediate data.

rr, ← nn No flags affected

7	6	5	4	3	2	1	0
0	0	rp		0	0	0	1
n (low-order byte)							
n (high-order byte)							

LD BC, nn
LD DE, nn
LD HL, nn
LD SP, nn

Timing: M cycles—3
T states—10 (4, 3, 3)

Addressing Mode: Source—Immediate Extended
Destination—Register

7	6	5	4	3	2	1	0
1	1	N_X	1	1	1	0	1
0	0	1	0	0	0	0	1
n (low-order byte)							
n (high-order byte)							

LD IX, nn (for $N_X = 0$)

LD IY, nn (for $N_X = 1$)

Timing: M cycles—4
T states—14 (4, 4, 3, 3)

Addressing Mode: Source—Immediate Extended
Destination—Register

LD SP, ss

Load the SP from 16-bit register ss.

SP ← ss No flags affected

7	6	5	4	3	2	1	0
1	1	1	1	1	0	0	1

LD SP, HL

Timing: M cycles—1
T states—6

Addressing Mode: Source—Register
Destination—Register (Implied)

7	6	5	4	3	2	1	0
1	1	N_X	1	1	1	0	1
1	1	1	1	1	0	0	1

LD SP, IX (for $N_X = 0$)

LD SP, IY (for $N_X = 1$)

Timing: M cycles—2
T states—10 (4, 6)

Addressing Mode: Source—Register
Destination—Register (Implied)

REGISTER TO MEMORY

LD (nn), rr

Load memory location nn with contents of 16-bit register, rr.

(nn) ← rr_L No flags affected

(nn + 1) ← rr_H

7	6	5	4	3	2	1	0
0	0	1	0	0	0	1	0
n (low-order byte)							
n (high-order byte)							

LD (nn), HL
(note an alternate
opcode below)

Timing: M cycles—5
T states—16 (4, 3, 3, 3, 3)

Addressing Mode: Source—Register
Destination—Direct

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
0	1	rp		0	0	1	1
n (low-order byte)							
n (high-order byte)							

LD (nn), BC
LD (nn), DE
LD (nn), HL
LD (nn), SP

Timing: M cycles—6
T states—20 (4, 4, 3, 3, 3, 3)

Addressing Mode: Source—Register
Destination—Direct

7	6	5	4	3	2	1	0
1	1	N_X	1	1	1	0	1
0	0	1	0	0	0	1	0
n (low-order byte)							
n (high-order byte)							

LD (nn), IX (for $N_X = 0$)

LD (nn) IY (for $N_X = 1$)

Timing: M cycles—6
T states—20 (4, 4, 3, 3, 3, 3)

Addressing Mode: Source—Register
Destination—Direct

12.5 16-Bit Loads (Continued)

PUSH qq

Push the contents of register pair qq onto the memory stack.

$(SP - 1) \leftarrow qq_H$ No flags affected

$(SP - 2) \leftarrow qq_L$

$SP \leftarrow SP - 2$

7	6	5	4	3	2	1	0
1	1	rs		0	1	0	1

PUSH BC
PUSH DE
PUSH HL
PUSH AF

Timing: M cycles—3
T states—11 (5, 3, 3)

Addressing Mode: Source—Register
Destination—Register Indirect (Stack)

7	6	5	4	3	2	1	0
1	1	N_X	1	1	1	0	1
1	1	1	0	0	1	0	1

PUSH IX (for $N_X = 0$)
PUSH IY (for $N_X = 1$)

Timing: M cycles—3
T states—15 (4, 5, 3, 3)

Addressing Mode: Source—Register
Destination—Register Indirect (Stack)

MEMORY TO REGISTER

LD rr, (nn)

Load 16-bit register from memory location nn.

$rr_L \leftarrow (nn)$ No flags affected

$rr_H \leftarrow (nn + 1)$

7	6	5	4	3	2	1	0
0	0	1	0	0	0	1	0

LD HL, (nn)
(note an alternate opcode below)

n (low-order byte)

n (high-order byte)

Timing: M cycles—5
T states—16 (4, 3, 3, 3, 3)

Addressing Mode: Source—Direct
Destination—Register

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
0	1	rp		0	0	1	1

LD BC, (nn)
LD DE, (nn)
LD HL, (nn)
LD SP, (nn)

n (low-order byte)

n (high-order byte)

Timing: M cycles—6
T states—20 (4, 4, 3, 3, 3, 3)

Addressing Mode: Source—Direct
Destination—Register

7	6	5	4	3	2	1	0
1	1	N_X	1	1	1	0	1
0	0	1	0	1	0	1	0

LD IX, (nn)(for $N_X = 0$)
LD IY, (nn) (for $N_X = 1$)

n (low-order byte)

n (high-order byte)

Timing: M cycles—6
T states—20 (4, 4, 3, 3, 3, 3)

Addressing Mode: Source—Direct
Destination—Register

POP qq

Pop the contents of the memory stack to register qq.

$qq_L \leftarrow (SP)$ No flags affected

$qq_H \leftarrow (SP + 1)$

$SP \leftarrow SP + 2$

7	6	5	4	3	2	1	0
1	1	rs		0	0	0	1

POP BC
POP DE
POP HL
POP AF

Timing: M cycles—3
T states—10 (4, 3, 3)

Addressing Mode: Source—Register Indirect (Stack)
Destination—Register

7	6	5	4	3	2	1	0
1	1	N_X	1	1	1	0	1
1	1	1	0	0	0	0	1

POP IX (for $N_X = 0$)
POP IY (for $N_X = 1$)

Timing: M cycles—4
T states—14 (4, 4, 3, 3)

Addressing Mode: Source—Register Indirect (Stack)
Destination—Register

12.6 8-Bit Arithmetic

REGISTER ADDRESSING ARITHMETIC

Op	C Before DAA	Hex Value In Upper Digit (Bits 7-4)	H Before DAA	Hex Value In Lower Digit (Bits 3-0)	Number Added To Byte	C After DAA
	0	0-9	0	0-9	00	0
	0	0-8	0	A-F	06	0
	0	0-9	1	0-3	06	0
ADD	0	A-F	0	0-9	60	1
ADC	0	9-F	0	A-F	66	1
INC	0	A-F	1	0-3	66	1
	1	0-2	0	0-9	60	1
	1	0-2	0	A-F	66	1
	1	0-3	1	0-3	66	1
SUB	0	0-9	0	0-9	00	0
SBC	0	0-8	1	6-F	FA	0
DEC	1	7-F	0	0-9	A0	1
NEG	1	6-F	1	6-F	9A	1

ADD A, r

Add contents of register r to the Accumulator.

A ← A+r

S: Set if negative result
Z: Set if zero result
H: Set if carry from bit 3
P/V: Set according to overflow condition
N: Reset
C: Set if carry from bit 7

7	6	5	4	3	2	1	0
1	0	0	0	0	r		

Timing: M cycles—1
T states—4

Addressing Mode: Source—Register
Destination—Implied

ADC A, r

Add contents of register r, plus the carry flag, to the Accumulator.

A ← A + r + CY

S: Set if negative result
Z: Set if zero result
H: Set if carry from bit 3
P/V: Set if result exceeds 2's complement range
N: Reset
C: Set if carry from bit 7

7	6	5	4	3	2	1	0
1	0	0	0	1	r		

Timing: M cycles—1
T states—4

Addressing Mode: Source—Register
Destination—Implied

SUB r

Subtract the contents of register r from the Accumulator.

A ← A − r

S: Set if result is negative
Z: Set if result is zero
H: Set if borrow from bit 4
P/V: Set if result exceeds 8-bit 2's complement range
N: Set
C: Set according to borrow

7	6	5	4	3	2	1	0
1	0	0	1	0	r		

Timing: M cycles—1
T states—4

Addressing Mode: Source—Register
Destination—Implied

SBC A, r

Subtract contents of register r and the carry bit C from the Accumulator.

A ← A − r − CY

S: Set if result is negative
Z: Set if result is zero
H: Set if borrow from bit 4
P/V: Set if result exceeds 8-bit 2's complement range
N: Set
C: Set according to borrow

7	6	5	4	3	2	1	0
1	0	0	1	1	r		

Timing: M cycles—1
T states—4

Addressing Mode: Source—Register
Destination—Implied

AND r

Logically AND the contents of the r register and the Accumulator.

A ← A ∧ r

S: Set if result is negative
Z: Set if result is zero
H: Set
P/V: Set if result parity is even
N: Reset
C: Reset

12.6 8-Bit Arithmetic (Continued)

7	6	5	4	3	2	1	0
1	0	1	0	0	r		

Timing: M cycles—1
T states—4
Addressing Mode: Source—Register
Destination—Implied

OR r

Logically OR the contents of the r register and the Accumulator.

A ← A ∨ r
S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: Reset

7	6	5	4	3	2	1	0
1	0	1	1	0	r		

Timing: M cycles—1
T states—4
Addressing Mode: Source—Register
Destination—Implied

XOR r

Logically exclusively OR the contents of the r register with the Accumulator.

A ← A ⊕ r
S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: Reset

7	6	5	4	3	2	1	0
1	0	1	0	1	r		

Timing: M cycles—1
T states—4
Addressing Mode: Source—Register
Destination—Implied

INC r

Increment register r.

r ← r + 1
S: Set if result is negative
Z: Set if result is zero
H: Set if carry from bit 3
P/V: Set only if r was X'7F before operation
N: Reset
C: N/A

7	6	5	4	3	2	1	0
0	0	r			1	0	0

Timing: M cycles—1
T states—4
Addressing Mode: Source—Register
Destination—Register

CP r

Compare the contents of register r with the Accumulator and set the flags accordingly.

A − r
S: Set if result is negative
Z: Set if result is zero
H: Set if borrow from bit 4
P/V: Set if result exceeds 8-bit 2's complement range
N: Set
C: Set according to borrow

7	6	5	4	3	2	1	0
1	0	1	1	1	r		

Timing: M cycles—1
T states—4
Addressing Mode: Source—Register
Destination—Implied

DEC r

Decrement the contents of register r.

r ← r − 1
S: Set if result is negative
Z: Set if result is zero
H: Set according to a borrow from bit 4
P/V: Set only if r was X'80 prior to operation
N: Set
C: N/A

7	6	5	4	3	2	1	0
0	0	r			1	0	1

Timing: M cycles—1
T states—4
Addressing Mode: Source—Register
Destination—Register

CPL

Complement the Accumulator (1's complement).

A ← $\overline{A}$
S: N/A
Z: N/A
H: Set
P/V: N/A
N: Set
C: N/A

12.6 8-Bit Arithmetic (Continued)

7	6	5	4	3	2	1	0
0	0	1	0	1	1	1	1

Timing:	M cycles—1
	T states—4
Addressing Mode:	Implied

NEG

Negate the Accumulator (2's complement).

A ← 0 − A
S: Set if result is negative
Z: Set if result is zero
H: Set according to borrow from bit 4
P/V: Set only if Accumulator was X'80 prior to operation
N: Set
C: Set only if Accumulator was not X'00 prior to operation

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
0	1	0	0	0	1	0	0

Timing:	M cycles—2
	T states—8 (4, 4)
Addressing Mode:	Implied

CCF

Complement the carry flag.

CY ← $\overline{CY}$
S: N/A
Z: N/A
H: Previous carry
P/V: N/A
N: Reset
C: Complement of previous carry

7	6	5	4	3	2	1	0
0	0	1	1	1	1	1	1

Timing:	M cycles—1
	T states—4
Addressing Mode:	Implied

SCF

Set the carry flag.

CY ← 1
S: N/A
Z: N/A
H: Reset
P/V: N/A
N: Reset
C: Set

7	6	5	4	3	2	1	0
0	0	1	1	0	1	1	1

Timing:	M cycles—1
	T states—4
Addressing Mode:	Implied

DAA

Adjust the Accumulator for BCD addition and subtraction operations. To be executed after BCD data has been operated upon the standard binary ADD, ADC, INC, SUB, SBC, DEC or NEG instructions (see "Register Addressing Arithmetic" table).

———
S: Set according to bit 7 of result
Z: Set if result is zero
H: Set according to instructions
P/V: Set according to parity of result
N: N/A
C: Set according to instructions

7	6	5	4	3	2	1	0
0	0	1	0	0	1	1	1

Timing:	M cycles—1
	T states—4
Addressing Mode:	Implied

IMMEDIATELY ADDRESSED ARITHMETIC

ADD A, n

Add the immediate data n to the Accumulator.

A ← A + n
S: Set if result is negative
Z: Set if result is zero
H: Set if carry from bit 3
P/V: Set if result exceeds 8-bit 2's complement range
N: Reset
C: Set if carry from bit 7

7	6	5	4	3	2	1	0
1	1	0	0	0	1	1	0

n

Timing:	M cycles—2
	T states—7 (4, 3)
Addressing Mode:	Source—Immediate
	Destination—Implied

ADC A, n

Add, with carry, the immediate data n and the Accumulator.

A ← A + n + CY
S: Set if result is negative
Z: Set if result is zero
H: Set if carry from bit 3
P/V: Set if result exceeds 8-bit 2's complement range
N: Reset
C: Set according to carry from bit 7

12.6 8-Bit Arithmetic (Continued)

7	6	5	4	3	2	1	0
1	1	0	0	1	1	1	0

n

Timing: M cycles—2
T states—7 (4, 3)
Addressing Mode: Source—Immediate
Destination—Implied

SUB n

Subtract the immediate data n from the Accumulator.

A ← A − n
S: Set if result is negative
Z: Set if result is zero
H: Set if borrow from bit 4
P/V: Set if result exceeds 8-bit 2's complement range
N: Set
C: Set according to borrow condition

7	6	5	4	3	2	1	0
1	1	0	1	0	1	1	0

n

Timing: M cycles—2
T states—7 (4, 3)
Addressing Mode: Source—Immediate
Destination—Implied

SBC A, n

Subtract, with carry, the immediate data n from the Accumulator.

A ← A − n − CY
S: Set if result is negative
Z: Set if result is zero
H: Set if borrow from bit 4
P/V: Set if result exceeds 8-bit 2's complement range
N: Set
C: Set according to borrow condition

7	6	5	4	3	2	1	0
1	1	0	1	1	1	1	0

n

Timing: M cycles—2
T states—7 (4, 3)
Addressing Mode: Source—Immediate
Destination—Implied

AND n

The immediate data n is logically AND'ed to the Accumulator.

A ← A ∧ n
S: Set if result is negative
Z: Set if result is zero
H: Set
P/V: Set if result parity is even
N: Reset
C: Reset

7	6	5	4	3	2	1	0
1	1	1	0	0	1	1	0

n

Timing: M cycles—2
T states—7 (4, 3)
Addressing Mode: Source—Immediate
Destination—Implied

OR n

The immediate data n is logically OR'ed to the contents of the Accumulator.

A ← A ∨ s
S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: Reset

7	6	5	4	3	2	1	0
1	1	1	1	0	1	1	0

n

Timing: M cycles—2
T states—7 (4, 3)
Addressing Mode: Source—Immediate
Destination—Implied

XOR n

The immediate data n is exclusively OR'ed with the Accumulator.

A ← A ⊕ n
S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: Reset

12.6 8-Bit Arithmetic (Continued)

7	6	5	4	3	2	1	0
1	1	1	0	1	1	1	0
n							

Timing: M cycles—2
T states—7 (4, 3)

Addressing Mode: Source—Immediate
Destination—Implied

CP n

Compare the immediate data n with the contents of the Accumulator via subtraction and return the appropriate flags. The contents of the Accumulator are not affected.

A − n
S: Set if result is negative
Z: Set if result is zero
H: Set if borrow from bit 4
P/V: Set if result exceeds 8-bit 2's complement range
N: Set
C: Set according to borrow condition

7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	0
n							

Timing: M cycles—2
T states—7 (4, 3)

Addressing Mode: Immediate

MEMORY ADDRESSED ARITHMETIC

ADD A, m1

Add the contents of the memory location m_1 to the Accumulator.

A ← A + m_1
S: Set if result is negative
Z: Set if result is zero
H: Set if carry from bit 3
P/V: Set if result exceeds 8-bit 2's complement range
N: Reset
C: Set according to carry from bit 7

7	6	5	4	3	2	1	0	
1	0	0	0	0	1	1	0	ADD A, (HL)

Timing: M cycles—2
T states—7 (4, 3)

Addressing Mode: Source—Register Indirect
Destination—Implied

7	6	5	4	3	2	1	0	
1	1	N_X	1	1	1	0	1	ADD A, (IX + d) (for N_X=0) ADD A, (IY + d) (for N_X=1)
1	0	0	0	0	1	1	0	
d								

Timing: M cycles—5
T states—19 (4, 4, 3, 5, 3)

Addressing Mode: Source—Indexed
Destination—Implied

ADC A, m_1

Add the contents of the memory location m_1 plus the carry to the Accumulator.

A ← A + m_1 + CY
S: Set if result is negative
Z: Set if result is zero
H: Set if carry from bit 3
P/V: Set if result exceeds 8-bit 2's complement range
N: Reset
C: Set according to carry from bit 7

7	6	5	4	3	2	1	0	
1	0	0	0	1	1	1	0	ADC A, (HL)

Timing: M cycles—2
T states—7 (4, 3)

Addressing Mode: Source—Register Indirect
Destination—Implied

7	6	5	4	3	2	1	0	
1	1	N_X	1	1	1	0	1	ADC A, (IX + d) (for N_X=0) ADC A, (IY + d) (for N_X=1)
1	0	0	0	1	1	1	0	
d								

Timing: M cycles—5
T states—19 (4, 4, 3, 5, 3)

Addressing Mode: Source—Indexed
Destination—Implied

SUB m_1

Subtract the contents of memory location m_1 from the Accumulator.

A ← A − m_1
S: Set if result is negative
Z: Set if result is zero
H: Set if borrow from bit 4
P/V: Set if result exceeds 8-bit 2's complement range
N: Set
C: Set according to borrow condition

12.6 8-Bit Arithmetic (Continued)

7	6	5	4	3	2	1	0
1	0	0	1	0	1	1	0

SUB (HL)

Timing: M cycles—2
T states—7 (4, 3)

Addressing Mode: Source—Register Indirect
Destination—Implied

7	6	5	4	3	2	1	0
1	1	N_X	1	1	1	0	1
1	0	0	1	0	1	1	0
d							

SUB (IX + d) (for $N_X = 0$)
SUB (IY + d) (for $N_X = 1$)

Timing: M cycles—5
T states—19 (4, 4, 3, 5, 3)

Addressing Mode: Source—Indexed
Destination—Implied

SBC A, m_1

Subtract, with carry, the contents of memory location m_1 from the Accumulator.

$A \leftarrow A - m_1 - CY$

S: Set if result is negative
Z: Set if result is zero
H: Set if carry from bit 3
P/V: Set if result exceeds 8-bit 2's complement range
N: Set
C: Set according to borrow condition

7	6	5	4	3	2	1	0
1	0	0	1	1	1	1	0

SBC A, (HL)

Timing: M cycles—2
T states—7 (4, 3)

Addressing Mode: Source—Register Indirect
Destination—Implied

7	6	5	4	3	2	1	0
1	1	N_X	1	1	1	0	1
1	0	0	1	1	1	1	0
d							

SBC A, (IX + d) (for $N_X = 0$)
SBC A, (IY + d) (for $N_X = 1$)

Timing: M cycles—5
T states—19 (4, 4, 3, 5, 3)

Addressing Mode: Source—Indexed
Destination—Implied

AND m_1

The data in memory location m_1 is logically AND'ed to the Accumulator.

$A \leftarrow A \wedge m_1$

S: Set if result is negative
Z: Set if result is zero
H: Set
P/V: Set if result parity is even
N: Reset
C: Reset

7	6	5	4	3	2	1	0
1	0	1	0	0	1	1	0

AND (HL)

Timing: M cycles—2
T states—7 (4, 3)

Addressing Mode: Source—Register Indirect
Destination—Implied

7	6	5	4	3	2	1	0
1	1	N_X	1	1	1	0	1
1	0	1	0	0	1	1	0
d							

AND (IX + d) (for $N_X = 0$)
AND (IY + d) (for $N_X = 1$)

Timing: M cycles—5
T states—19 (4, 4, 3, 5, 3)

Addressing Mode: Source—Indexed
Destination—Implied

OR m_1

The data in memory location m_1 is logically OR'ed with the Accumulator.

$A \leftarrow A \vee m_1$

S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: Reset

7	6	5	4	3	2	1	0
1	0	1	1	0	1	1	0

OR (HL)

Timing: M cycles—2
T states—7 (4, 3)

Addressing Mode: Source—Register Indexed
Destination—Implied

7	6	5	4	3	2	1	0
1	1	N_X	1	1	1	0	1
1	0	1	1	0	1	1	0
d							

OR (IX + d) (for $N_X = 0$)
OR (IY + d) (for $N_X = 1$)

Timing: M cycles—5
T states—19 (4, 4, 3, 5, 3)

Addressing Mode: Source—Indexed
Destination—Implied

12.6 8-Bit Arithmetic (Continued)

XOR m_1

The data in memory location m_1 is exclusively OR'ed with the data in the Accumulator.

$A \leftarrow A \oplus m_1$

S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: Reset

7	6	5	4	3	2	1	0
1	0	1	0	1	1	1	0

XOR (HL)

Timing: M cycles—2
T states—7 (4, 3)

Addressing Mode: Source—Register Indexed
Destination—Implied

7	6	5	4	3	2	1	0
1	1	N_X	1	1	1	0	1
1	0	1	0	1	1	1	0
d							

XOR (IX + d) (for $N_X = 0$)
XOR (IY + d) (for $N_X = 1$)

Timing: M cycles—5
T states—19 (4, 4, 3, 5, 3)

Addressing Mode: Source—Indexed
Destination—Implied

CP m_1

Compare the data in memory location m_1 with the data in the Accumulator via subtraction.

$A - m_1$

S: Set if result is negative
Z: Set if result is zero
H: Set if borrow from bit 4
P/V: Set if result exceeds 8-bit 2's complement range
N: Set
C: Set according to borrow condition

7	6	5	4	3	2	1	0
1	0	1	1	1	1	1	0

CP (HL)

Timing: M cycles—2
T states—7 (4, 3)

Addressing Mode: Source—Register Indirect
Destination—Implied

7	6	5	4	3	2	1	0
1	1	N_X	1	1	1	0	1
1	0	1	1	1	1	1	0
d							

CP (IX + d) (for $N_X = 0$)
CP (IY + d) (for $N_X = 1$)

Timing: M cycles—5
T states—19 (4, 4, 3, 5, 3)

Addressing Mode: Source—Indexed
Destination—Implied

INC m_1

Increment data in memory location m_1.

$m_1 \leftarrow m_1 + 1$

S: Set if result is negative
Z: Set if result is zero
H: Set according to carry from bit 3
P/V: Set if data was X'7F before operation
N: Reset
C: N/A

7	6	5	4	3	2	1	0
0	0	1	1	0	1	0	0

INC (HL)

Timing: M cycles—3
T states—11 (4, 4, 3)

Addressing Mode: Source—Register Indexed
Destination—Register Indexed

7	6	5	4	3	2	1	0
1	1	N_X	1	1	1	0	1
0	0	1	1	0	1	0	0
d							

INC (IX + d) (for $N_X = 0$)
INC (IY + d) (for $N_X = 1$)

Timing: M cycles—6
T states—23 (4, 4, 3, 5, 4, 3)

Addressing Mode: Source—Indexed
Destination—Indexed

DEC m_1

Decrement data in memory location m_1.

$m_1 \leftarrow m_1 - 1$

S: Set if result is negative
Z: Set if result is zero
H: Set according to borrow from bit 4
P/V: Set only if m_1 was X'80 before operation
N: Set
C: N/A

12.6 8-Bit Arithmetic (Continued)

7	6	5	4	3	2	1	0	
0	0	1	1	0	1	0	1	DEC (HL)

Timing: M cycles — 3
T states — 11 (4, 4, 3)
Addressing Mode: Source — Register Indexed
Destination — Register Indexed

7	6	5	4	3	2	1	0	
1	1	N_X	1	1	1	0	1	DEC (IX + d) (for N_X = 0) DEC (IY + d) (for N_X = 1)
0	0	1	1	0	1	0	1	
d								

Timing: M cycles — 6
T states — 23 (4, 4, 3, 5, 4, 3)
Addressing Mode: Source — Indexed
Destination — Indexed

12.7 16-Bit Arithmetic

ADD ss, pp

Add the contents of the 16-bit register pp to the contents of the 16-bit register ss.

ss ← ss + pp
S: N/A
Z: N/A
H: Set if carry from bit 11
P/V: N/A
N: Reset
C: Set if carry from bit 15

7	6	5	4	3	2	1	0	
0	0	pp		1	0	0	1	ADD HL, pp

Timing: M cycles — 3
T states — 11 (4, 4, 3)
Addressing Mode: Source — Register
Destination — Register

7	6	5	4	3	2	1	0	
1	1	N_X	1	1	1	0	1	ADD IX, pp (for N_X = 0) ADD IY, pp (for N_X = 1)
0	0	pp		1	0	0	1	

Timing: M cycles — 4
T states — 15 (4, 4, 4, 3)
Addressing Mode: Source — Register
Destination — Register

ADC HL, pp

The contents of the 16-bit register pp are added, with the carry bit, to the HL register.

HL ← HL + pp + CY
S: Set if result is negative
Z: Set if result is zero
H: Set according to carry out of bit 11
P/V: Set if result exceeds 16-bit 2's complement range
N: Reset
C: Set if carry out of bit 15

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
0	1	pp		1	0	1	0

Timing: M cycles — 4
T states — 15 (4, 4, 4, 3)
Addressing Mode: Source — Register
Destination — Register

SBC HL, pp

Subtract, with carry, the contents of the 16-bit pp register from the 16-bit HL register.

HL ← HL − pp − CY
S: Set if result is negative
Z: Set if result is zero
H: Set according to borrow from bit 12
P/V: Set if result exceeds 16-bit 2's complement range
N: Set
C: Set according to borrow condition

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
0	1	pp		0	0	1	0

Timing: M cycles — 4
T states — 15 (4, 4, 4, 3)
Addressing Mode: Source — Register
Destination — Register

INC rr

Increment the contents of the 16-bit register rr.

rr ← rr + 1 No flags affected

7	6	5	4	3	2	1	0	
0	0	rp		0	0	1	1	INC BC INC DE INC HL INC SP

Timing: M cycles — 1
T states — 6
Addressing Mode: Register

7	6	5	4	3	2	1	0	
1	1	N_X	1	1	1	0	1	INC IX (for N_X=0) INC IY (for N_X=1)
0	0	1	0	0	0	1	1	

Timing: M cycles — 2
T states — 10 (4, 6)
Addressing Mode: Register

12.7 16-Bit Arithmetic (Continued)

DEC rr

Decrement the contents of the 16-bit register rr.

rr ← rr − 1 No flags affected

7	6	5	4	3	2	1	0
0	0	rp		1	0	1	1

DEC BC
DEC DE
DEC HL
DEC SP

Timing: M cycles — 1
T states — 6

Addressing Mode: Register

7	6	5	4	3	2	1	0
1	1	N_X	1	1	1	0	1
0	0	1	0	1	0	1	1

DEC IX (for $N_X = 0$)
DEC IY (for $N_X = 1$)

Timing: M cycles — 2
T states — 10 (4, 6)

Addressing Mode: Register

12.8 Bit Set, Reset, and Test

REGISTER

SET b, r

Bit b in register r is set.

R_b ← 1 No flags affected

7	6	5	4	3	2	1	0
1	1	0	0	1	0	1	1
1	1	b			r		

Timing: M cycles — 2
T states — 8 (4, 4)

Addressing Mode: Bit/Register

RES b, r

Bit b in register r is reset.

r_b ← 0 No flags affected

7	6	5	4	3	2	1	0
1	1	0	0	1	0	1	1
1	0	b			r		

Timing: M cycles — 2
T states — 8 (4, 4)

Addressing Mode: Bit/Register

BIT b, r

Bit b in register r is tested with the result put in the Z flag.

Z ← $\overline{r_b}$
S: Undefined
Z: Inverse of tested bit
H: Set
P/V: Undefined
N: Reset
C: N/A

7	6	5	4	3	2	1	0
1	1	0	0	1	0	1	1
0	1	b			r		

Timing: M cycles — 2
T states — 8 (4, 4)

Addressing Mode: Bit/Register

MEMORY

SET b, m_1

Bit b in memory location m_1 is set.

m_{1b} ← 1 No flags affected

7	6	5	4	3	2	1	0
1	1	0	0	1	0	1	1
1	1	b			1	1	0

SET b, (HL)

Timing: M cycles — 4
T states — 15 (4, 4, 4, 3)

Addressing Mode: Bit/Register Indirect

7	6	5	4	3	2	1	0
1	1	N_X	1	1	1	0	1
1	1	0	0	1	0	1	1
d							
1	1	b			1	1	0

SET b, (IX+d) (for $N_X = 0$)
SET b, (IY+d) (for $N_X = 1$)

Timing: M cycles — 6
T states — 23 (4, 4, 3, 5, 4, 3)

Addressing Mode: Bit/Indexed

RES b, m_1

Bit b in memory location m_1 is reset.

m_{1b} ← 0 No flags affected

7	6	5	4	3	2	1	0
1	1	0	0	1	0	1	1
1	0	b			1	1	0

RES b, (HL)

Timing: M cycles — 4
T states — 15 (4, 4, 4, 3)

Addressing Mode: Bit/Register Indirect

7	6	5	4	3	2	1	0
1	1	N_X	1	1	1	0	1
1	1	0	0	1	0	1	1
d							
1	0	b			1	1	0

RES b, (IX+d) (for $N_X = 0$)
RES b, (IY+d) (for $N_X = 1$)

Timing: M cycles — 6
T states — 23 (4, 4, 3, 5, 4, 3)

Addressing Mode: Bit/Indexed

12.8 Bit Set, Reset, and Test (Continued)

BIT B, m_1

Bit b in memory location m_1 is tested via the Z flag.

$Z \leftarrow \overline{m_{1b}}$

S: Undefined
Z: Inverse of tested bit
H: Set
P/V: Undefined
N: Reset
C: N/A

7	6	5	4	3	2	1	0	
1	1	0	0	1	0	1	1	BIT b, (HL)
0	1	b			1	1	0	

Timing: M cycles — 3
T states — 12 (4, 4, 4)
Addressing Mode: Bit/Register Indirect

7	6	5	4	3	2	1	0	
1	1	N_X	1	1	1	0	1	BIT b, (IX+d) (for $N_X=0$) BIT b, (IY+d) (for $N_X=1$)
1	1	0	0	1	0	1	1	
d								
0	1	b			1	1	0	

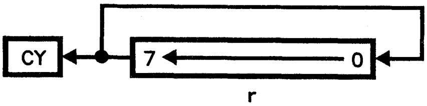
Timing: M cycles — 5
T states — 20 (4, 4, 3, 5, 4)
Addressing Mode: Bit/Indexed

12.9 Rotate and Shift

REGISTER

RLC r

Rotate register r left circular.



TL/C/5171-57

S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: Set according to bit 7 of r

7	6	5	4	3	2	1	0	
1	1	0	0	1	0	1	1	RLC r
0	0	0	0	0	r			(Note alternate for A register below)

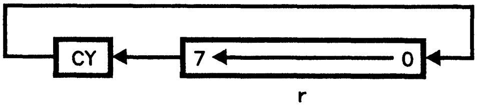
Timing: M cycles — 2
T states — 8 (4, 4)
Addressing Mode: Register

7	6	5	4	3	2	1	0	
0	0	0	0	0	1	1	1	RLCA

Timing: M cycles — 1
T states — 4
Addressing Mode: Implied
(Note RLCA does not affect S, Z, or P/V flags.)

RL r

Rotate register r left through carry.



TL/C/5171-58

S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: Set according to bit 7 of r

7	6	5	4	3	2	1	0	
1	1	0	0	1	0	1	1	RL r
0	0	0	1	0	r			(Note alternate for A register below)

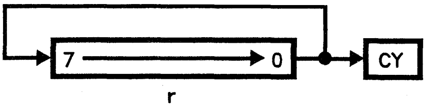
Timing: M cycles — 2
T states — 8 (4, 4)
Addressing Mode: Register

7	6	5	4	3	2	1	0	
0	0	0	1	0	1	1	1	RLA

Timing: M cycles — 1
T states — 4
Addressing Mode: Implied
(Note RLA does not affect S, Z, or P/V flags.)

RRC r

Rotate register r right circular.



TL/C/5171-59

S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: Set according to bit 0 of r

12.9 Rotate and Shift (Continued)

7	6	5	4	3	2	1	0
1	1	0	0	1	0	1	1

RRC r

0	0	0	0	1	r		

(Note alternate for A register below)

Timing: M cycles — 2
T states — 8 (4, 4)

Addressing Mode: Register

7	6	5	4	3	2	1	0
0	0	0	0	1	1	1	1

RRCA

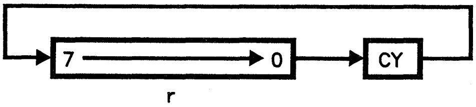
Timing: M cycles — 1
T states — 4

Addressing Mode: Implied

(Note RRCA does not affect S, Z, or P/V flags.)

RR r

Rotate register r right through carry.



TL/C/5171-60

S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: Set according to bit 0 of r

7	6	5	4	3	2	1	0
1	1	0	0	1	0	0	1

RR r

0	0	0	1	1	r		

(Note alternate for A register below)

Timing: M cycles — 2
T states — 8 (4, 4)

Addressing Mode: Register

7	6	5	4	3	2	1	0
0	0	0	1	1	1	1	1

RRA

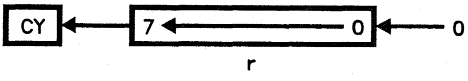
Timing: M cycles — 1
T states — 4

Addressing Mode: Implied

(Note RRA does not affect S, Z, or P/V flags.)

SLA r

Shift register r left arithmetic.



TL/C/5171-61

S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: Set according to bit 7 of r

7	6	5	4	3	2	1	0
1	1	0	0	1	0	1	1

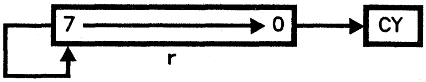
0	0	1	0	0	r		

Timing: M cycles — 2
T states — 8 (4, 4)

Addressing Mode: Register

SRA r

Shift register r right arithmetic.



TL/C/5171-62

S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: Set according to bit 0 of r

7	6	5	4	3	2	1	0
1	1	0	0	1	0	1	1

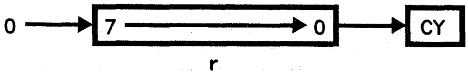
0	0	1	0	1	r		

Timing: M cycles — 2
T states — 8 (4, 4)

Addressing Mode: Register

SRL r

Shift register r right logical.



TL/C/5171-63

S: Reset
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: Set according to bit 0 of r

7	6	5	4	3	2	1	0
1	1	0	0	1	0	1	1

0	0	1	1	1	r		

Timing: M cycles — 2
T states — 8 (4, 4)

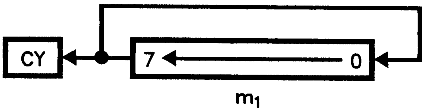
Addressing Mode: Register

12.9 Rotate and Shift (Continued)

MEMORY

RLC m_1

Rotate date in memory location m_1 left circular.



TL/C/5171-64

S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: Set according to bit 7 of m_1

7	6	5	4	3	2	1	0
1	1	0	0	1	0	1	1
0	0	0	0	0	1	1	0

RLC (HL)

Timing: M cycles — 4
T states — 15 (4, 4, 4, 3)

Addressing Mode: Register indirect

7	6	5	4	3	2	1	0
1	1	N_X	1	1	1	0	1
1	1	0	0	1	0	1	1
d							
0	0	0	0	0	1	1	0

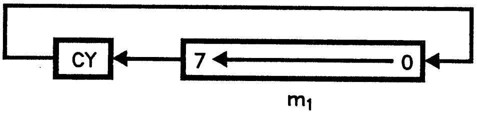
RLC (IX+d) (for N_X=0)
RLC (IY+d) (for N_X=1)

Timing: M cycles — 6
T states — 23 (4, 4, 3, 5, 4, 3)

Addressing Mode: Indexed

RL m_1

Rotate the data in memory location m_1 left though carry.



TL/C/5171-65

S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: Set according to bit 7 of m_1

7	6	5	4	3	2	1	0
1	1	0	0	1	0	1	1
0	0	0	1	0	1	1	0

RL (HL)

Timing: M cycles — 4
T states — 15 (4, 4, 4, 3)

Addressing Mode: Register Indirect

7	6	5	4	3	2	1	0
1	1	N_X	1	1	1	0	1
1	1	0	0	1	0	1	1
d							
0	0	0	1	0	1	1	0

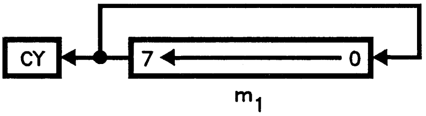
RL (IX+d) (for N_X=0)
RL (IY+d) (for NX=1)

Timing: M cycles — 6
T states — 23 (4, 4, 3, 5, 4, 3)

Addressing Mode: Indexed

RRC m_1

Rotate the data in memory location m_1 right circular.



TL/C/5171-66

S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: Set according to bit 0 of m_1

7	6	5	4	3	2	1	0
1	1	0	0	1	0	1	1
0	0	0	0	1	1	1	0

RRC (HL)

Timing: M cycles — 4
T states — 15 (4, 4, 4, 3)

Addressing Mode: Register Indirect

7	6	5	4	3	2	1	0
1	1	N_X	1	1	1	0	1
1	1	0	0	1	0	1	1
d							
0	0	0	0	1	1	1	0

RRC (IX + d) (for N_X = 0)
RRC (IY + d) (for N_X = 1)

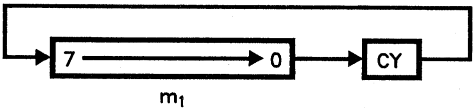
Timing: M cycles — 6
T states — 23 (4, 4, 3, 5, 4, 3)

Addressing Mode: Indexed

12.9 Rotate and Shift (Continued)

RR m_1

Rotate the data in memory location m_1 right through the carry.



TL/C/5171-67

S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: Set according to bit 0 of m_1

7	6	5	4	3	2	1	0	
1	1	0	0	1	0	1	1	RR (HL)
0	0	0	1	1	1	1	0	

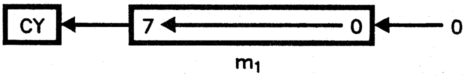
Timing: M cycles — 4
T states — 15 (4, 4, 4, 3)
Addressing Mode: Register Indirect

7	6	5	4	3	2	1	0	
1	1	N_X	1	1	1	0	1	RR (IX + d) (for $N_X = 0$)
1	1	0	0	1	0	1	1	RR (IY + d) (for $N_X = 1$)
d								
0	0	0	1	1	1	1	0	

Timing: M cycles — 6
T states — 23 (4, 4, 3, 5, 4, 3)
Addressing Mode: Indexed

SLA m_1

Shift the data in memory location m_1 left arithmetic.



TL/C/5171-68

S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: Set according to bit 7 of m_1

7	6	5	4	3	2	1	0	
1	1	0	0	1	0	1	1	SLA (HL)
0	0	1	0	0	1	1	0	

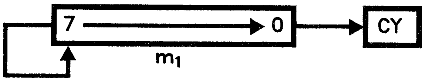
Timing: M cycles — 4
T states — 15 (4, 4, 4, 3)
Addressing Mode: Register Indirect

7	6	5	4	3	2	1	0	
1	1	N_X	1	1	1	0	1	SLA (IX + d) (for $N_X = 0$)
1	1	0	0	1	0	1	1	SLA (IY + d) (for $N_X = 1$)
d								
0	0	1	0	0	1	1	0	

Timing: M cycles — 6
T states — 23 (4, 4, 3, 5, 4, 3)
Addressing Mode: Indexed

SRA m_1

Shift the data in memory location m_1 right arithmetic.



TL/C/5171-69

S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: Set according to bit 0 of m_1

7	6	5	4	3	2	1	0	
1	1	0	0	1	0	1	1	SRA (HL)
0	0	1	0	1	1	1	0	

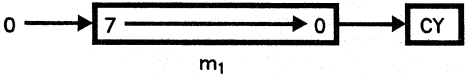
Timing: M cycles — 4
T states — 15 (4, 4, 4, 3)
Addressing Mode: Register Indirect

7	6	5	4	3	2	1	0	
1	1	N_X	1	1	1	0	1	SRA (IX + d) (for $N_X = 0$)
1	1	0	0	1	0	1	1	SRA (IY + d) (for $N_X = 1$)
d								
0	0	1	0	1	1	1	0	

Timing: M cycles — 6
T states — 23 (4, 4, 3, 5, 4, 3)
Addressing Mode: Indexed

SRL m_1

Shift right logical the data in memory location m_1.



TL/C/5171-70

S: Reset
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: Set according to bit 0 of m_1

12.9 Rotate and Shift (Continued)

7	6	5	4	3	2	1	0
1	1	0	0	1	0	1	1
0	0	1	1	1	1	1	0

SRL (HL)

Timing: M cycles — 4

T states — 15 (4, 4, 4, 3)

Addressing Mode: Register Indirect

7	6	5	4	3	2	1	0
1	1	N_X	1	1	1	0	1
1	1	0	0	1	0	1	1
d							
0	0	1	1	1	1	1	0

SRL (IX + d) (for N_X = 0)

SRL (IY + d) (for N_X = 1)

Timing: M cycles — 6

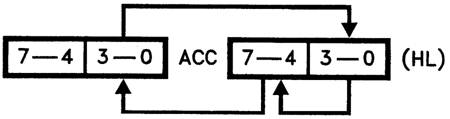
T states — 23 (4, 4, 3, 5, 4, 3)

Addressing Mode: Indexed

REGISTER/MEMORY

RLD

Rotate digit left and right between the Accumulator and memory (HL).



TL/C/5171-71

S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: N/A

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
0	1	1	0	1	1	1	1

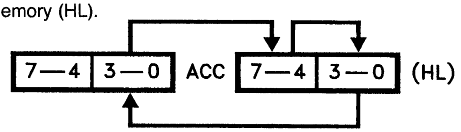
Timing: M cycles — 5

T states — 18 (4, 4, 3, 4, 3)

Addressing Mode: Implied/Register Indirect

RRD

Rotate digit right and left between the Accumulator and memory (HL).



TL/C/5171-72

S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: N/A

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
0	1	1	0	0	1	1	1

Timing: M cycles — 5

T states — 18 (4, 4, 3, 4, 3)

Addressing Mode: Implied/Register Indirect

12.10 Exchanges

REGISTER/REGISTER

EX DE, HL

Exchange the contents of the 16-bit register pairs DE and HL.

DE ⟷ HL No flags affected

7	6	5	4	3	2	1	0
1	1	1	0	1	0	1	1

Timing: M cycles — 1

T states — 4

Addressing Mode: Register

EX AF, A'F'

The contents of the Accumulator and flag register are exchanged with their corresponding alternate registers, that is A and F are exchanged with A' and F'.

A ⟷ A' No flags affected

F ⟷ F'

7	6	5	4	3	2	1	0
0	0	0	0	1	0	0	0

Timing: M cycles — 1

T states — 4

Addressing Mode: Register

12.10 Exchanges (Continued)

EXX

Exchange the contents of the BC, DE, and HL registers with their corresponding alternate register.

BC $\longleftrightarrow$ B'C'	No flags affected
DE $\longleftrightarrow$ D'E'	
HL $\longleftrightarrow$ H'L'	

7	6	5	4	3	2	1	0
1	1	0	1	1	0	0	1

Timing:	M cycles — 1
	T states — 4
Addressing Mode:	Implied

REGISTER/MEMORY

EX (SP), ss

Exchange the two bytes at the top of the external memory stack with the 16-bit register ss.

(SP) $\longleftrightarrow$ SS_L	No flags affected
(SP + 1) $\longleftrightarrow$ SS_H	

7	6	5	4	3	2	1	0	
1	1	1	0	0	0	1	1	EX (SP), HL

Timing:	M cycles — 5
	T states — 19 (4, 3, 4, 3, 5)
Addressing Mode:	Register/Register Indirect

7	6	5	4	3	2	1	0	
1	1	N_X	1	1	1	0	1	EX (SP), IX (for $N_X = 0$)
1	1	1	0	0	0	1	1	EX (SP),IY (for $N_X = 1$)

Timing:	M cycles — 6
	T states — 23 (4, 4, 3, 4, 3, 5)
Addressing Mode:	Register/Register Indirect

12.11 Memory Block Moves and Searches

SINGLE OPERATIONS

LDI

Move data from memory location (HL) to memory location (DE), increment memory pointers, and decrement byte counter BC.

(DE) $\leftarrow$ (HL)	S: N/A
DE $\leftarrow$ DE + 1	Z: N/A
HL $\leftarrow$ HL + 1	H: Reset
BC $\leftarrow$ BC − 1	P/V: Set if BC −1 $\neq$0, otherwise reset
	N: Reset
	C: N/A

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
1	0	1	0	0	0	0	0

Timing:	M cycles — 4
	T states — 16 (4, 4, 3, 5)
Addressing Mode:	Register Indirect

LDD

Move data from memory location (HL) to memory location (DE), and decrement memory pointer and byte counter BC.

(DE) $\leftarrow$ (HL)	S: N/A
DE $\leftarrow$ DE − 1	Z: N/A
HL $\leftarrow$ HL − 1	H: Reset
BC $\leftarrow$ BC − 1	P/V: Set if BC −1 $\neq$0, otherwise reset
	N: Reset
	C: N/A

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
1	0	1	0	1	0	0	0

Timing:	M cycles — 4
	T states — 16 (4, 4, 3, 5)
Addressing Mode:	Register Indirect

CPI

Compare data in memory location (HL) to the Accumulator, increment the memory pointer, and decrement the byte counter. The Z flag is set if the comparison is equal.

A − (HL)	S: Set if result of comparison subtract is negative
HL $\leftarrow$ HL + 1	
BC $\leftarrow$ BC − 1	Z: Set if result of comparison is zero
Z $\leftarrow$ 1 if A = (HL)	
	H: Set according to borrow from bit 4
	P/V: Set if BC − 1$\neq$ 0, otherwise reset
	N: Set
	C: N/A

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
1	0	1	0	0	0	0	1

Timing:	M cycles — 4
	T states — 16 (4, 4, 3, 5)
Addressing Mode:	Register Indirect

CPD

Compare data in memory location (HL) to the Accumulator, and decrement the memory pointer and byte counter. The Z flag is set if the comparison is equal.

A − (HL)	S: Set if result is negative
HL $\leftarrow$ HL − 1	Z: Set if result of comparison is zero
BC $\leftarrow$ BC − 1	
Z $\leftarrow$ 1 if A = (HL)	H: Set according to borrow from bit 4
	P/V: Set if BC − 1$\neq$ 0, otherwise reset
	N: Set
	C: N/A

12.11 Memory Block Moves and Searches (Continued)

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
1	0	1	0	1	0	0	1

Timing:	M cycles — 4
	T states — 16 (4, 4, 3, 5)
Addressing Mode:	Register Indirect

REPEAT OPERATIONS

LDIR

Move data from memory location (HL) to memory location (DE), increment memory pointers, decrement byte counter BC, and repeat until BC = 0.

(DE) ← (HL)	S: N/A
DE ← DE + 1	Z: N/A
HL ← HL + 1	H: Reset
BC ← BC − 1	P/V: Reset
Repeat until	N: Reset
BC = 0	C: N/A

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
1	0	1	1	0	0	0	0

Timing:	For BC≠0	M cycles — 5
		T states — 21 (4, 4, 3, 5, 5)
	For BC=0	M cycles — 4
		T states — 16 (4, 4, 3, 5)
Addressing Mode:		Register Indirect

(Note that each repeat is accomplished by a decrement of the BC, so that refresh, etc. continues for each cycle.)

LDDR

Move data from memory location (HL) to memory location (DE), decrement memory pointers and byte counter BC, and repeat until BC = 0.

(DE) ← (HL)	S: N/A
DE ← DE − 1	Z: N/A
HL ← HL − 1	H: Reset
BC ← BC − 1	P/V: Reset
Repeat until	N: Reset
BC = 0	C: N/A

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
1	0	1	1	1	0	0	0

Timing:	For BC≠0	M cycles — 5
		T states — 21 (4, 4, 3, 5, 5)
	For BC=0	M cycles — 4
		T states — 16 (4, 4, 3, 5)
Addressing Mode:		Register Indirect

(Note that each repeat is accomplished by a decrement of the BC, so that refresh, etc. continues for each cycle.)

CPIR

Compare data in memory location (HL) to the Accumulator, increment the memory, decrement the byte counter BC, and repeat until BC = 0 or (HL) equals A.

A − (HL)	S: Set if sign of subtraction performed for comparison is negative
HL ← HL + 1	Z: Set if A = (HL), otherwise reset
BC ← BC + 1	H: Set according to borrow from bit 4
Repeat until BC = 0 or A = (HL)	P/V: Set if BC − 1 ≠ 0, otherwise reset
	N: Set
	C: N/A

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
1	0	1	1	0	0	0	1

Timing:	For BC ≠ 0	M cycles — 5
		T states — 21 (4, 4, 3, 5, 5)
	For BC = 0	M cycles — 4
		T states — 16 (4, 4, 3, 5)
Addressing Mode:		Register Indirect

(Note that each repeat is accomplished by a decrement of the PC, so that refresh, etc. continues for each cycle.)

CPDR

Compare data in memory location (HL) to the contents of the Accumulator, decrement the memory pointer and byte counter BC, and repeat until BC = 0, or until (HL) equals the Accumulator.

A − (HL)	S: Set if sign of subtraction performed for comparison is negative
HL ← HL − 1	Z: Set according to equality of A and (HL), set if true
BC ← BC − 1	H: Set according to borrow from bit 4
Repeat until BC = 0 or A = (HL)	P/V: Set if BC − 1 ≠ 0, otherwise reset
	N: Set
	C: N/A

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
1	0	1	1	1	0	0	1

Timing:	For BC ≠ 0	M cycles — 5
		T states — 21 (4, 4, 3, 5, 5)
	For BC = 0	M cycles — 4
		T states — 16 (4, 4, 3, 5)
Addressing Mode:		Register Indirect

(Note that each repeat is accomplished by a decrement of the BC, so that refresh, etc. continues for each cycle.)

12.12 Input/Output

IN A, (n)

Input data to the Accumulator from the I/O device at address N.

A ← (n)   No flags affected

7	6	5	4	3	2	1	0
1	1	0	1	1	0	1	1
n							

Timing: M cycles — 3
T states — 11 (4, 3, 4)

Addressing Mode: Source — Direct
Destination — Register

IN r, (C)

Input data to register r from the I/O device addressed by the contents of register C. If r=110 only flags are affected.

r ← (C)

S: Set if result is negative
Z: Set if result is zero
H: Reset
P/V: Set if result parity is even
N: Reset
C: N/A

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
0	1	r			0	0	0

Timing: M cycles — 3
T states — 12 (4, 4, 4)

Addressing Mode: Source — Register Indirect
Destination — Register

OUT (C), r

Output register r to the I/O device addressed by the contents of register C.

(C) ← r   No flags affected

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
0	1	r			0	0	1

Timing: M cycles — 3
T states — 12 (4, 4, 4)

Addressing Mode: Source — Register
Destination — Register Indirect

INI

Input data from the I/O device addressed by the contents of register C to the memory location pointed to by the contents of the HL register. The HL pointer is incremented and the byte counter B is decremented.

(HL) ← (C)
B ← B − 1
HL ← HL + 1

S: Undefined
Z: Set if B−1=0, otherwise reset
H: Undefined
P/V: Undefined
N: Set
C: N/A

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
1	0	1	0	0	0	1	0

Timing: M cycles — 4
T states — 16 (4, 5, 3, 4)

Addressing Mode: Implied/Source — Register Indirect
Destination — Register Indirect

OUTI

Output data from memory location (HL) to the I/O device at port address (C), increment the memory pointer, and decrement the byte counter B.

(C) ← (HL)
B ← B − 1
HL ← HL + 1

S: Undefined
Z: Set if B−1=0, otherwise reset
H: Undefined
P/V: Undefined
N: Set
C: N/A

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
1	0	1	0	0	0	1	1

Timing: M cycles — 4
T states — 16 (4, 5, 3, 4)

Addressing Mode: Implied/Source — Register Indirect
Destination — Register Indirect

IND

Input data from I/O device at port address (C) to memory location (HL), and decrement HL memory pointer and byte counter B.

(HL) ← (C)
HL ← HL − 1
B ← B − 1

S: Undefined
Z: Set if B−1=0, otherwise reset
H: Undefined
P/V: Undefined
N: Set
C: N/A

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
1	0	1	0	1	0	1	0

Timing: M cycles — 4
T states — 16 (4, 5, 3, 4)

Addressing Mode: Implied/Source — Register Indirect
Destination — Register Indirect

12.12 Input/Output (Continued)

OUT (n), A

Output the Accumulator to the I/O device at address n.

(n) ← A	No flags affected

7	6	5	4	3	2	1	0
1	1	0	1	0	0	1	1
n							

Timing:	M cycles — 3
	T states — 11 (4, 3, 4)
Addressing Mode:	Source — Register
	Destination — Direct

OUTD

Data is output from memory location (HL) to the I/O device at port address (C), and the HL memory pointer and byte counter B are decremented.

(C) ← (HL)	S: Undefined
B ← B − 1	Z: Set if B−1=0, otherwise reset
HL ← HL − 1	H: Undefined
	P/V: Undefined
	N: Set
	C: N/A

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
1	0	1	0	1	0	1	1

Timing:	M cycles — 4
	T states — 16 (4, 5, 3, 4)
Addressing Mode:	Implied/Source — Register Indirect
	Destination — Register Indirect

INIR

Data is input from the I/O device at port address (C) to memory location (HL), the HL memory pointer is incremented, and the byte counter B is decremented. The cycle is repeated until B = 0.

(Note that B is tested for zero after it is decremented. By loading B initially with zero, 256 data transfers will take place.)

(HL) ← (C)	S: Undefined
HL ← HL + 1	Z: Set
B ← B − 1	H: Undefined
Repeat until B = 0	P/V: Undefined
	N: Set
	C: N/A

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
1	0	1	1	0	0	1	0

Timing:	For B ≠ 0	M cycles — 5
		T states — 21 (4, 5, 3, 4, 5)
	For B = 0	M cycles — 4
		T states — 16 (4, 5, 3, 4)
Addressing Mode:		Implied/Source — Register Indirect
		Destination — Register Indirect

(Note that at the end of each data transfer cycle, interrupts may be recognized and two refresh cycles will be performed.)

OTIR

Data is output to the I/O device at port address (C) from memory location (HL), the HL memory pointer is incremented, and the byte counter B is decremented. The cycles are repeated until B = 0.

(Note that B is tested for zero after it is decremented. By loading B initially with zero, 256 data transfers will take place.)

(C) ← (HL)	S: Undefined
HL ← HL + 1	H: Undefined
B ← B − 1	Z: Set
Repeat until B = 0	P/V: Undefined
	N: Set
	C: N/A

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
1	0	1	1	0	0	1	1

Timing:	For B ≠ 0	M cycles — 5
		T states — 21 (4, 5, 3, 4, 5)
	For B = 0	M cycles — 4
		T states — 16 (4, 5, 3, 4)
Addressing Mode:		Implied/Source — Register Indirect
		Destination — Register Indirect

(Note that at the end of each data transfer cycle, interrupts may be recognized and two refresh cycles will be performed.)

12.12 Input/Output (Continued)

INDR

Data is input from the I/O device at address (C) to memory location (HL), then the HL memory pointer is byte counter B are decremented. The cycle is repeated until B = 0.

(Note that B is tested for zero after it is decremented. By loading B initially with zero, 256 data transfers will take place.)

(HL) ← (C)	S: Undefined
HL ← HL − 1	Z: Set
B ← B − 1	H: Undefined
Repeat until B = 0	P/V: Undefined
	N: Set
	C: N/A

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
1	0	1	1	0	0	1	0

Timing:	For B ≠ 0	M cycles — 5
		T states — 21 (4, 5, 3, 4, 5)
	For B = 0	M cycles — 4
		T states — 16 (4, 5, 3, 4)
Addressing Mode:		Implied/Source — Register Indirect
		Destination — Register Indirect

(Note that after each data transfer cycle, interrupts may be recognized and two refresh cycles are performed.)

OTDR

Data is output from memory location (HL) to the I/O device at port address (C), then the HL memory pointer and byte counter B are decremented. The cycle is repeated until B = 0.

(Note that B is tested for zero after it is decremented. By loading B initially with zero, 256 data transfers will take place.)

(C) ← (HL)	S: Undefined
HL ← HL − 1	Z: Set
B ← B − 1	H: Undefined
Repeat until B = 0	P/V: Undefined
	N: Set
	C: N/A

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
1	0	1	1	1	0	1	1

Timing:	For B ≠ 0	M cycles — 5
		T states — 21 (4, 5, 3, 4, 5)
	For B = 0	M cycles — 4
		T states — 16 (4, 5, 3, 4)
Addressing Mode:		Implied/Source — Register Indirect
		Destination — Register Indirect

(Note that after each data transfer cycle the NSC800 will accept interrupts and perform two refresh cycles.)

12.13 CPU Control

NOP

The CPU performs no operation.

— — —	No flags affected

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0

Timing:	M cycles — 1
	T states — 4
Addressing Mode:	N/A

HALT

The CPU halts execution of the program. Dummy op-code fetches are performed from the next memory location to keep the refresh circuits active until the CPU is interrupted or reset from the halted state.

— — —	No flags affected

7	6	5	4	3	2	1	0
0	1	1	1	0	1	1	0

Timing:	M cycles — 1
	T states — 4
Addressing Mode:	N/A

DI

Disable system level interrupts.

IFF_1 ← 0	No flags affected
IFF_2 ← 0	

7	6	5	4	3	2	1	0
1	1	1	1	0	0	1	1

Timing:	M cycles — 1
	T states — 4
Addressing Mode:	N/A

EI

The system level interrupts are enabled. During execution of this instruction, and the next one, the maskable interrupts will be disabled.

IFF_1 ← 1	No flags affected
IFF_2 ← 1	

7	6	5	4	3	2	1	0
1	1	1	1	1	0	1	1

Timing:	M cycles — 1
	T states — 4
Addressing Mode:	N/A

IM 0

The CPU is placed in interrupt mode 0.

— — —	No flags affected

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
0	1	0	0	0	1	1	0

Timing:	M cycles — 2
	T states — 8 (4, 4)
Addressing Mode:	N/A

12.13 CPU Control (Continued)

IM 1

The CPU is placed in interrupt mode 1.

— — — No flags affected

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
0	1	0	1	0	1	1	0

Timing: M cycles — 2
T states — 8 (4, 4)
Addressing Mode: N/A

IM 2

The CPU is placed in interrupt mode 2.

— — — No flags affected

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
0	1	0	1	1	1	1	0

Timing: M cycles — 2
T states — 8 (4, 4)
Addressing Mode: N/A

12.14 Program Control

JUMPS

JP nn

Unconditional jump to program location nn.

PC ← nn No flags affected

7	6	5	4	3	2	1	0
1	1	0	0	0	0	1	1
n (low-order byte)							
n (high-order byte)							

Timing: M cycles — 3
T states — 10 (4, 3, 3)
Addressing Mode: Direct

JP (ss)

Unconditional jump to program location pointed to by register ss.

PC ← ss No flags affected

7	6	5	4	3	2	1	0	
1	1	1	0	1	0	0	1	JP (HL)

Timing: M cycles — 1
T states — 4
Addressing Mode: Register Indirect

7	6	5	4	3	2	1	0	
1	1	N_X	1	1	1	0	1	JP (IX) (for $N_X = 0$) JP (IY) (for $N_X = 1$)
1	1	1	0	1	0	0	1	

Timing: M cycles — 2
T states — 8 (4, 4)
Addressing Mode: Register Indirect

JP cc, nn

Conditionally jump to program location nn based on testable flag states.

If cc true, No flags affected
PC ← nn,
otherwise continue

7	6	5	4	3	2	1	0
1	1	cc			0	1	0
n (low-order byte)							
n (high-order byte)							

Timing: M cycles — 3
T states — 10 (4, 3, 3)
Addressing Mode: Direct

JR d

Unconditional jump to program location calculated with respect to the program counter and the displacement d.

PC ← PC + d No flags affected

7	6	5	4	3	2	1	0
0	0	0	1	1	0	0	0
d − 2							

Timing: M cycles — 3
T states — 12 (4, 3, 5)
Addressing Mode: PC Relative

JR kk, d

Conditionally jump to program location calculated with respect to the program counter and the displacement d, based on limited testable flag states.

If kk true, No flags affected
PC ← PC + d,
otherwise continue

7	6	5	4	3	2	1	0
0	0	1	kk		0	0	0
d − 2							

Timing: if kk met M cycles — 3
(true) T states — 12 (4, 3, 5)
if kk not met M cycles — 2
(not true) T states — 7 (4, 3)
Addressing Mode: PC Relative

12.14 Program Control (Continued)

DJNZ d

Decrement the B register and conditionally jump to program location calculated with respect to the program counter and the displacement d, based on the contents of the B register.

B ← B − 1 No flags affected

If B = 0 continue,

else PC ← PC + d

7	6	5	4	3	2	1	0
0	0	0	1	0	0	0	0
d − 2							

Timing: If B ≠ 0 M cycles — 3

T states — 13 (5, 3, 5)

If B = 0 M cycles — 2

T states — 8 (5, 3)

Addressing Mode: PC Relative

CALLS

CALL nn

Unconditional call to subroutine at location nn.

(SP − 1) ← PC_H No flags affected

(SP − 2) ← PC_L

SP ← SP − 2

PC ← nn

7	6	5	4	3	2	1	0
1	1	0	0	1	1	0	1
n (low-order byte)							
n (high-order byte)							

Timing: M Cycles — 5

T states — 17 (4, 3, 4, 3, 3)

Addressing Mode: Direct

CALL cc, nn

Conditional call to subroutine at location nn based on testable flag stages.

If cc true, No flags affected

(SP − 1) ← PC_H

(SP − 2) ← PC_L

SP ← SP − 2

PC ← nn,

else continue

7	6	5	4	3	2	1	0
1	1	cc			1	0	0
n (low-order byte)							
n (high-order byte)							

Timing: If cc true M cycles — 5

T states 17 (4, 3, 4, 3, 3)

If cc not true M cycles — 3

T states — 10 (4, 3, 3)

Addressing Mode: Direct

RETURNS

RET

Unconditional return from subroutine or other return to program location pointed to by the top of the stack.

PC_L ← (SP) No flags affected

PC_H ← (SP + 1)

SP ← SP + 2

7	6	5	4	3	2	1	0
1	1	0	0	1	0	0	1

Timing: M cycles — 3

T states — 10 (4, 3, 3)

Addressing Mode: Register Indirect

RET cc

Conditional return from subroutine or other return to program location pointed to by the top of the stack.

If cc true, No flags affected

PC_L ← (SP)

PC_H ← (SP + 1)

SP ← SP + 2,

else continue

7	6	5	4	3	2	1	0
1	1	cc			0	0	0

Timing: If cc true M cycles — 3

T states — 11 (5, 3, 3)

If cc not true M cycles — 1

T states — 5

Addressing Mode: Register Indirect

RETI

Unconditional return from interrupt handling subroutine. Functionally identical to RET instruction. Unique opcode allows monitoring by external hardware.

PC_L ← (SP) No flags affected

PC_H ← (SP + 1)

SP ← SP + 2

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
0	1	0	0	1	1	0	1

Timing: M cycles — 4

T states — 14 (4, 4, 3, 3)

Addressing Mode: Register Indirect

12.14 Program Control (Continued)

RETN

Unconditional return from non-maskable interrupt handling subroutine. Functionally similar to RET instruction, except interrupt enable state is restored to that prior to non-maskable interrupt.

$PC_L \leftarrow (SP)$ No flags affected

$PC_H \leftarrow (SP + 1)$

$SP \leftarrow SP + 2$

$IFF_1 \leftarrow IFF_2$

7	6	5	4	3	2	1	0
1	1	1	0	1	1	0	1
0	1	0	0	0	1	0	1

Timing: M cycles — 4

T states — 14 (4, 4, 3, 3)

Addressing Mode: Register Indirect

RESTARTS

RST P

The present contents of the PC are pushed onto the memory stack and the PC is loaded with dedicated program locations as determined by the specific restart executed.

$(SP - 1) \leftarrow PC_H$ No flags affected

$(SP - 2) \leftarrow PC_L$

$SP \leftarrow SP - 2$

$PC_H \leftarrow 0$

$PC_L \leftarrow P$

7	6	5	4	3	2	1	0
1	1	t			1	1	1

Timing: M cycles — 3

T states — 11 (5, 3, 3)

Addressing Mode: Modified Page Zero

p	00H	08H	10H	18H	20H	28H	30H	38H
t	000	001	010	011	100	101	110	111

12.15 Instruction Set: Alphabetical Order

ADC	A, (HL)	8E
ADC	A, (IX+d)	DD 8Ed
ADC	A, (IY+d)	FD 8Ed
ADC	A, A	8F
ADC	A, B	88
ADC	A, C	89
ADC	A, D	8A
ADC	A, E	8B
ADC	A, H	8C
ADC	A, L	8D
ADC	A, n	CE n
ADC	HL, BC	ED 4A
ADC	HL, DE	ED 5A
ADC	HL, HL	ED 6A
ADC	HL, SP	ED 7A
ADD	A, (HL)	86
ADD	A, (IX+d)	DD 86d
ADD	A, (IY+d)	FD 86d
ADD	A, A	87
ADD	A, B	80
ADD	A, C	81
ADD	A, D	82
ADD	A, E	83
ADD	A, H	84
ADD	A, L	85
ADD	A, n	C6 n
ADD	HL, BC	09
ADD	HL, DE	19
ADD	HL, HL	29
ADD	HL, SP	39
ADD	IX, BC	DD 09
ADD	IX, DE	DD 19
ADD	IX, IX	DD 29
ADD	IX, SP	DD 39
ADD	IY, BC	FD 09
ADD	IY, DE	FD 19
ADD	IY, IY	FD 29
ADD	IY, SP	FD 39
AND	(HL)	A6
AND	(IX+d)	DD A6d
AND	(IY+d)	FD A6d
AND	A	A7
AND	B	A0
AND	C	A1
AND	D	A2
AND	E	A3
AND	H	A4
AND	L	A5
AND	n	E6 n
BIT	0, (HL)	CB 46
BIT	0, (IX+d)	DD CBd46
BIT	0, (IY+d)	FD CBd46
BIT	0, A	CB 47
BIT	0, B	CB 40
BIT	0, C	CB 41
BIT	0, D	CB 42
BIT	0, E	CB 43
BIT	0, H	CB 44
BIT	0, L	CB 45
BIT	1, (HL)	CB 4E
BIT	1, (IX+d)	DD CBd4E
BIT	1, (IY+d)	FD CBd4E
BIT	1, A	CB 4F
BIT	1, B	CB 48
BIT	1, C	CB 49
BIT	1, D	CB 4A
BIT	1, E	CB 4B
BIT	1, H	CB 4C
BIT	1, L	CB 4D
BIT	2, (HL)	CB 56
BIT	2, (IX+d)	DD CBd56
BIT	2, (IY+d)	FD CBd56
BIT	2, A	CB 57
BIT	2, B	CB 50
BIT	2, C	CB 51
BIT	2, D	CB 52
BIT	2, E	CB 53
BIT	2, H	CB 54
BIT	2, L	CB 55
BIT	3, (HL)	CB 5E
BIT	3, (IX+d)	DD CBd5E
BIT	3, (IY+d)	FD CBd5E
BIT	3, A	CB 5F
BIT	3, B	CB 58
BIT	3, C	CB 59
BIT	3, D	CB 5A
BIT	3, E	CB 5B
BIT	3, H	CB 5C
BIT	3, L	CB 5D
BIT	4, (HL)	CB 66
BIT	4, (IX+d)	DD CBd66
BIT	4, (IY+d)	FD CBd66
BIT	4, A	CB 67
BIT	4, B	CB 60
BIT	4, C	CB 61
BIT	4, D	CB 62
BIT	4, E	CB 63
BIT	4, H	CB 64
BIT	4, L	CB 65
BIT	5, (HL)	CB 6E
BIT	5, (IX+d)	DD CBd6E
BIT	5, (IY+d)	FD CBd6E
BIT	5, A	CB 6F
BIT	5, B	CB 68
BIT	5, C	CB 69
BIT	5, D	CB 6A

(nn)=address of memory location

nn=Data (16 bit)

n=Data (8 bit)

d=signed displacement

d2=d−2

12.15 Instruction Set: Alphabetical Order (Continued)

BIT	5, E	CB 6B
BIT	5, H	CB 6C
BIT	5, L	CB 6D
BIT	6, (HL)	CB 76
BIT	6, (IX+d)	DD CBd76
BIT	6, (IY+d)	FD CBd76
BIT	6, A	CB 77
BIT	6, B	CB 70
BIT	6, C	CB 71
BIT	6, D	CB 72
BIT	6, E	CB 73
BIT	6, H	CB 74
BIT	6, L	CB 75
BIT	7, (HL)	CB 7E
BIT	7, (IX+d)	DD CBd7E
BIT	7, (IY+d)	FD CBd7E
BIT	7, A	CB 7F
BIT	7, B	CB 78
BIT	7, C	CB 79
BIT	7, D	CB 7A
BIT	7, E	CB 7B
BIT	7, H	CB 7C
BIT	7, L	CB 7D
CALL	C, nn	DCnn
CALL	M, nn	FCnn
CALL	NC, nn	D4nn
CALL	nn	CDnn
CALL	NZ, nn	C4nn
CALL	P, nn	F4nn
CALL	PE, nn	ECnn
CALL	PO, nn	E4nn
CALL	Z, nn	CCnn
CCF		3F
CP	(HL)	BE
CP	(IX+d)	DD BEd
CP	(IY+d)	FD BEd
CP	A	BF
CP	B	B8
CP	C	B9
CP	D	BA
CP	E	BB
CP	H	BC
CP	L	BD
CP	n	FE n
CPD		ED A9
CPDR		ED B9
CPI		ED A1
CPIR		ED B1
CPL		2F
DAA		27
DEC	(HL)	35
DEC	(IX+d)	DD 35d
DEC	(IY+d)	FD 35d
DEC	A	3D
DEC	B	05
DEC	BC	0B
DEC	C	0D
DEC	D	15
DEC	DE	1B
DEC	E	1D
DEC	H	25
DEC	HL	2B
DEC	IX	DD 2B
DEC	IY	FD 2B
DEC	L	2D
DEC	SP	3B
DI		F3
DJNZ	d2	10 d2
EI		FB
EX	(SP), HL	E3
EX	(SP), IX	DD E3
EX	(SP), IY	FD E3
EX	AF, A'F'	08
EX	DE, HL	EB
EXX		D9
HALT		76
IM	0	ED 46
IM	1	ED 56
IM	2	ED 5E
IN	A, (C)	ED78
IN	A, (n)	DB n
IN	B, (C)	ED 40
IN	C, (C)	ED 48
IN	D, (C)	ED 50
IN	E, (C)	ED 58
IN	H, (C)	ED 60
IN	L, (C)	ED 68
INC	(HL)	34
INC	(IX+d)	DD 34d
INC	(IY+d)	FD 34d
INC	A	3C
INC	B	04
INC	BC	03
INC	C	0C
INC	D	14
INC	DE	13
INC	E	1C
INC	H	24
INC	HL	23
INC	IX	DD 23
INC	IY	FD 23
INC	L	2C
INC	SP	33
IND		ED AA
INDR		ED BA
INI		ED A2

(nn) = Address of memory location
nn = Data (16 bit)
n = Data (8 bit)

d = signed displacement
d2 = d − 2

12.15 Instruction Set: Alphabetical Order (Continued)

INIR		ED B2
JP	(HL)	E9
JP	(IX)	DD E9
JP	(IY)	FD E9
JP	C, nn	DAnn
JP	M, nn	FAnn
JP	NC, nn	D2nn
JP	nn	C3nn
JP	NZ, nn	C2nn
JP	P, nn	F2nn
JP	PE, nn	EAnn
JP	PO, nn	E2nn
JP	Z, nn	CAnn
JR	C, d2	38 d2
JR	d2	18 d2
JR	NC, d2	30 d2
JR	NZ, d2	20 d2
JR	Z, d2	28 d2
LD	(BC), A	02
LD	(DE), A	12
LD	(HL), A	77
LD	(HL), B	70
LD	(HL), C	71
LD	(HL), D	72
LD	(HL), E	73
LD	(HL), H	74
LD	(HL), L	75
LD	(HL), n	36 n
LD	(IX+d), A	DD 77d
LD	(IX+d), B	DD 70d
LD	(IX+d), C	DD 71d
LD	(IX+d), D	DD 72d
LD	(IX+d), E	DD 73d
LD	(IX+d), H	DD 74d
LD	(IX+d), L	DD 75d
LD	(IX+d), n	DD 36dn
LD	(IY+d), A	FD 77d
LD	(IY+d), B	FD 70d
LD	(IY+d), C	FD 71d
LD	(IY+d), D	FD 72d
LD	(IY+d), E	FD 73d
LD	(IY+d), H	FD 74d
LD	(IY+d), L	FD 75d
LD	(IY+d), n	FD 36dn
LD	(nn), A	32nn
LD	(nn), BC	ED 43nn
LD	(nn), DE	ED 53nn
LD	(nn), HL	22nn
LD	(nn), IX	DD 22nn
LD	(nn), IY	FD 22nn
LD	(nn), SP	ED 73nn
LD	A, (BC)	0A
LD	A, (DE)	1A
LD	A, (HL)	7E
LD	A, (IX+d)	DD 7Ed
LD	A, (IY+d)	FD 7Ed
LD	A, (nn)	3Ann
LD	A, A	7F
LD	A, B	78
LD	A, C	79
LD	A, D	7A
LD	A, E	7B
LD	A, H	7C
LD	A, I	ED 57
LD	A, L	7D
LD	A, n	3E n
LD	B, (HL)	46
LD	B, (IX+d)	DD 46d
LD	B, (IY+d)	FD 46d
LD	B, A	47
LD	B, B	40
LD	B, C	41
LD	B, D	42
LD	B, E	43
LD	B, H	44
LD	B, L	45
LD	B, n	06 n
LD	BC, (nn)	ED 4B
LD	BC, nn	01nn
LD	C, (HL)	4E
LD	C, (IX+d)	DD 4Ed
LD	C, (IY+d)	FD 4Ed
LD	C, A	4F
LD	C, B	48
LD	C, C	49
LD	C, D	4A
LD	C, E	4B
LD	C, H	4C
LD	C, L	4D
LD	C, n	0E n
LD	D, (HL)	56
LD	D, (IX+d)	DD 56d
LD	D, (IY+d)	FD 56d
LD	D, A	57
LD	D, B	50
LD	D, C	51
LD	D, D	52
LD	D, E	53
LD	D, H	54
LD	D, L	55
LD	D, n	16 n
LD	DE, (nn)	ED 5Bnn
LD	DE, nn	11nn
LD	E, (HL)	5E
LD	E, (IX+d)	DD 5Ed
LD	E, (IY+d)	FD 5Ed

(nn)=Address of memory location   d=signed displacement

nn=Data (16 bit)   d2=d−2

n=Data (8 bit)

12.15 Instruction Set: Alphabetical Order (Continued)

LD	E, A	5F
LD	E, B	58
LD	E, C	59
LD	E, D	5A
LD	E, E	5B
LD	E, H	5C
LD	E, L	5D
LD	E, n	1E n
LD	H, (HL)	66
LD	H, (IX+d)	DD 66d
LD	H, (IY+d)	FD 66d
LD	H, A	67
LD	H, B	60
LD	H, C	61
LD	H, D	62
LD	H, E	63
LD	H, H	64
LD	H, L	65
LD	H, n	26 n
LD	HL, (nn)	2Ann
LD	HL, nn	21nn
LD	I, A	ED 47
LD	IX, (nn)	DD 2Ann
LD	IX, nn	DD 21nn
LD	IY, (nn)	FD 2Ann
LD	IY, nn	FD 21nn
LD	L, (HL)	6E
LD	L, (IX+d)	DD 6Ed
LD	L, (IY+d)	FD 6Ed
LD	L, A	6F
LD	L, B	68
LD	L, C	69
LD	L, D	6A
LD	L, E	6B
LD	L, H	6C
LD	L, L	6D
LD	L, n	2E n
LD	SP, (nn)	ED 7Bnn
LD	SP, HL	F9
LD	SP, IX	DD F9
LD	SP, IY	FD F9
LD	SP, nn	31nn
LDD		ED A8
LDDR		ED B8
LDI		ED A0
LDIR		ED B0
NEG		ED n
NOP		00
OR	(HL)	B6
OR	(IX+d)	DD B6d
OR	(IY+d)	FD B6d
OR	A	B7
OR	B	B0
OR	C	B1
OR	D	B2
OR	E	B3
OR	H	B4
OR	L	B5
OR	n	F6 n
OTDR		ED BB
OTIR		ED B3
OUT	(C), A	ED 79
OUT	(C), B	ED 41
OUT	(C), C	ED 49
OUT	(C), D	ED 51
OUT	(C), E	ED 59
OUT	(C), H	ED 61
OUT	(C), L	ED 69
OUT	n, A	D3 n
OUTD		ED AB
OUTI		ED A3
POP	AF	F1
POP	BC	C1
POP	DE	D1
POP	HL	E1
POP	IX	DD E1
POP	IY	FD E1
PUSH	AF	F5
PUSH	BC	C5
PUSH	DE	D5
PUSH	HL	E5
PUSH	IX	DD E5
PUSH	IY	FD E5
RES	0, (HL)	CB 86
RES	0, (IX+d)	DD CBd86
RES	0, (IY+d)	FD CBd86
RES	0, A	CB 87
RES	0, B	CB 80
RES	0, C	CB 81
RES	0, D	CB 82
RES	0, E	CB 83
RES	0, H	CB 84
RES	0, L	CB 85
RES	1, (HL)	CB 8E
RES	1, (IX+d)	DD CBd8E
RES	1, (IY+d)	FD CBd8E
RES	1, A	CB 8F
RES	1, B	CB 88
RES	1, C	CB 89
RES	1, D	CB 8A
RES	1, E	CB 8B
RES	1, H	CB 8C
RES	1, L	CB 8D
RES	2, (HL)	CB 96
RES	2, (IX+d)	DD CBd96
RES	2, (IY+d)	FD CBd96

(nn) = Address of memory location
nn = Data (16 bit)
n = Data (8 bit)

d = signed displacement
d2 = d − 2

12.15 Instruction Set: Alphabetical Order (Continued)

RES	2, A	CB 97
RES	2, B	CB 90
RES	2, C	CB 91
RES	2, D	CB 92
RES	2, E	CB 93
RES	2, H	CB 94
RES	2, L	CB 95
RES	3, (HL)	CB 9E
RES	3, (IX+d)	DD CBd9E
RES	3, (IY+d)	FD CBd9E
RES	3, A	CB 9F
RES	3, B	CB 98
RES	3, C	CB 99
RES	3, D	CB 9A
RES	3, E	CB 9B
RES	3, H	CB 9C
RES	3, L	CB 9D
RES	4, (HL)	CB A6
RES	4, (IX+d)	DD CBdA6
RES	4, (IY+d)	FD CBdA6
RES	4, A	CB A7
RES	4, B	CB A0
RES	4, C	CB A1
RES	4, D	CB A2
RES	4, E	CB A3
RES	4, H	CB A4
RES	4, L	CB A5
RES	5, (HL)	CB AE
RES	5, (IX+d)	DD CBdAE
RES	5, (IY+d)	FD CBdAE
RES	5, A	CB AF
RES	5, B	CB A8
RES	5, C	CB A9
RES	5, D	CB AA
RES	5, E	CB AB
RES	5, H	CB AC
RES	5, L	CB AD
RES	6, (HL)	CB B6
RES	6, (IX+d)	DD CBdB6
RES	6, (IY+d)	FD CBdB6
RES	6, A	CB B7
RES	6, B	CB B0
RES	6, C	CB B1
RES	6, D	CB B2
RES	6, E	CB B3
RES	6, H	CB B4
RES	6, L	CB B5
RES	7, (HL)	CB BE
RES	7, (IX+d)	DD CBdBE
RES	7, (IY+d)	FD CBdBE
RES	7, A	CB BF
RES	7, B	CB B8
RES	7, C	CB B9

RES	7, D	CB BA
RES	7, E	CB BB
RES	7, H	CB BC
RES	7, L	CB BD
RET		C9
RET	C	D8
RET	M	F8
RET	NC	D0
RET	NZ	C0
RET	P	F0
RET	PE	E8
RET	PO	E0
RET	Z	C8
RETI		ED 4D
RETN		ED 45
RL	(HL)	CB 16
RL	(IX+d)	DD CBd16
RL	(IY+d)	FD CBd16
RL	A	CB 17
RL	B	CB 10
RL	C	CB 11
RL	D	CB 12
RL	E	CB 13
RL	H	CB 14
RL	L	CB 15
RLA		17
RLC	(HL)	CB 06
RLC	(IX+d)	DD CBd06
RLC	(IY+d)	FD CBd06
RLC	A	CB 07
RLC	B	CB 00
RLC	C	CB 01
RLC	D	CB 02
RLC	E	CB 03
RLC	H	CB 04
RLC	L	CB 05
RLCA		07
RLD		ED 6F
RR	(HL)	CB 1E
RR	(IX+d)	DD CBd1E
RR	(IY+d)	FD CBd1E
RR	A	CB 1F
RR	B	CB 18
RR	C	CB 19
RR	D	CB 1A
RR	E	CB 1B
RR	H	CB 1C
RR	L	CB 1D
RRA		1F
RRC	(HL)	CB OE
RRC	(IX+d)	DD CBd0E
RRC	(IY+d)	FD CBd0E
RRC	A	CB 0F

(nn) = Address of memory location
nn = Data (16 bit)
n = Data (8 bit)

d = signed displacement
d2 = d − 2

12.15 Instruction Set: Alphabetical Order (Continued)

RRC	B	CB 08
RRC	C	CB 09
RRC	D	CB 0A
RRC	E	CB 0B
RRC	H	CB 0C
RRC	L	CB 0D
RRCA		0F
RRD		ED 67
RST	0	C7
RST	08H	CF
RST	10H	D7
RST	18H	DF
RST	20H	E7
RST	28H	EF
RST	30H	F7
RST	38H	FF
SBC	A, (HL)	9E
SBC	A, (IX+d)	DD 9Ed
SBC	A, (IY+d)	FD 9Ed
SBC	A, A	9F
SBC	A, B	98
SBC	A, C	99
SBC	A, D	9A
SBC	A, E	9B
SBC	A, H	9C
SBC	A, L	9D
SBC	A, n	DE n
SBC	HL, BC	ED 42
SBC	HL, DE	ED 52
SBC	HL, HL	ED 62
SBC	HL, SP	ED 72
SCF		37
SET	0, (HL)	CB C6
SET	0, (IX+d)	DD CBdC6
SET	0, (IY+d)	FD CBdC6
SET	0, A	CB C7
SET	0, B	CB C0
SET	0, C	CB C1
SET	0, D	CB C2
SET	0, E	CB C3
SET	0, H	CB C4
SET	0, L	CB C5
SET	1, (HL)	CB CE
SET	1, (IX+d)	DD CBdCE
SET	1, (IY+d)	FD CBdCE
SET	1, A	CB CF
SET	1, B	CB C8
SET	1, C	CB C9
SET	1, D	CB CA
SET	1, E	CB CB
SET	1, H	CB CC
SET	1, L	CB CD
SET	2, (HL)	CB D6
SET	2, (IX+d)	DD CBdD6
SET	2, (IY+d)	FD CBdD6
SET	2, A	CB D7
SET	2, B	CB D0
SET	2, C	CB D1
SET	2, D	CB D2
SET	2, E	CB D3
SET	2, H	CB D4
SET	2, L	CB D5
SET	3, (HL)	CB DE
SET	3, (IX+d)	DD CBdDE
SET	3, (IY+d)	FD CBdDE
SET	3, A	CB DF
SET	3, B	CB D8
SET	3, C	CB D9
SET	3, D	CB DA
SET	3, E	CB DB
SET	3, H	CB DC
SET	3, L	CB DD
SET	4, (HL)	CB E6
SET	4, (IX+d)	DD CBdE6
SET	4, (IY+d)	FD CBdE6
SET	4, A	CB E7
SET	4, B	CB E0
SET	4, C	CB E1
SET	4, D	CB E2
SET	4, E	CB E3
SET	4, H	CB E4
SET	4, L	CB E5
SET	5, (HL)	CB EE
SET	5, (IX+d)	DD CBdEE
SET	5, (IY+d)	FD CBdEE
SET	5, A	CB EF
SET	5, B	CB E8
SET	5, C	CB E9
SET	5, D	CB EA
SET	5, E	CB EB
SET	5, H	CB EC
SET	5, L	CB ED
SET	6, (HL)	CB F6
SET	6, (IX+d)	DD CBdF6
SET	6, (IY+d)	FD CBdF6
SET	6, A	CB F7
SET	6, B	CB F0
SET	6, C	CB F1
SET	6, D	CB F2
SET	6, E	CB F3
SET	6, H	CB F4
SET	6, L	CB F5
SET	7, (HL)	CB FE
SET	7, (IX+d)	DD CBdFE
SET	7, (IY+d)	FD CBdFE
SET	7, A	CB FF

(nn) = Address of memory location

nn = Data (16 bit)

n = Data (8 bit)

d = displacement

d2 = d − 2

12.15 Instruction Set: Alphabetical Order (Continued)

SET	7, B	CB F8
SET	7, C	CB F9
SET	7, D	CB FA
SET	7, E	CB FB
SET	7, H	CB FC
SET	7, L	CB FD
SLA	(HL)	CB 26
SLA	(IX+d)	DD CBd26
SLA	(IY+d)	FD CBd26
SLA	A	CB 27
SLA	B	CB 20
SLA	C	CB 21
SLA	D	CB 22
SLA	E	CB 23
SLA	H	CB 24
SLA	L	CB 25
SRA	(HL)	CB 2E
SRA	(IX+d)	DD CBd2E
SRA	(IY+d)	FD CBd2E
SRA	A	CB 2F
SRA	B	CB 28
SRA	C	CB 29
SRA	D	CB 2A
SRA	E	CB 2B
SRA	H	CB 2C
SRA	L	CB 2D
SRL	(HL)	CB 3E
SRL	(IX+d)	DD CBd3E
SRL	(IY+d)	FD CBd3E
SRL	A	CB 3F
SRL	B	CB 38
SRL	C	CB 39
SRL	D	CB 3A
SRL	E	CB 3B
SRL	H	CB 3C
SRL	L	CB 3D
SUB	(HL)	96
SUB	(IX+d)	DD 96d
SUB	(IY+d)	FD 96d
SUB	A	97
SUB	B	90
SUB	C	91
SUB	D	92
SUB	E	93
SUB	H	94
SUB	L	95
SUB	n	D6 n
XOR	(HL)	AE
XOR	(IX+d)	DD AEd
XOR	(IY+d)	FD AEd
XOR	A	AF
XOR	B	A8
XOR	C	A9
XOR	D	AA
XOR	E	AB
XOR	H	AC
XOR	L	AD
XOR	n	EE n

12.16 Instruction Set: Numerical Order

Op Code	Mnemonic
00	NOP
01nn	LD BC,nn
02	LD (BC),A
03	INC BC
04	INC B
05	DEC B
06n	LD B,n
07	RLCA
08	EX AF,A'F'
09	ADD HL,BC
0A	LD A,(BC)
0B	DEC BC
0C	INC C
0D	DEC C
0En	LD C,n
0F	RRCA
10d2	DJNZ d2
11nn	LD DE,nn
12	LD (DE),A
13	INC DE
14	INC D
15	DEC D
16n	LD D,n
17	RLA
18d2	JR d2
19	ADD HL,DE
1A	LD A,(DE)
1B	DEC DE
1C	INC E
1D	DEC E
1En	LD E,n
1F	RRA
20d2	JR NZ,d2
21nn	LD HL,nn
22nn	LD (nn),HL
23	INC HL
24	INC H
25	DEC H
26n	LD H, n
27	DAA
28d2	JR Z,d2
29	ADD HL,HL
2Ann	LD HL,(nn)
2B	DEC HL
2C	INC L
2D	DEC L
2En	LD L,n
2F	CPL
30d2	JR NC,d2
31nn	LD SP,nn
32nn	LD (nn),A
33	INC SP
34	INC (HL)
35	DEC (HL)
36n	LD (HL),n
37	SCF
38	JR C,d2
39	ADD HL,SP
3Ann	LD A,(nn)
3B	DEC SP
3C	INC A
3D	DEC A
3En	LD A,n

(nn) = Address of memory location
nn = Data (16 bit)
n = Data (8 bit)

d = displacement
d2 = d − 2

12.16 Instruction Set: Numerical Order (Continued)

Op Code	Mnemonic
3F	CCF
40	LD B,B
41	LD B,C
42	LD B,D
43	LD B,E
44	LD B,H
45	LD B,L
46	LD B,(HL)
47	LD B,A
48	LD C,B
49	LD C,C
4A	LD C,D
4B	LD C,E
4C	LD C,H
4D	LD C,L
4E	LD C,(HL)
4F	LD C,A
50	LD D,B
51	LD D,C
52	LD D,D
53	LD D,E
54	LD D,H
55	LD D,L
56	LD D,(HL)
57	LD D,A
58	LD E,B
59	LD E,C
5A	LD E,D
5B	LD E,E
5C	LD E,H
5D	LD E,L
5E	LD E,(HL)
5F	LD E,A
60	LD H,B
61	LD H,C
62	LD H,D
63	LD H,E
64	LD H,H
65	LD H,L
66	LD H,(HL)
67	LD H,A
68	LD L,B
69	LD L,C
6A	LD L,D
6B	LD L,E
6C	LD L,H
6D	LD L,L
6E	LD L,(HL)
6F	LD L,A
70	LD (HL),B
71	LD (HL),C
72	LD (HL),D
73	LD (HL),E
74	LD (HL),H
75	LD (HL),L
76	HALT
77	LD (HL),A
78	LD A,B
79	LD A,C
7A	LD A,D
7B	LD A,E
7C	LD A,H
7D	LD A,L
7E	LD A,(HL)
7F	LD A,A
80	ADD A,B
81	ADD A,C
82	ADD A,D
83	ADD A,E
84	ADD A,H
85	ADD A,L
86	ADD A,(HL)
87	ADD A,A
88	ADC A,B
89	ADC A,C
8A	ADC A,D
8B	ADC A,E
8C	ADC A,H
8D	ADC A,L
8E	ADC A,(HL)
8F	ADC A,A
90	SUB B
91	SUB C
92	SUB D
93	SUB E
94	SUB H
95	SUB L
96	SUB (HL)
97	SUB A
98	SBC A,B
99	SBC A,C
9A	SBC A,D
9B	SBC A,E
9C	SBC A,H
9D	SBC A,L
9E	SBC A,(HL)
9F	SBC A,A
A0	AND B
A1	AND C
A2	AND D
A3	AND E
A4	AND H
A5	AND L
A6	AND (HL)
A7	AND A
A8	XOR B
A9	XOR C
AA	XOR D
AB	XOR E
AC	XOR H
AD	XOR L
AE	XOR (HL)
AF	XOR A
B0	OR B
B1	OR C
B2	OR D
B3	OR E
B4	OR H
B5	OR L
B6	OR (HL)
B7	OR A
B8	CP B
B9	CP C
BA	CP D
BB	CP E
BC	CP H
BD	CP L
BE	CP (HL)
BF	CP A
C0	RET NZ
C1	POP BC
C2nn	JP NZ,nn
C3nn	JP nn
C4nn	CALL NZ,nn
C5	PUSH BC
C6n	ADD A,n
C7	RST 0
C8	RET Z
C9	RET
CAnn	JP Z,nn
CB00	RLC B
CB01	RLC C
CB02	RLC D
CB03	RLC E
CB04	RLC H
CB05	RLC L
CB06	RLC (HL)
CB07	RLC A
CB08	RRC B
CB09	RRC C
CB0A	RRC D
CB0B	RRC E
CB0C	RRC H
CB0D	RRC L
CB0E	RRC (HL)
CB0F	RRC A
CB10	RL B
CB11	RL C
CB12	RL D

(nn) = Address of memory location
nn = Data (16 bit)
n = Data (8-bit)

d = displacement
d2 = d − 2

12.16 Instruction Set: Numerical Order (Continued)

Op Code	Mnemonic
CB13	RL E
CB14	RL H
CB15	RL L
CB16	RL (HL)
CB17	RL A
CB18	RR B
CB19	RR C
CB1A	RR D
CB1B	RR E
CB1C	RR H
CB1D	RR L
CB1E	RR (HL)
CB1F	RR A
CB20	SLA B
CB21	SLA C
CB22	SLA D
CB23	SLA E
CB24	SLA H
CB25	SLA L
CB26	SLA (HL)
CB27	SLA A
CB28	SRA B
CB29	SRA C
CB2A	SRA D
CB2B	SRA E
CB2C	SRA H
CB2D	SRA L
CB2E	SRA (HL)
CB2F	SRA A
CB38	SRL B
CB39	SRL C
CB3A	SRL D
CB3B	SRL E
CB3C	SRL H
CB3D	SRL L
CB3E	SRL (HL)
CB3F	SRL A
CB40	BIT 0,B
CB41	BIT 0,C
CB42	BIT 0,D
CB43	BIT 0,E
CB44	BIT 0,H
CB45	BIT 0,L
CB46	BIT 0,(HL)
CB47	BIT 0,A
CB48	BIT 1,B
CB49	BIT 1,C
CB4A	BIT 1,D
CB4B	BIT 1,E
CB4C	BIT 1,H
CB4D	BIT 1,L
CB4E	BIT 1,(HL)
CB4F	BIT 1,A
CB50	BIT 2,B
CB51	BIT 2,C
CB52	BIT 2,D
CB53	BIT 2,E
CB54	BIT 2,H
CB55	BIT 2,L
CB56	BIT 2,(HL)
CB57	BIT 2,A
CB58	BIT 3,B
CB59	BIT 3,C
CB5A	BIT 3,D
CB5B	BIT 3,E
CB5C	BIT 3,H
CB5D	BIT 3,L
CB5E	BIT 3,(HL)
CB5F	BIT 3,A
CB60	BIT 4,B
CB61	BIT 4,C
CB62	BIT 4,D
CB63	BIT 4,E
CB64	BIT 4,H
CB65	BIT 4,L
CB66	BIT 4,(HL)
CB67	BIT 4,A
CB68	BIT 5,B
CB69	BIT 5,C
CB6A	BIT 5,D
CB6B	BIT 5,E
CB6C	BIT 5,H
CB6D	BIT 5,L
CB6E	BIT 5,(HL)
CB6F	BIT 5,A
CB70	BIT 6,B
CB71	BIT 6,C
CB72	BIT 6,D
CB73	BIT 6,E
CB74	BIT 6,H
CB75	BIT 6,L
CB76	BIT 6,(HL)
CB77	BIT 6,A
CB78	BIT 7,B
CB79	BIT 7,C
CB7A	BIT 7,D
CB7B	BIT 7,E
CB7C	BIT 7,H
CB7D	BIT 7,L
CB7E	BIT 7,(HL)
CB7F	BIT 7,A
CB80	RES 0,B
CB81	RES 0,C
CB82	RES 0,D
CB83	RES 0,E
CB84	RES 0,H
CB85	RES 0,L
CB86	RES 0,(HL)
CB87	RES 0,A
CB88	RES 1,B
CB89	RES 1,C
CB8A	RES 1,D
CB8B	RES 1,E
CB8C	RES 1,H
CB8D	RES 1,L
CB8E	RES 1,(HL)
CB8F	RES 1,A
CB90	RES 2,B
CB91	RES 2,C
CB92	RES 2,D
CB93	RES 2,E
CB94	RES 2,H
CB95	RES 2,L
CB96	RES 2,(HL)
CB97	RES 2,A
CB98	RES 3,B
CB99	RES 3,C
CB9A	RES 3,D
CB9B	RES 3,E
CB9C	RES 3,H
CB9D	RES 3,L
CB9E	RES 3,(HL)
CB9F	RES 3,A
CBA0	RES 4,B
CBA1	RES 4,C
CBA2	RES 4,D
CBA3	RES 4,E
CBA4	RES 4,H
CBA5	RES 4,L
CBA6	RES 4,(HL)
CBA7	RES 4,A
CBA8	RES 5,B
CBA9	RES 5,C
CBAA	RES 5,D
CBAB	RES 5,E
CBAC	RES 5,H
CBAD	RES 5,L
CBAE	RES 5,(HL)
CBAF	RES 5,A
CBB0	RES 6,B
CBB1	RES 6,C
CBB2	RES 6,D
CBB3	RES 6,E
CBB4	RES 6,H
CBB5	RES 6,L
CBB6	RES 6,(HL)

(nn) = Address of memory location   d = displacement

nn = Data (16 bit)   d2 = d − 2

n = Data (8-bit)

12.16 Instruction Set: Numerical Order (Continued)

Op Code	Mnemonic
CBB7	RES 6,A
CBB8	RES 7,B
CBB9	RES 7,C
CBBA	RES 7,D
CBBB	RES 7,E
CBBC	RES 7,H
CBBD	RES 7,L
CBBE	RES 7,(HL)
CBBF	RES 7,A
CBC0	SET 0,B
CBC1	SET 0,C
CBC2	SET 0,D
CBC3	SET 0,E
CBC4	SET 0,H
CBC5	SET 0,L
CBC6	SET 0,(HL)
CBC7	SET 0,A
CBC8	SET 1,B
CBC9	SET 1,C
CBCA	SET 1,D
CBCB	SET 1,E
CBCC	SET 1,H
CBCD	SET 1,L
CBCE	SET 1,(HL)
CBCF	SET 1,A
CBD0	SET 2,B
CBD1	SET 2,C
CBD2	SET 2,D
CBD3	SET 2,E
CBD4	SET 2,H
CBD5	SET 2,L
CBD6	SET 2,(HL)
CBD7	SET 2,A
CBD8	SET 3,B
CBD9	SET 3,C
CBDA	SET 3,D
CBDB	SET 3,E
CBDC	SET 3,H
CBDD	SET 3,L
CBDE	SET 3,(HL)
CBDF	SET 3,A
CBE0	SET 4,B
CBE1	SET 4,C
CBE2	SET 4,D
CBE3	SET 4,E
CBE4	SET 4,H
CBE5	SET 4,L
CBE6	SET 4,(HL)
CBE7	SET 4,A
CBE8	SET 5,B
CBE9	SET 5,C
CBEA	SET 5,D
CBEB	SET 5,E
CBEC	SET 5,H
CBED	SET 5,L
CBEE	SET 5,(HL)
CBEF	SET 5,A
CBF0	SET 6,B
CBF1	SET 6,C
CBF2	SET 6,D
CBF3	SET 6,E
CBF4	SET 6,H
CBF5	SET 6,L
CBF6	SET 6,(HL)
CBF7	SET 6,A
CBF8	SET 7,B
CBF9	SET 7,C
CBFA	SET 7,D
CBFB	SET 7,E
CBFC	SET 7,H
CBFD	SET 7,L
CBFE	SET 7,(HL)
CBFF	SET 7,A
CCnn	CALL Z,nn
CDnn	CALL nn
CEn	ADC A,n
CF	RST 8
D0	RET NC
D1	POP DE
D2nn	JP NC,nn
D3n	OUT (n),A
D4nn	CALL NC,nn
D5	PUSH DE
D6n	SUB n
D7	RST 10H
D8	RET C
D9	EXX
DAnn	JP,C,nn
DBn	IN A,(n)
DCnn	CALL C,nn
DD09	ADD IX,BC
DD19	ADD IX,DE
DD21nn	LD IX,nn
DD22	LD (nn),IX
DD23	INC IX
DD29	ADD IX,IX
DD2Ann	LD IX,(nn)
DD2B	DEC IX
DD34d	INC (IX+d)
DD35d	DEC (IX+d)
DD36dn	LD (IX+d),n
DD39	ADD IX,SP
DD46d	LD B,(IX+d)
DD4Ed	LD C,(IX+d)
DD56d	LD D,(IX+d)
DD5Ed	LD E,(IX+d)
DD66d	LD H,(IX+d)
DD6Ed	LD L,(IX+d)
DD70d	LD (IX+d),B
DD71d	LD (IX+d),C
DD72d	LD (IX+d),D
DD73d	LD (IX+d),E
DD74d	LD (IX+d),H
DD75d	LD (IX+d),L
DD77d	LD (IX+d),A
DD7Ed	LD A,(IX+d)
DD86d	ADD A,(IX+d)
DD8Ed	ADC A,(IX+d)
DD96d	SUB (IX+d)
DD9Ed	SBC A,(IX+d)
DDA6d	AND (IX+d)
DDAEd	XOR (IX+d)
DDB6d	OR (IX+d)
DDBEd	CP (IX+d)
DDCBd06	RLC (IX+d)
DDCBd0E	RRC (IX+d)
DDCBd16	RL (IX+d)
DDCBd1E	RR (IX+d)
DDCBd26	SLA (IX+d)
DDCBd2E	SRA (IX+d)
DDCBd3E	SRL (IX+d)
DDCBd46	BIT 0,(IX+d)
DDCBd4E	BIT 1,(IX+d)
DDCBd56	BIT 2,(IX+d)
DDCBd5E	BIT 3,(IX+d)
DDCBd66	BIT 4,(IX+d)
DDCBd6E	BIT 5,(IX+d)
DDCBd76	BIT 6,(IX+d)
DDCBd7E	BIT 7,(IX+d)
DDCBd86	RES 0,(IX+d)
DDCBd8E	RES 1,(IX+d)
DDCBd96	RES 2,(IX+d)
DDCBd9E	RES 3,(IX+d)
DDCBdA6	RES 4,(IX+d)
DDCBdAE	RES 5,(IX+d)
DDCBdB6	RES 6,(IX+d)
DDCBdBE	RES 7,(IX+d)
DDCBdC6	SET 0,(IX+d)
DDCBdCE	SET 1,(IX+d)
DDCBdD6	SET 2,(IX+d)
DDCBdDE	SET 3,(IX+d)
DDCBdE6	SET 4,(IX+d)
DDCBdEE	SET 5,(IX+d)
DDCBdF6	SET 6,(IX+d)
DDCBdFE	SET 7,(IX+d)
DDE1	POP IX
DDE3	EX (SP),IX
DDE5	PUSH IX
DDE9	JP (IX)

(nn) = Address of memory location   d = displacement

nn = Data (16 bit)   d2 = d − 2

n = Data (8-bit)

12.16 Instruction Set: Numerical Order (Continued)

Op Code	Mnemonic
DDF9	LD SP,IX
DEn	SCB A,n
DF	RST 18H
E0	RET PO
E1	POP HL
E2nn	JP PO,nn
E3	EX (SP),HL
E4nn	CALL PO,nn
E5	PUSH HL
E6n	AND n
E7	RST 20H
E8	RET PE
E9	JP (HL)
EAnn	JP PE,nn
EB	EX DE,HL
ECnn	CALL PE,nn
ED40	IN B,(C)
ED41	OUT (C),B
ED42	SBC HL,BC
ED43nn	LD (nn),BC
ED44	NEG
ED45	RETN
ED46	IM 0
ED47	LD I,A
ED48	IN C,(C)
ED49	OUT (C),C
ED4A	ADC HL,BC
ED4Bnn	LD BC,(nn)
ED4D	RETI
ED50	IN D,(C)
ED51	OUT (C),D
ED52	SBC HL,DE
ED53nn	LD (nn),DE
ED56	IM 1
ED57	LD A,I
ED58	IN E,(C)
ED59	OUT (C), E
ED5A	ADC HL,DE
ED5Bnn	LD DE,(nn)
ED5E	IM 2
ED60	IN H,(C)
ED61	OUT (C),H
ED62	SBC HL,HL
ED67	RRD
ED68	IN L,(C)
ED69	OUT (C),L
ED6A	ADC HL,HL
ED6F	RLD
ED72	SBC HL,SP
ED73nn	LD (nn),SP
ED78	IN A,(C)
ED79	OUT (C),A
ED7A	ADC HL,SP
ED7Bnn	LD SP,(nn)
EDA0	LDI
EDA1	CPI
EDA2	INI
EDA3	OUTI
EDA8	LDD
EDA9	CPD
EDAA	IND
EDAB	OUTD
EDB0	LDIR
EDB1	CPIR
EDB2	INIR
EDB3	OTIR
EDB8	LDDR
EDB9	CPDR
EDBA	INDR
EDBB	OTDR
EEn	XOR n
EF	RST 28H
F0	RET P
F1	POP AF
F2nn	JP P,nn
F3	DI
F4nn	CALL P,nn
F5	PUSH AF
F6n	OR n
F7	RST 30H
F8	RET M
F9	LD SP,HL
FAnn	JP M,nn
FB	EI
FCnn	CALL M,nn
FD09	ADD IY,BC
FD19	ADD IY,DE
FD21nn	LD IY,nn
FD22nn	LD (nn),IY
FD23	INC IY
FD29	ADD IY,IY
FD2Ann	LD IY,(nn)
FD2B	DEC IY
FD34d	INC (IY+d)
FD35d	DEC (IY+d)
FD36dn	LD (IY+d),n
FD39	ADD IY,SP
FD46d	LD B,(IY+d)
FD4Ed	LD C,(IY+d)
FD56d	LD D,(IY+d)
FD5Ed	LD E,(IY+d)
FD66d	LD H,(IY+d)
FD6Ed	LD L,(IY+d)
FD70d	LD (IY+d),B
FD71d	LD (IY+d),C
FD72d	LD (IY+d),D
FD73d	LD (IY+d),E
FD74d	LD (IY+d),H
FD75d	LD (IY+d),L
FD77d	LD (IY+d),A
FD7Ed	LD A,(IY+d)
FD86d	ADD A,(IY+d)
FD8Ed	ADC A,(IY+d)
FD96d	SUB (IY+d)
FD9Ed	SBC A,(IY+d)
FDA6d	AND (IY+d)
FDAEd	XOR (IY+d)
FDB6d	OR (IY+d)
FDBEd	CP (IY+d)
FDE1	POP IY
FDE3	EX (SP), IY
FDE5	PUSH IY
FDE9	JP (IY)
FDF9	LD SP,IY
FDCBd06	RLC (IY+d)
FDCBd0E	RRC (IY+d)
FDCBd16	RL (IY+d)
FDCBd1E	RR (IY+d)
FDCBd26	SLA (IY+d)
FDCBd2E	SRA (IY+d)
FDCBd3E	SRL (IY+d)
FDCBd46	BIT 0,(IY+d)
FDCBd4E	BIT 1,(IY+d)
FDCBd56	BIT 2,(IY+d)
FDCBd5E	BIT 3,(IY+d)
FDCBd66	BIT 4,(IY+d)
FDCBd6E	BIT 5,(IY+d)
FDCBd76	BIT 6,(IY+d)
FDCBd7E	BIT 7,(IY+d)
FDCBd86	RES 0,(IY+d)
FDCBd8E	RES 1,(IY+d)
FDCBd96	RES 2,(IY+d)
FDCBd9E	RES 3,(IY+d)
FDCBdA6	RES 4,(IY+d)
FDCBdAE	RES 5,(IY+d)
FDCBdB6	RES 6,(IY+d)
FDCBdBE	RES 7,(IY+d)
FDCBdC6	SET 0,(IY+d)
FDCBdCE	SET 1,(IY+d)
FDCBdD6	SET 2,(IY+d)
FDCBdDE	SET 3,(IY+d)
FDCBdE6	SET 4,(IY+d)
FDCBdEE	SET 5,(IY+d)
FDCBdF6	SET 6,(IY+d)
FDCBdFE	SET 7,(IY+d)
FEn	CP n
FF	RST 38H

(nn) = Address of memory location

nn = Data (16 bit)

n = Data (8-bit)

d = displacement

d2 = d − 2

13.0 Data Acquisition System

A natural application for the NSC800 is one that requires remote operation. Since power consumption is low if the system consists of only CMOS components, the entire package can conceivably operate from only a battery power source. In the application described herein, the only source of power will be from a battery pack composed of a stacked array of NiCad batteries (see *Figure 20*).

The application is that of a remote data acquisition system. Extensive use is made of some of the other LSI CMOS components manufactured by National: notably the ADC0816 and MM58167. The ADC0816 is a 16-channel analog-to-digital converter which operates from a 5V source. The MM58167 is a microprocessor-compatible real-time clock (RTC). The schematic for this system is shown in *Figure 20.* All the necessary features of the system are contained in six integrated circuits: NSC800, NSC810A, NSC831, HN6136P, ADC0816, and MM58167. Some other small scale integration CMOS components are used for normal interface requirements. To reduce component count, linear selection techniques are used to generate chip selects for the NSC810A and NSC831. Included also is a current loop communication link to enable the remote system to transfer data collected to a host system.

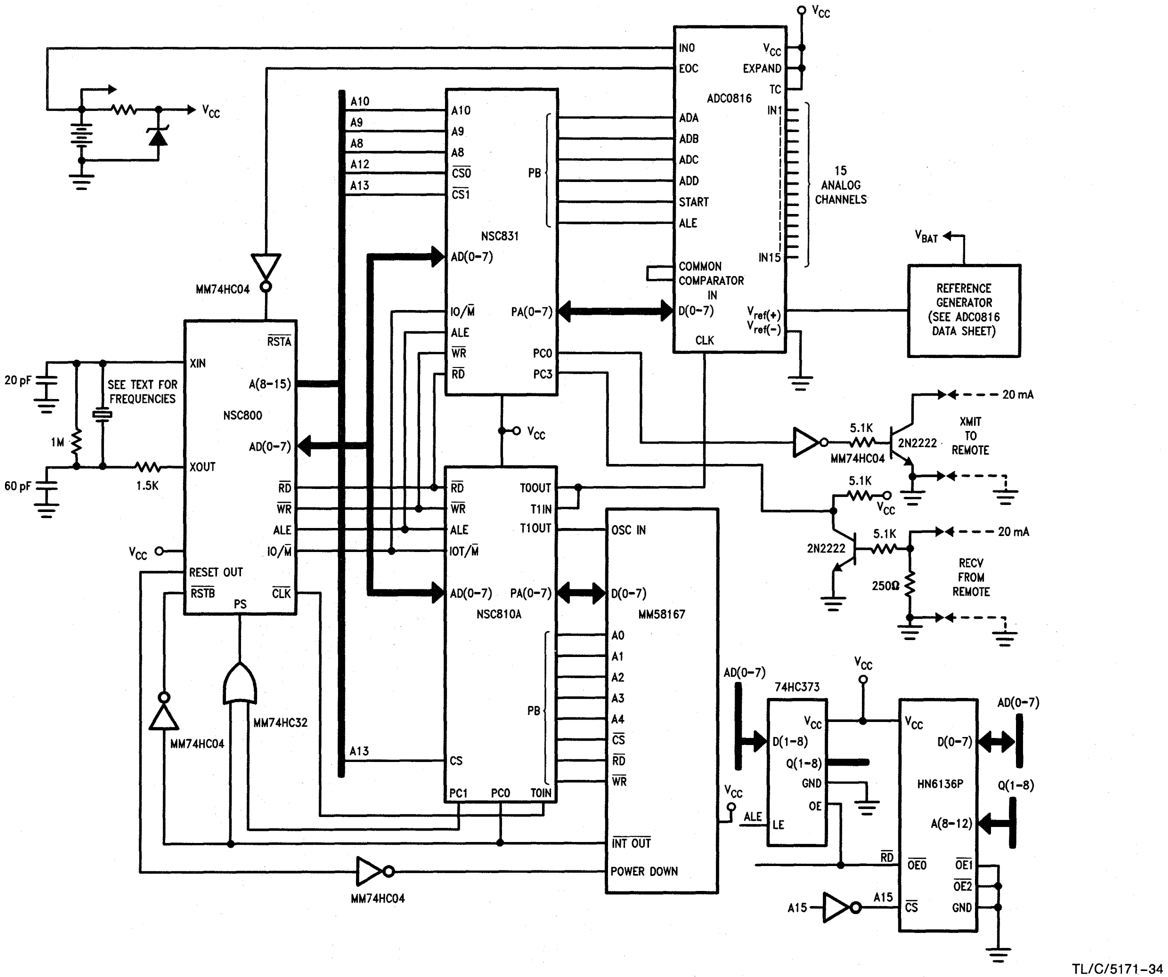
In order to keep component count low and maximize effectiveness, many of the features of the NSC800 family have been utilized. The RAM section of the NSC810A is used as a data buffer to store intermediate measurements and as scratch pad memory for calculations. Both timers contained in the NSC810A are used to produce the clocks required by the A/D converter and the RTC. The Power-Save feature of the NSC800 makes it possible to reduce system power consumption when it is not necessary to collect any data. One of the analog input channels of the A/D is connected to the battery pack to enable the CPU to monitor its own voltage supply and notify the host that a battery change is needed.

In operation, the NSC800 makes readings on various input conditions through the ADC0816. The type of devices connected to the A/D input depends on the nature of the remote environment. For example, the duties of the remote system might be to monitor temperature variations in a large building. In this case, the analog inputs would be connected to temperature transducers. If the system is situated in a process control environment, it might be monitoring fluid flow, temperatures, fluid levels, etc. In either case, operation would be necessary even if a power failure occurred, thus the need for battery operation or at least battery backup. At some fixed times or at some particular time durations, the system takes readings by selecting one of the analog input channels, commands the A/D to perform a conversion, reads the data, and then formats it for transmission; or, the system checks the readings against set points and transmits a warning if the set points are exceeded. With the addition of the RTC, the host need not command the remote system to take these readings each time it is necessary. The NSC800 could simply set up the RTC to interrupt it at a previously defined time and when the interrupt occurs, make the readings. The resultant values could be stored in the NSC810A for later correlation. In the example of temperature monitoring in a building, it might be desired to know the high and low temperatures for a 12-hour period. After compiling the information, the system could dump the data to the host over the communications link. Note from the schematic that the current for the communication link is supplied by the host to remove the constant current drain from the battery supply.

The required clocks for the two peripheral devices are generated by the two timers in the NSC810A. Through the use of various divisors, the master clock generated by the NSC800 is divided down to produce the clocks. Four examples are shown in the table following *Figure 20.*

All the crystal frequencies are standard frequencies. The various divisors listed are selected to produce, from the master clock frequency of the NSC800, an exact 32,768 Hz clock for the MM58167 and a clock within the operating range of the A/D converter.

The MM58167 is a programmable real-time clock that is microprocessor compatible. Its data format is BCD. It allows the system to program its interrupt register to produce an interrupt output either on a time of day match (which includes the day of the week, the date and month) and/or every month, week, day, hour, minute, second, or tenth of a second. With this capability added to the system, precise time of day measurements are possible without having the CPU do timekeeping. The interrupt output can be connected, through the use of one port bit of the NSC810A, to put the CPU in the power-save mode and reenable it at a preset time. The interrupt output is also connected to one of the hardware restart inputs ($\overline{\text{RSTB}}$) to enable time duration measurements. This power-down mode of operation would not be possible if the NSC800 had the duties of timekeep-



TL/C/5171-34

FIGURE 20. Remote Data Acquisition

13.0 Data Acquisition System (Continued)

ing. When in the power-save mode, the system power requirements are decreased by about 50%, thus extending battery life.

Communication with the peripheral devices (MM58167 and ADC0816) is accomplished through the I/O ports of the NSC810A and NSC831. The peripheral devices are not connected to the bus of the NSC800 as they are not directly compatible with a multiplexed bus structure. Therefore, additional components would be required to place them on the microprocessor bus. Writing data into the MM58167 is performed by first putting the desired data on Port A, followed by selecting the address of the internal register and applying the chip select through the use of Port B. A bit set and clear operation is performed to emulate a pulse on the bit of Port B connected to the $\overline{\text{WR}}$ input of the MM58167. For a read operation, the same sequence of operations is performed except that Port A is set for the input mode of operation and the $\overline{\text{RD}}$ line is pulsed. Similar techniques are used to read converted data from the A/D converter. When a conversion is desired, the CPU selects a channel and commands the ADC0816 to start a conversion. When the conversion is complete, the converter will produce an End-of-Conversion signal which is connected to the $\overline{\text{RSTA}}$ interrupt input of the NSC800.

When operating, the system shown consumes about 125 mw. When in the power-save mode, power consumption is decreased to about 70 mw. If, as is likely, the system is in the power-save mode most of the time, battery life can be quite long depending on the amp-hour rating of the batteries incorporated into the system. For example, if the battery pack is rated at 5 amp-hours, the system should be able to operate for about 400-500 hours before a battery charge or change is required.

As shown in the schematic (refer to *Figure 20*), analog input IN0 is connected to the battery source. In this way, the CPU can monitor its own power source and notify the host that it needs a battery replacement or charge. Since the battery source shown is a stacked array of 7 NiCads producing 8.4V, the converter input is connected in the middle so that it can take a reading on two or three of the cells. Since NiCad batteries have a relatively constant voltage output until very nearly discharged, the CPU can sense that the "knee" of the discharge curve has been reached and notify the host.

Typical Timer Output Frequencies

Crystal Frequency	CPU Clock Output	Timer 0 Output	Timer 1 Output
2.097152 MHz	1.048576 MHz	262.144 kHz divisor = 4	32.768 kHz divisor = 8
3.276800 MHz	1.638400 MHz	327.680 kHz divisor = 5	32.768 kHz divisor = 10
4.194304 MHz	2.097152 MHz	262.144 kHz divisor = 8	32.768 kHz divisor = 8
4.915200 MHz	2.457600 MHz	491.520 kHz divisor = 5	32.768 kHz divisor = 15

14.0 NSC800M/883B MIL-STD-833 Class C Screening

National Semiconductor offers the NSC800D and NSC800E with full class B screening per MIL-STD-883 for Military/Aerospace programs requiring high reliability. In addition, this screening is available for all of the key NSC800 peripheral devices.

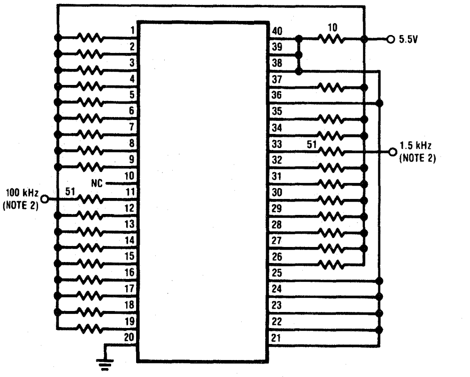
Electrical testing is performed in accordance with RESTS800X, which tests or guarantees all of the electrical performance characteristics of the NSC800 data sheet. A copy of the current revision of RETS800X is available upon request.

100% Screening Flow

Test	MIL-STD-883 Method/Condition	Requirement
Internal Visual	2010B	100%
Stabilization Bake	1008 C 24 Hrs. @ +150°C	100%
Temperature Cycling	1010 C 10 Cycles −65°C/+150°C	100%
Constant Acceleration	2001 E 30,000 G's, Y1 Axis	100%
Fine Leak	1014 A or B	100%
Gross Leak	1014C	100%
Burn-In	1015 160 Hrs. @ +125°C (using burn-in circuits shown below)	100%
Final Electrical	+25°C DC per RETS800X	100%
PDA	10% Max	
	+125°C AC and DC per RETS800X	100%
	−55°C AC and DC per RETS800X	100%
	+25°C AC per RETS800X	100%
QA Acceptance	5005	Sample Per
Quality Conformance		Method 5005
External Visual	2009	100%

15.0 Burn-In Circuits

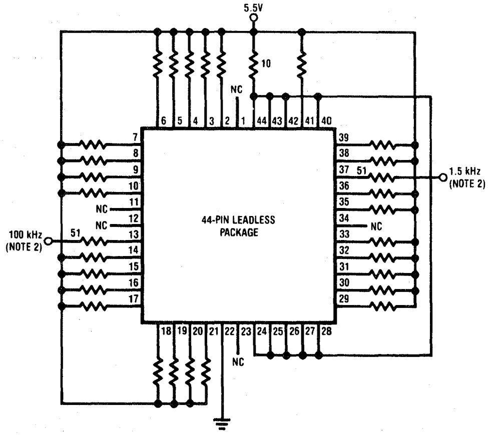
5240HR
NSC800D/883B (Dual-In-Line)



TL/C/5171-32

Top View

5241HR
NSC800E/883B (Leadless Chip Carrier)



TL/C/5171-33

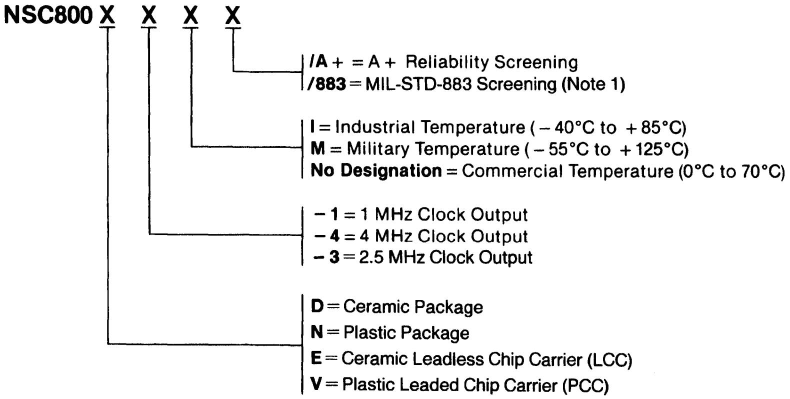
All resistors 2.7 kΩ unless marked otherwise.

Note 1: All resistors are ¼W ± 5% unless otherwise specified.

Note 2: All clocks 0V to 3V, 50% duty cycle, in phase with < 1 µs rise and fall time.

Note 3: Device to be cooled down under power after burn-in.

16.0 Ordering Information



NSC800 X X X X

/A+ = A+ Reliability Screening
/883 = MIL-STD-883 Screening (Note 1)

I = Industrial Temperature (−40°C to +85°C)
M = Military Temperature (−55°C to +125°C)
No Designation = Commercial Temperature (0°C to 70°C)

−1 = 1 MHz Clock Output
−4 = 4 MHz Clock Output
−3 = 2.5 MHz Clock Output

D = Ceramic Package
N = Plastic Package
E = Ceramic Leadless Chip Carrier (LCC)
V = Plastic Leaded Chip Carrier (PCC)

TL/C/5171–35

Note 1: Do not specify a temperature option; all parts are screened to military temperature.

17.0 Reliability Information

Gate Count 2750

Transistor Count 11,000

Section 2

Peripherals

2

Section 2 Contents

National Semiconductor Corporation

NSC810A RAM-I/O-Timer

General Description

The NSC810A, the luxury model of our NSC800™ peripheral line, sports triple ported I/O, dual 16-bit timers and a 1024-bit static storage area. The three ports can be combined for a total of 22 general purpose I/O lines. In addition, port A has several strobed mode operations. Note the single instruction I/O bit operations for quick and efficient data handling from the ports. The timers feature 6 modes of operation and prescalers for those complicated timing tasks. The NSC810A comes in two models: the Dual-In-Line (DIP) and the surface mount chip carrier (LCC). It also comes in three exciting temperature ranges (Commercial, Industrial, and Military) and two reliability flows (extended burn-in and military class B in accordance with Method 5004 of MIL-STD-883). This is brought to you through the microCMOS silicon gate technology of National Semiconductor.

Features

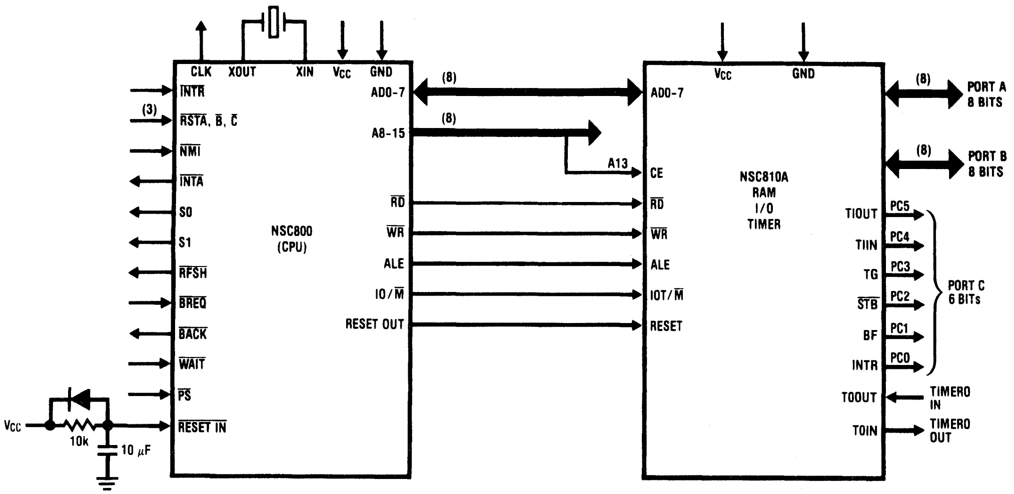
- Three programmable I/O ports
- Dual 16-bit programmable counter/timers
- 2.4V–6.0V power supply
- Very low power consumption
- Fully static operation
- Single-instruction I/O bit operations
- Timer operation—DC to 5 MHz
- Bus compatible with NSC800™ family
- Speed: compatible with NSC800

  NSC810A-4 → NSC800-4 @ 4.0 MHz

  NSC810A-3 → NSC800 @ 2.5 MHz

  NSC810A-1 → NSC800-1 @ 1.0 MHz

NSC810A Connection Diagram



TL/C/5517-1

2

Table of Contents

1.0 Absolute Maximum Ratings

(Note 1)

If Military/Aerospace specified devices are required, contact the National Semiconductor Sales Office/ Distributors for availability and specifications.

Storage Temperature Range	−65°C to +150°C
Voltage at Any Pin with Respect to Ground	−0.3V to V_{CC} + 0.3V
V_{CC}	7V
Power Dissipation	1W
Lead Temperature (Soldering, 10 seconds)	300°C

2.0 Operating Conditions

V_{CC} = 5V ± 10%

NSC810A-1 →	0°C to +70°C −40°C to +85°C
NSC810A-3 →	0°C to +70°C −40°C to +85°C −55°C to +125°C
NSC810A-4 →	0°C to +70°C −40°C to +85°C −55°C to +125°C

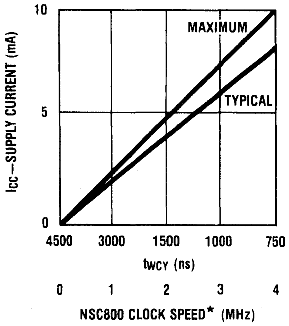
3.0 DC Electrical Characteristics V_{CC}=5V ±10%, GND=0V, unless otherwise specified.

Symbol	Parameter	Conditions	Min	Typ	Max	Units
V_{IH}	Logical 1 Input Voltage		0.8 V_{CC}		V_{CC}	V
V_{IL}	Logical 0 Input Voltage		0		0.2 V_{CC}	V
V_{OH}	Logical 1 Output Voltage	I_{OH} = −1.0 mA I_{OUT} = −10 μA	2.4 V_{CC}−0.5			V V
V_{OL}	Logical 0 Output Voltage	I_{OL} = 2 mA I_{OUT} = 10 μA	0 0		0.4 0.1	V V
I_{IL}	Input Leakage Current	0 ≤ V_{IN} ≤ V_{CC}	−10.0		10.0	μA
I_{OL}	Output Leakage Current	0 ≤ V_{IN} ≤ V_{CC}	−10.0		10.0	μA
I_{CC}	Active Supply Current	I_{OUT} = 0, Timer = Mode 1, T0IN = T1IN = 2.5 Mhz, t_{WCY} = 750 ns, T_A = 25°C		8	10	mA
I_Q	Quiescent Current	No Input Switching, T_A = 25°C, RESET = 0, IO/$\overline{M}$ = 1, $\overline{RD}$ = 1, $\overline{WR}$ = 1, ALE = 1, V_{IN} = V_{CC}, t_{IN} = 0 Hz, t_{OUT} = 0		10	100	μA
C_{IN}	Input Capacitance			4	7	pF
C_{OUT}	Output Capacitance			6	10	pF
V_{CC}	Power Supply Voltage	(Note 2)	2.4	5	6	V
V_{DRV}	Data Retention Voltage		1.8			V

Note 1: Absolute maximum ratings are those values beyond which the safety of the device cannot be guaranteed. Continuous operation at these limits is not intended; operation should be limited to those conditions specified under DC Electrical Characteristics.

Note 2: Operation at lower power supply voltages will reduce the maximum operating speed. Operation at voltages other than 5V ±10% is guaranteed by design, not tested.

I_{CC} vs Speed



TL/C/5517-2

*When NSC810A is used with NSC800

4.0 AC Electrical Characteristics V_{CC} = 5V ±10%, GND = 0V

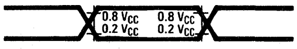
Symbol	Parameter	Conditions	NSC810A-1		NSC810A-3		NSC810-4		Units
			Min	Max	Min	Max	Min	Max	
t_{ACC}	Access Time from ALE	C_L = 150 pF		1000		400		300	ns
t_{AH}	AD0-7, CE, IOT/$\overline{M}$ Hold Time		100		60		30		ns
t_{ALE}	ALE Strobe Width (High)		200		125		100		ns
t_{ARW}	ALE to $\overline{RD}$ or $\overline{WR}$ Strobe		150		120		75		ns
t_{AS}	AD0-7, CE, IOT/$\overline{M}$ Set-Up Time		100		45		25		ns
t_{DH}	Data Hold Time		150		90		40		ns
t_{DO}	Port Data Output Valid			350		310		300	ns
t_{DS}	Data Set-Up Time		100		80		50		ns
t_{PE}	Peripheral Bus Enable			320		200		200	ns
t_{PH}	Peripheral Data Hold Time		150		125		100		ns
t_{PS}	Peripheral Data Set-Up Time		100		75		50		ns
t_{PZ}	Peripheral Bus Disable (TRI-STATE®)			150		150		150	ns
t_{RB}	$\overline{RD}$ to BF Invalid			300		300		300	ns
t_{RD}	Read Strobe Width		400		320		185		ns
t_{RDD}	Data Bus Disable		0	100	0	100	0	75	ns
t_{RI}	$\overline{RD}$ to $\overline{INTR}$ Output			320		320		300	ns
t_{RWA}	$\overline{RD}$ or $\overline{WR}$ to Next ALE		125		100		75		ns
t_{SB}	$\overline{STB}$ to BF Valid			300		300		300	ns
t_{SH}	Peripheral Data Hold with Respect to $\overline{STB}$		150		125		100		ns
t_{SI}	$\overline{STB}$ to $\overline{INTR}$ Output			300		300		300	ns
t_{SS}	Peripheral Data Set-Up with Respect to $\overline{STB}$		100		75		50		ns
t_{SW}	$\overline{STB}$ Width		400		320		220		ns
t_{WB}	$\overline{WR}$ to BF Output			340		340		300	ns
t_{WI}	$\overline{WR}$ to $\overline{INTR}$ Output			320		320		300	ns
t_{WR}	$\overline{WR}$ Strobe Width		400		320		220		ns
t_{WCY}	Width of Machine Cycle		3000		1200		750		ns

Note: Test conditions: t_{WCY} = 3000 ns for NSC810A-1, 1200 ns for NSC810A-3, 750 ns for NSC810A-4

5.0 Timer AC Electrical Characteristics

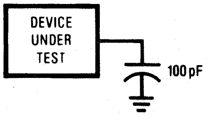
Symbol	Parameter	Conditions	Min	Typ	Max	Units
F_C	Clock Frequency		DC		2.5	MHz
F_{CP}	Clock Frequency	Prescale Selected	DC		5.0	MHz
t_{CW}	Clock Pulse Width		150			ns
t_{CWP}	Clock Pulse Width	Prescale Selected	75			ns
t_{GS}	Gate Set-Up Time	With Respect to Negative Clock Edge	100			ns
t_{GH}	Gate Hold Time	With Respect to Negative Clock Edge	250			ns
t_{CO}	Clock to Output Delay	C_L = 100 pF			350	ns

AC TESTING INPUT/OUTPUT WAVEFORM



TL/C/5517-3

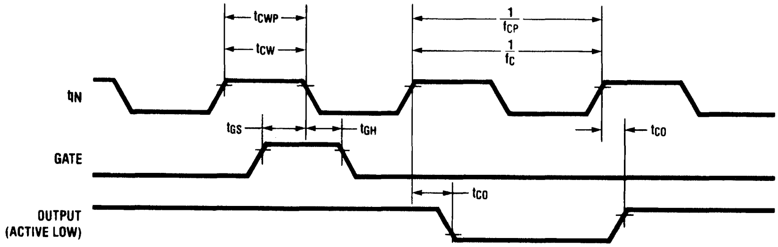
AC TESTING LOAD CIRCUIT



TL/C/5517-4

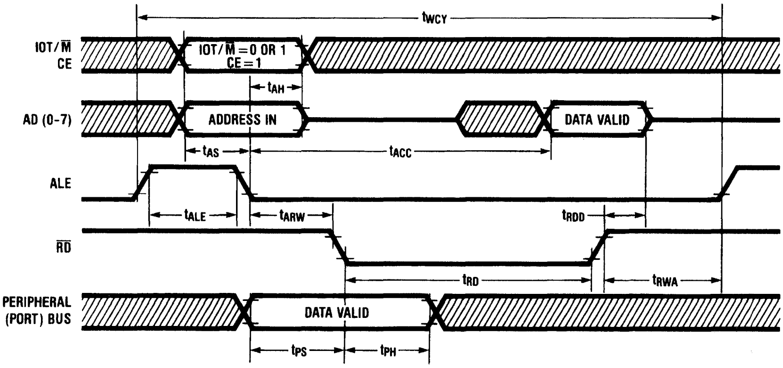
6.0 Timing Waveforms

Timer Waveforms



TL/C/5517-5

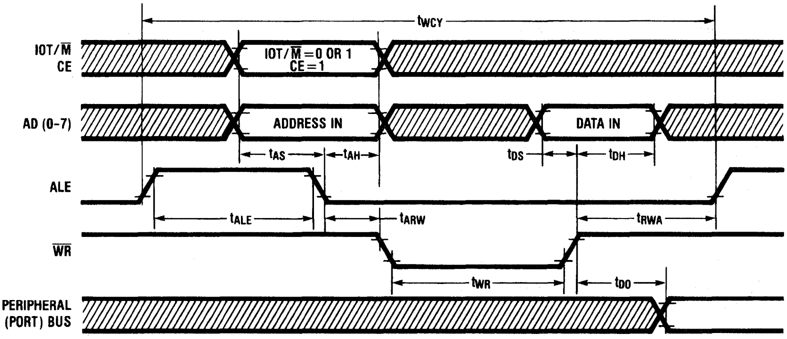
Read Cycle (Read from RAM, Port or Timer)



TL/C/5517-6

Note: Diagonal lines indicate interval of invalid data.

Write Cycle (Write to RAM, Port or Timer)

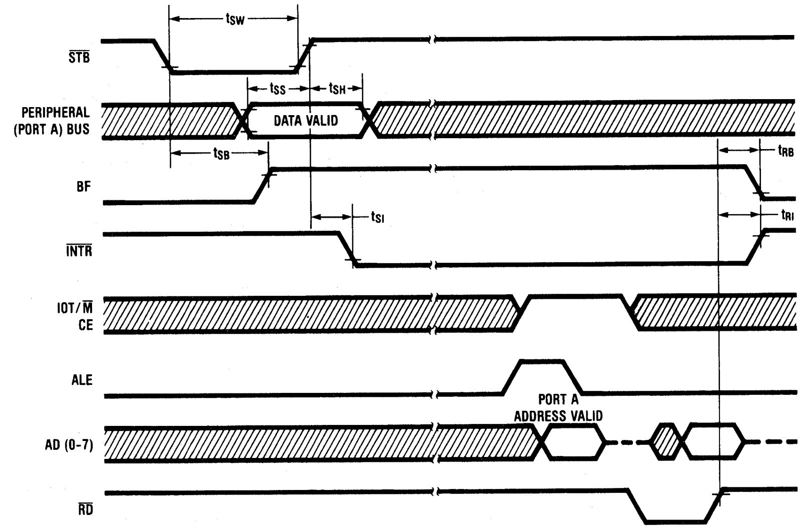


TL/C/5517-7

Note: Diagonal lines indicate interval of invalid data.

6.0 Timing Waveforms (Continued)

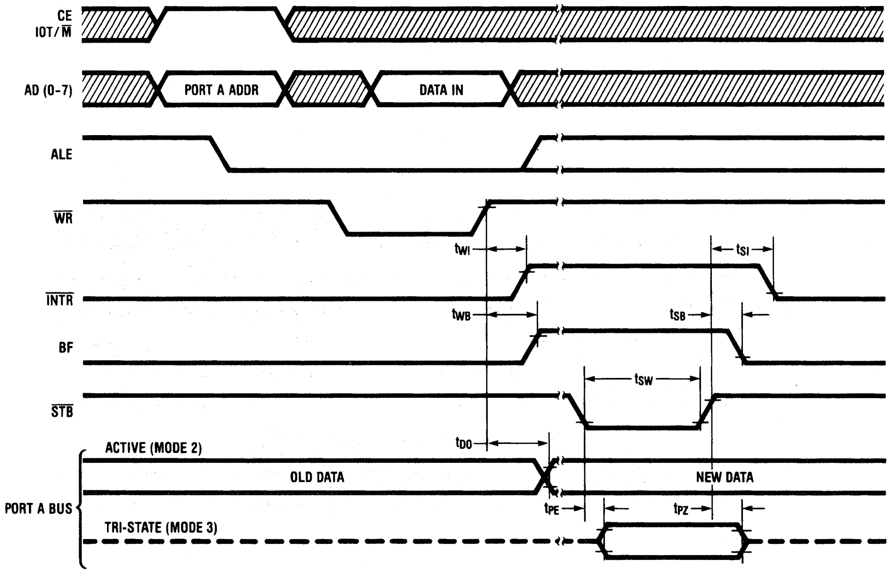
Strobed Mode Input



TL/C/5517-8

Note: Diagonal lines indicate interval of invalid data.

Strobed Mode Output



TL/C/5517-9

Note: Diagonal lines indicate interval of invalid data.

7.0 Pin Descriptions

The function and mnemonic for the NSC810A signals are described below:

7.1 INPUT SIGNALS

Reset (RESET): RESET is an active-high input that resets all registers to 0 (low). The RAM contents remain unaltered.

Input/Output Timer or RAM Select (IOT/$\overline{\text{M}}$): IOT/$\overline{\text{M}}$ is an I/O memory select input line. A logic 1 (high) input selects the I/O-timer portion of the chip; a logic 0 (low) input selects the RAM portion of the chip. IOT/$\overline{\text{M}}$ is latched at the falling edge of ALE.

Chip Enable (CE): CE is an active-high input that allows access to the NSC810A. CE is latched at the falling edge of ALE.

Read ($\overline{\text{RD}}$): The $\overline{\text{RD}}$ is an active-low input that enables a read operation of the RAM or I/O-timer location.

Write ($\overline{\text{WR}}$): The $\overline{\text{WR}}$ is an active-low input that enables a write operation to RAM or I/O-timer locations.

Address Latch Enable (ALE): The falling edge of the ALE input latches AD0–AD7, CE and IOT/$\overline{\text{M}}$ inputs to form the address for RAM, I/O or timer.

Timer 0 Input (T0IN): T0IN is the clock input for timer 0.

7.2 OUTPUT SIGNALS

Timer 0 Output (T0OUT): T0OUT is the programmable output of timer 0. After reset, T0OUT is set high.

7.3 POWER SUPPLY SIGNALS

Positive DC Voltage (V_{CC}): V_{CC} is the 5V supply pin.

Ground (GND): Ground reference pin.

7.4 INPUT/OUTPUT SIGNALS

Address/Data Bus (AD0–AD7): The multiplexed bidirectional address/data bus; AD0–AD7 pins, are in the high impedance state when the NSC810A is not selected. AD0–AD7 will latch address inputs at the falling edge of ALE. The address will designate a location in RAM, I/O or timer. $\overline{\text{WR}}$ input enables 8-bit data to be written into the addressed location. $\overline{\text{RD}}$ input enables 8-bit data to be read from the addressed location. The $\overline{\text{RD}}$ or $\overline{\text{WR}}$ inputs occur while ALE is low.

Port A, 0–7 (PA0–PA7): Port A is an 8-bit basic mode input/output port, also capable of strobed mode I/O utilizing three control signals from port C. Strobed mode of operation on port A has three different modes; strobed input, strobed output with active peripheral bus, strobed output with TRI-STATE peripheral bus.

Port B, 0–7 (PB0–PB7): Port B is an 8-bit basic mode input/output port.

Port C, 0–5 (PC0–PC5): Port C is a 6-bit basic mode I/O port. Each pin has a programmable second function, as follows:

PC0/$\overline{\text{INTR}}$: $\overline{\text{INTR}}$ is an active-low, strobed mode interrupt request to the Central Processor Unit (CPU).

PC1/BF: BF is an active-high, strobed mode, buffer full output to peripheral devices.

PC2/$\overline{\text{STB}}$: $\overline{\text{STB}}$ is an active-low, strobed mode input from peripheral devices.

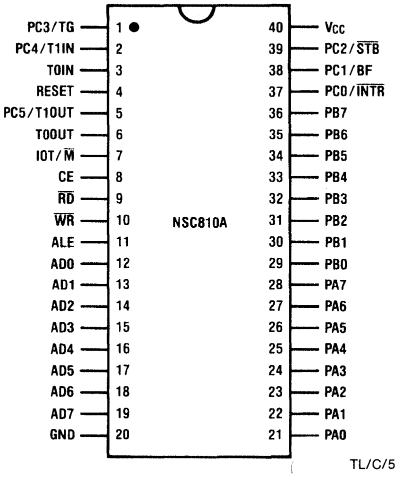
PC3/TG: TG is the timer gating signal.

PC4/T1IN: T1IN is the clock input for timer 1.

PC5/T1OUT: T1OUT is the programmable output of timer 1.

8.0 Connection Diagrams

Dual-In-Line Package

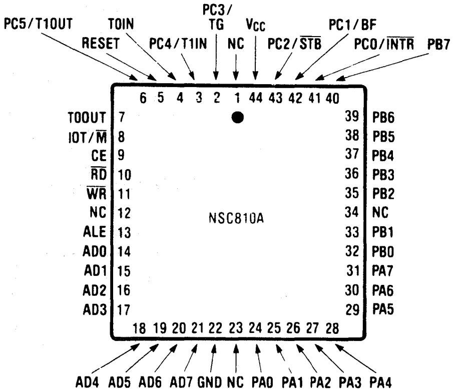


TL/C/5517-10

Top View

Order Number NSC810AD or NSC810AN
See NS Package Number D40C or N40A

Chip Carrier



TL/C/5517-11

Top View

NC = no connect

Order Number NSC810AE or NSC810AV
See NS Package Number E44B or V44A

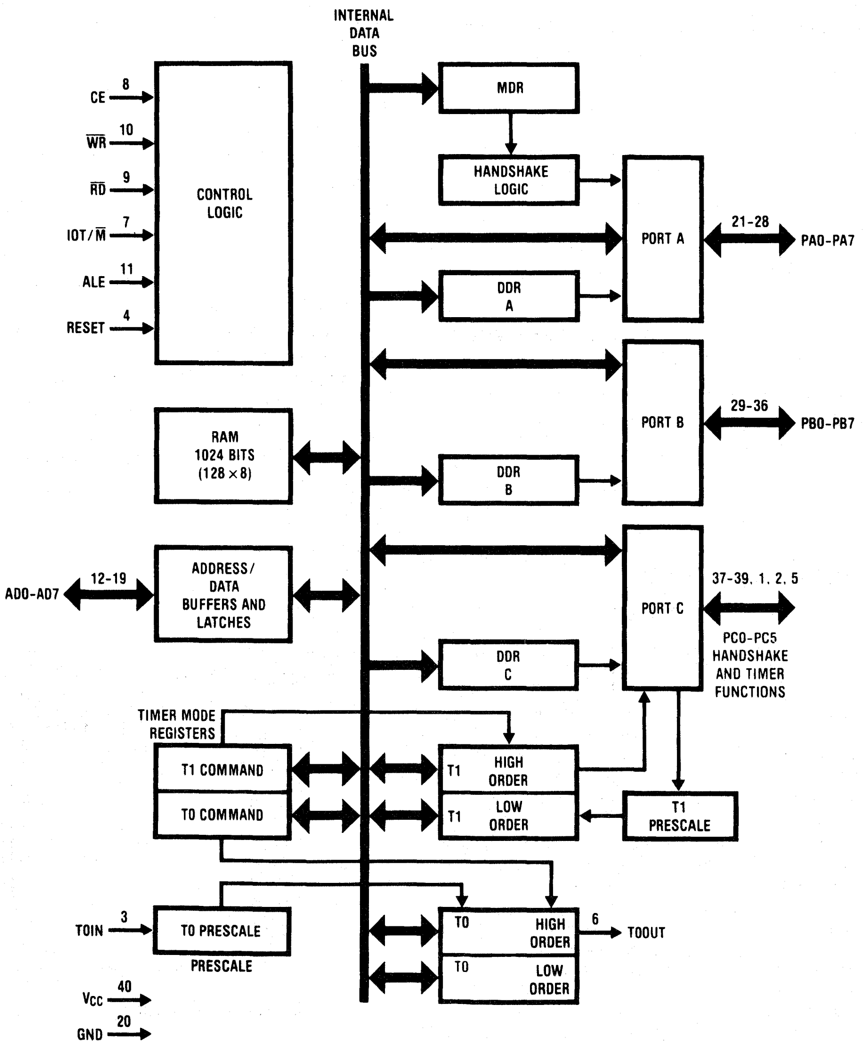
9.0 Functional Description

Figure 1 is a detailed block diagram of the NSC810A. The functional description that follows describes the RAM, I/O and TIMER sections.

9.1 RANDOM ACCESS MEMORY (RAM)

The memory portion of the RAM-I/O-timer is accessed by a 7-bit address input to pins AD0 through AD6. The IOT/$\overline{\text{M}}$ input must be low (RAM select) and the CE input must be high at the falling edge of ALE to address the RAM. Address bit AD7 is a "don't care" for RAM addressing. Timing for RAM read and write operations is shown in the timing diagrams. The RAM is 128 x 8.

9.2 DETAILED BLOCK DIAGRAM



TL/C/5517-12

FIGURE 1

9.0 Functional Description (Continued)

9.3 I/O PORTS

The three I/O ports, labeled A, B, and C, can be programmed to be almost any combination of Input and Output bits. Ports A and B are configured as 8 bits wide, while port C is 6 bits. There are four different modes of operation for the ports. Three of the modes are for timed transfer of data between the peripheral and the NSC810A, this is called strobed I/O. The fourth mode is for direct transfer without handshaking with the peripheral.

The NSC810A can be programmed to operate in four different modes. One of these modes (Basic I/O) allows direct transfer of I/O data without any handshaking between the NSC810A and the peripheral. The other three modes (Strobed I/O) provide for timed transfers of I/O data with handshaking between the NSC810A and the peripheral.

The determination of the mode, data direction and data is done by five registers which are, handily, under program control. The Mode Definition Register (MDR), oddly enough, determines which mode the device will operate in, while the Data Direction Register (DDR) establishes the direction of the data transfer. The Data register contains the data that is being sent or has been received. The other two registers (bit-set, bit-clear) allow the individual bits in the data register to be set or cleared without affecting the other bits. Each port has its own set of these registers, except the MDR which affects ports A and C only.

In the strobed I/O modes, port C bits 0, 1 and 2 function as INTR (for the processor), BF, and STB respectively.

9.3.1 Registers

As can be seen in Table I, all the registers affecting I/O transfer are grouped at the lower address locations, this allows quicker handling and more maneuverability in tight data transfers. Also note in Table I that the NSC810A uses 23 I/O addresses out of a block of 26. The upper three bits of the address are determined by the chip enable address.

- **Mode Definition Register (MDR)**

As noted above this register defines the operating mode for ports A and C (port B is always in the basic I/O mode). The upper 3 bits of port C will also be in the basic I/O mode even when the lower 3 bits are being used for handshaking.

The four modes are as follows:

Mode 0—Basic I/O (Input or Output)

Mode 1—Strobed Mode Input

Mode 2—Strobed Mode Output (Active Peripheral Bus)

Mode 3—Strobed Mode Output (TRI-STATE Peripheral Bus)

The address assignment of the MDR is xxx00111 as shown in Table I. Table II specifies the data that must be loaded into the MDR to select the mode.

- **Data Direction Registers (DDR)**

Each port has a DDR that determines whether an individual port bit will be an input or an output. This can be considered the traffic light for the transfer of data between the CPU and the peripheral. Each port bit has a corresponding bit in this register. If the DDR bit is set (1) the port bit is an output; if it is cleared (0) the port bit is an input. The DDR bits cannot be written to individually. The register as a whole must be set to be consistent with all desired port bit directions.

TABLE I. I/O and Timer Address Designations

8-Bit Address Field Bits								Designation I/O Port, Timer, etc.	R (Read) W (Write)
7	6	5	4	3	2	1	0		
x	x	x	0	0	0	0	0	Port A (Data)	R/W
x	x	x	0	0	0	0	1	Port B (Data)	R/W
x	x	x	0	0	0	1	0	Port C (Data)	R/W
x	x	x	0	0	0	1	1	Not Used	**
x	x	x	0	0	1	0	0	DDR - Port A	W
x	x	x	0	0	1	0	1	DDR - Port B	W
x	x	x	0	0	1	1	0	DDR - Port C	W
x	x	x	0	0	1	1	1	Mode Definition Reg.	W
x	x	x	0	1	0	0	0	Port A - Bit-Clear	W
x	x	x	0	1	0	0	1	Port B - Bit-Clear	W
x	x	x	0	1	0	1	0	Port C - Bit-Clear	W
x	x	x	0	1	0	1	1	Not Used	**
x	x	x	0	1	1	0	0	Port A - Bit-Set	W
x	x	x	0	1	1	0	1	Port B - Bit-Set	W
x	x	x	0	1	1	1	0	Port C - Bit-Set	W
x	x	x	0	1	1	1	1	Not Used	**
x	x	x	1	0	0	0	0	Timer 0 (LB)	*
x	x	x	1	0	0	0	1	Timer 0 (HB)	*
x	x	x	1	0	0	1	0	Timer 1 (LB)	*
x	x	x	1	0	0	1	1	Timer 1 (HB)	*
x	x	x	1	0	1	0	0	STOP Timer 0	W
x	x	x	1	0	1	0	1	START Timer 0	W
x	x	x	1	0	1	1	0	STOP Timer 1	W
x	x	x	1	0	1	1	1	START Timer 1	W
x	x	x	1	1	0	0	0	Timer 0 Mode	R/W
x	x	x	1	1	0	0	1	Timer 1 Mode	R/W
x	x	x	1	1	0	1	0	Not Used	**
x	x	x	1	1	0	1	1	Not Used	**
x	x	x	1	1	1	0	0	Not Used	**
x	x	x	1	1	1	0	1	Not Used	**
x	x	x	1	1	1	1	0	Not Used	**
x	x	x	1	1	1	1	1	Not Used	**

x = don't care

LB = low-order byte

HB = high-order byte

* A write accesses the modulus register, a read the read buffer.

** A read from an unused location reads invalid data, a write does not affect any operation of NSC810A.

TABLE II. Mode Definition Register Bit Assignments

Mode	Bit							
	7	6	5	4	3	2	1	0
0	x	x	x	x	x	x	x	0
1	x	x	x	x	x	x	0	1
2	x	x	x	x	x	0	1	1
3	x	x	x	x	x	1	1	1

9.0 Functional Description (Continued)

Any write or read to the port bits contradicting the direction established by the DDR will not affect the port bits output or input. However, a write to a port bit, defined as an input, will modify the output latch and a read to a port bit, defined as an output, will read this output latch. See *Figure 2.*

• Data Registers

These registers contain the actual data being transferred between the CPU and the peripheral. In Basic I/O, data presented by the peripheral (read cycle) will be latched on the falling edge of RD. Data presented by the CPU (write cycle) will be valid after the rising edge of WR (see AC characteristics for exact timing).

During Strobed I/O, data presented by the peripheral must be valid on the rising edge of STB. Data received by the peripheral will be valid on the rising edge of STB. Data latched by the port on the rising edge of STB will be preserved until the next CPU read or STB signal.

• Bit Set-Clear Registers

The I/O features of the RAM-I/O-timer allow modification of a single bit or several bits of a port with the Bit-Set and Bit-Clear commands. The address selected indicates whether a Bit-Set or Clear will take place. The incoming data on the address/data bus is latched at the trailing edge of the WR strobe and is treated as a mask. All bits containing 1s will cause the indicated operation to be performed on the corresponding port bit. All bits of the mask with 0s cause the corresponding port bits to remain unchanged. Three sample operations are shown in Table III using port B as an example.

TABLE III. Bit-Set and Clear Examples

Operation Port B	Set B7	Clear B2 and B0	Set B4, B3 and B1
Address	xxx01101	xxx01001	xxx01101
Data	10000000	00000101	00011010
Port Pins			
Prior State	00001111	10001111	10001010
Next State	10001111	10001010	10011010

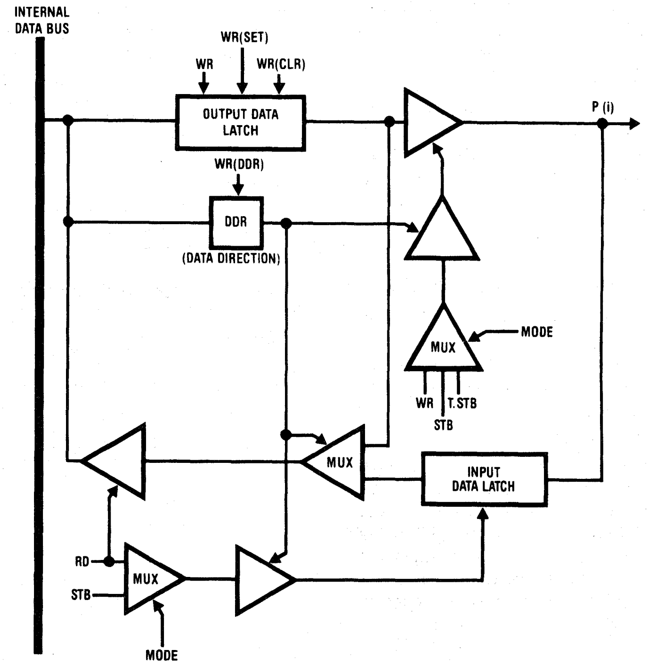
9.3.2 Modes

Two data transfer modes are implemented: Basic I/O and Strobed I/O. Strobed I/O can be further subdivided into three categories: Strobed Input, Strobed Output (active peripheral bus) and Strobed Output (TRI-STATE peripheral bus). The following descriptions detail the functions of these categories.

• Basic I/O

Basic I/O mode uses the RD and WR CPU bus signals to latch data at the peripheral bus. This mode is the permanent mode of operation for ports B and C. Port A is in this mode if the MDR is set to mode 0. Read and write byte operations and bit operations can be done in Basic I/O. Timing for these modes is shown in the AC Characteristics Table and described with the data register definitions.

When the NSC810A is reset, all registers are cleared to zero. This results in the basic mode of operation being selected, all port bits are made inputs and the output latch for each port bit is cleared to zero. The NSC810A, at this point, can read data from any peripheral port without further setup. If outputs are desired, the CPU merely has to program the appropriate DDR and then send data to the data ports.



TL/C/5517-13

FIGURE 2

9.0 Functional Description (Continued)

• Strobed I/O

Strobed I/O Mode uses the $\overline{\text{STB}}$, BF and $\overline{\text{INTR}}$ signals to latch the data and indicate that new data is available for transfer. Port A is used for the transfer of data when in any of the Strobed modes. Port B can still be used for Basic I/O and the lower 3-bits of port C are now the three handshake signals for Strobed I/O. Timing for this mode is shown in the AC Characteristic Tables.

Initializing the NSC810A for Strobed I/O Mode is done by loading the data shown in Table IV Into the specified register. The registers should be loaded in the order (left to right) that they appear in Table IV.

TABLE IV. Mode Definition Register Configurations

Mode	MDR	DDR Port A	DDR Port C	Port C Output Latch
Basic I/O	xxxxxxx0	Port bit directions are determined by the bits of each port's DDR		
Strobed Input	xxxxxx01	00000000	xxx011	xxx1xx
Strobed Output (Active)	xxxxx011	11111111	xxx011	xxx1xx
Strobed Output (TRI-STATE)	xxxxx111	11111111	xxx011	xxx1xx

• Strobed Input (Mode 1)

During strobed input operations, an external device can load data into port A with the $\overline{\text{STB}}$ signal. Data is input to the PA0–7 input latches on the leading (negative) edge of $\overline{\text{STB}}$, causing BF to go high (true). On the trailing (positive) edge of $\overline{\text{STB}}$ the data is latched and the interrupt signal, $\overline{\text{INTR}}$, becomes valid indicating to the CPU that new data is available. $\overline{\text{INTR}}$ becomes valid only if the interrupt is enabled, that is the output data latch for PC2 is set to 1.

When the CPU reads port A, address x'00, the trailing edge of the $\overline{\text{RD}}$ strobe causes BF and $\overline{\text{INTR}}$ to become inactive, indicating that the strobed input cycle has been completed.

• Strobed Output—Active (Mode 2)

During strobed output operations, an external device can read data from port A using the $\overline{\text{STB}}$ signal. Data is initially loaded into port A by the CPU writing to I/O address x'00. On the trailing edge of $\overline{\text{WR}}$, $\overline{\text{INTR}}$ is set inactive and BF becomes valid indicating new data is available for the external device. When the external device is ready to accept the data in port A it pulses the $\overline{\text{STB}}$ signal. The rising edge of $\overline{\text{STB}}$ resets BF and activates the $\overline{\text{INTR}}$ signal. $\overline{\text{INTR}}$ becomes valid only if the interrupt is enabled, that is the output latch for PC2 is set to 1. $\overline{\text{INTR}}$ in this mode indicates a condition that requires CPU intervention (the output of the next byte of data).

• Strobed Output—TRI-STATE (Mode 3)

The Strobed Output TRI-STATE Mode and the Strobed Output active (peripheral) bus mode function in a similar manner with one exception. The exception is that the data signals on PA0-7 assume the high impedance state at all times except when accessed by the $\overline{\text{STB}}$ signal. Strobed Mode 3 is identical to Strobed Mode 2, except as indicated above.

Example Mode 1 (Strobed Input):

Action Taken	$\overline{\text{INTR}}$	BF	Results of Action
INITIALIZATION			
Reset NSC810A	H	L	Basic input mode all ports.
Load 01'H into MDR	H	L	Strobed input mode entered; no byte loads to port C after this step; bit-set and clear commands to $\overline{\text{INTR}}$ and BF no longer work.
Load 00'H into DDR A	H	L	Sets data direction register for port A to input; data from port A peripheral bus is available to the CPU if the $\overline{\text{STB}}$ signal is used, other handshake signals aren't initialized, yet.
Load 03'H into DDR C	H	L	Sets data direction register of port C; buffer full signal works after this step and it is unaffected by the bit-set and clear registers.
Load 04'H into Port C Bit-Set Register	H	L	Sets output latch (PC2) to enable $\overline{\text{INTR}}$; $\overline{\text{INTR}}$ will latch active whenever $\overline{\text{STB}}$ goes low; $\overline{\text{INTR}}$ can be disabled by a bit-clear to PC2.*
OPERATION			
$\overline{\text{STB}}$ pulses low	L	H	Data on peripheral bus is latched into port A; $\overline{\text{INTR}}$ is cleared by a CPU read of port A or a bit-clear of $\overline{\text{STB}}$.
CPU reads Port A	H	L	CPU gets data from port A; $\overline{\text{INTR}}$ is cleared; peripheral is signalled to send next byte via an inactive BF signal. Repeat last two steps until EOT at which time CPU sends bit-clear to the output latch (PC2).

* Port C can be read by the CPU at anytime, allowing polled operation instead of interrupt driven operation.

9.0 Functional Description (Continued)

Example Mode 2 (Strobed Output—active peripheral bus):

Action Taken	$\overline{\text{INTR}}$	BF	Results of Action
INITIALIZE			
Reset NSC810A	H	L	basic input mode all ports.
Load 03'H into MDR	H	L	strobed output mode entered; no byte loads to port C after this step; bit-set and clear commands to $\overline{\text{INTR}}$ and BF no longer work.
Load FF'H into DDR A	H	L	Sets data direction register for port A to output; data from port A is available to the peripheral if the $\overline{\text{STB}}$ signal is used other handshake signals aren't initialized, yet.
Load 03'H into DDR C	H	L	Sets data direction register of port C; buffer full signal works after this step and it is unaffected by the bit-set and clear registers
Load 04'H into Port C Bit-Set Register	L	L	Sets output latch (PC2) to enable $\overline{\text{INTR}}$; active $\overline{\text{INTR}}$ indicates that CPU should send data; $\overline{\text{INTR}}$ becomes inactive whenever the CPU loads port A; $\overline{\text{INTR}}$ can be disabled by a bit-clear to $\overline{\text{STB}}$.*
OPERATION			
CPU writes to Port A	H	H	Data on CPU bus is latched into port A; $\overline{\text{INTR}}$ is set by the CPU write to port A; active BF indicates to peripheral that
$\overline{\text{STB}}$ pulses low	L	L	data is valid; Peripheral gets data from port A; $\overline{\text{INTR}}$ is reset active; The active $\overline{\text{INTR}}$ signals the CPU to send the next byte. Repeat last two steps until EOT at which time CPU sends bit-clear to the output latch (PC2).

*Port C can be read by the CPU at any time, allowing polled operation instead of interrupt driven operation.

In addition to its timing function, $\overline{\text{STB}}$ enables port A outputs to active logic levels. This Mode 3 operation allows other data sources, in addition to the NSC810A, to access the peripheral bus.

- **Handshaking Signals**

In the Strobed mode of operation, the lower 3-bits of port C transmit/receive the handshake signals (PC0 = $\overline{\text{INTR}}$, PC1 = BF, PC2 = $\overline{\text{STB}}$).

$\overline{\text{INTR}}$ (Strobe Mode Interrupt) is an active-low interrupt from the NSC810A to the CPU. In strobed input mode, the CPU reads the valid data at port A to clear the interrupt. In strobed output mode, the CPU clears the interrupt by writing data to port A.

The $\overline{\text{INTR}}$ output can be enabled or disabled, thus giving it the ability to control strobed data transfer. It is enabled or disabled, respectively, by setting or clearing bit 2 of the port C output data latch ($\overline{\text{STB}}$).

PC2 is always an input during strobed mode of operation, its output data latch is not needed. Therefore, during strobed mode of operation it is internally gated with the interrupt signal to generate the $\overline{\text{INTR}}$ output. Reset clears this bit to zero, so it must be set to one to enable the $\overline{\text{INTR}}$ pin for strobed operation.

Once the strobed mode of operation is programmed, the only way to change the output data latch of PC2 is by using the Bit-Set and Clear registers. The port C byte write command will not alter the output data latch of PC2 during the strobed mode of operation.

$\overline{\text{STB}}$ (Strobe) is an active low input from the peripheral device, signalling a data transfer. The NSC810A latches data on the rising edge of $\overline{\text{STB}}$ if the port bit is an input and the peripheral should latch data on the rising edge of $\overline{\text{STB}}$ if the port bit is an output.

BF (Buffer Full) is a high active output from the NSC810A. For input port bits, it indicates that new data has been received from the peripheral. For output port bits, it indicates that new data is available for the peripheral.

Note: In either input or output mode the BF may be cleared by rewriting the MDR.

9.4 TIMERS

The NSC810A has two timers. These are independently programmable, 16-bit binary down-counters. Full count is reached at n + 1, where n is the count loaded into the modulus registers. Timer outputs provide six distinct modes of operation and allow the CPU to check the present count at anytime. Each timer has an independent clock input and output. Start and stop words from the CPU can individually start and stop the timers in any of the modes. A common gate signal can start and stop both timers in three of the six modes. Timer 0 has three possible input clock prescalers ÷1, ÷2 and ÷64. Timer 1 has two possible input clock prescalers ÷1 and ÷2.

Primary components of one timer are shown in *Figure 3*. The timer mode register is a read/write register providing

9.0 Functional Description (Continued)

the primary characterization of the timer output. The start/stop logic and prescaler block divides the clock input by the prescale factor, passing the output (INTCLK) to the binary down-counter. This block also gates the clock input signal (TIN) with the timer gate signal (TG). The timer block loads the modulus from the modulus register and uses (INTCLK) to count to zero. It loads the current count into the read buffer block where the CPU can access it at anytime. This timer block also indicates to the output control logic when the modulus is loaded (or reloaded) and when the count reaches 0. The output control logic block drives the output pins according to the timer mode register and the timer block. The output of the timer block (*Figure 3*) (terminal count) is related to the input TIN by:

$$\text{terminal count} = \frac{\text{TIN}}{p[2(m + 1)]}$$

where:

TIN = the input frequency

p = the programmed prescale

m = the modulus

This relationship can be seen directly (TOUT) in Mode 5 (square wave) as it is not masked by the subsequent output logic.

9.4.1 Registers

There are five control registers for each timer. These are shown in the second group of Table I. They determine all timer functions and outputs.

• Modulus Registers and Read Buffer

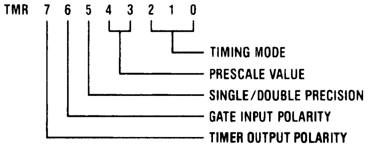
There are two modulus registers per timer (low byte, high byte). These are write only registers, and the two 8-bit values loaded by the CPU are combined into a 16-bit modulus for the timer's down counter.

When the CPU reads from the modulus register addresses, it actually accesses the read buffers. These contain the low and high byte of the decremented modulus. This count is constantly updated by the timer block on the falling edge of INTCLK and can be read without stopping the timers (see single/double precision).

• Timer Mode Register

The timer mode register determines the operating configuration and the active input and output signal levels. Each timer has its own timer mode register, allowing independent operation.

The timer mode register (TMR) may be written or read at any time; however, to assure accurate timing it is important to modify the mode only when the timer is stopped (see Timer Programming). The timer mode is selected from one of six modes by TMR bits 0, 1, and 2 (see Table V). Bits 3 and 4 select the prescale value if the prescaler is to be used. Bits 5, 6 and 7 select the modulus width (8- or 16-bits), gate input polarity, and timer output polarity (active-high or low), respectively. The bit functions of the TMR are illustrated in *Figure 4*.

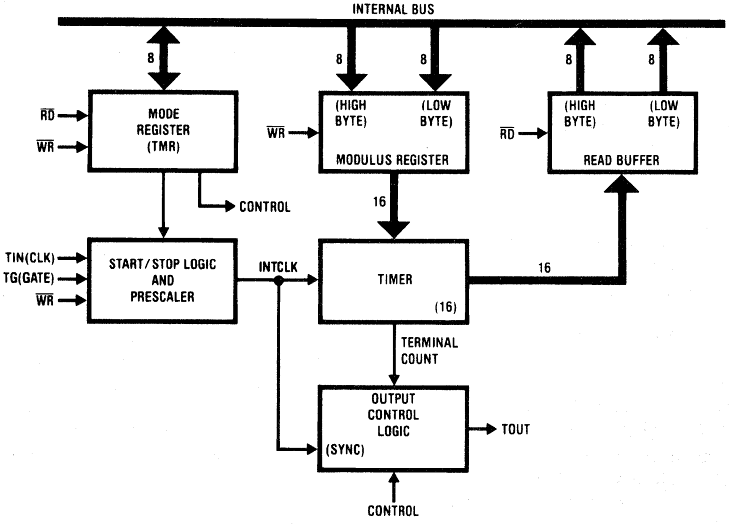


TL/C/5517-15

FIGURE 4. Timer Mode Register

TABLE V. Mode Selection

Bit	2	1	0	–	Timer Function
	0	0	0	–	Timer Stopped and Reset
	0	0	1	–	Event Counter
	0	1	0	–	Event Timer (Stopwatch)
	0	1	1	–	Event Timer (Resetting)
	1	0	0	–	One Shot
	1	0	1	–	Square Wave
	1	1	0	–	Pulse Generator
	1	1	1	–	Timer Stopped and Reset



TL/C/5517-14

FIGURE 3. Timer Internal Block Diagram (One of Two Timers)

9.0 Functional Description (Continued)

— Timer Prescaler

There is a prescale function associated with each timer. It serves as an additional divisor to lengthen the counts for each timer circuit. The value of the divisor is fixed and selectable in each TMR, as shown below.

	Bits		
TMR0	**4**	**3**	**Prescale**
	0	0	÷1
	0	1	÷2
	1	1	÷64

The ÷64 is not available on timer 1; TMR1 bit 4 is a "don't care."

	Bits		
TMR1	**4**	**3**	**Prescale**
	x	0	÷1
	x	1	÷2

The timer prescale divides the input clock (TIN) and provides the output (INTCLK) to the drive the timer block (*Figure 3*).

— Single/Double Precision

Bit 5 of the TMR determines whether a single or double byte can be accurately read from the read buffer. This option does not affect the use of the modulus registers by the timer block (i.e., the modulus used is always a double byte regardless of the precision mode selected).

The read buffer keeps track of the count and is constantly being updated by the timer block. In order to allow the CPU to read the read buffer, the NSC810A must discontinue updates to this buffer during the read. The precision bit determines whether one or two bytes in the read buffer will be frozen during the read process. In double precision mode, the NSC810A freezes high and low bytes in the read buffer for two consecutive read cycles. In the single precision mode, the NSC810A freezes the read buffer for only one read cycle. Read accesses should be done as follows.

When the TMR bit 5 is:

0— (double byte) read or write the low byte first, then the high byte to maintain proper read/write communications.

1— (single byte) In this mode either the high or low byte of the count can be read at any given instant but not both bytes consecutively. Always write the low byte first, then the high byte to load the modulus.

The following example illustrates this point. If the read buffer had a value of 0200 when the low byte was read and the down-counter decremented to 01FF before the high byte was read, then in the double precision mode the CPU would have read 00 and 02, respectively. In the single precision mode the CPU would have read 00 and 01.

NOTE: In the double precision mode, the high byte should be read immediately after the low byte. Do not access any other registers or unused address locations between the reads.

— Gate Input Polarity

In modes 2, 3 and 4, the TG input is the common hardware control for starting and stopping the timers.

The polarity of the gate input may be selected by the contents of bit 6 of the TMR. If bit 6 equals 0, the gate signal will be active-high or positive edge for mode 4; if bit 6 equals 1, the gate polarity will be active-low or negative edge for mode 4. Modes 2 and 3 are level sensitive. Mode 4 is edge sensitive.

— Timer Output Polarity

Like the gating function, the polarity of the output signal is programmable via bit 7 of the TMR. A zero will cause an active-low output; a one will generate an active-high output.

The output for T1 is multiplexed with port C, bit 5. (Similarly T1IN is multiplexed with port C, bit 4.) When any timer mode other than 0 or 7 is specified for T1, or when mode 2, mode 3, or mode 4 is specified for T0, the three port C pins, bit 3, bit 4, and bit 5, become TG, T1IN and T1OUT, respectively.

• Start and Stop Registers

This is the software start and stop for the timers. There is one start and one stop register for each timer. Writing any data to the start register of a timer starts that timer or transfers start and stop control to TG (in the gated modes 2, 3 and 4). Writing any data to the stop register stops the timer and removes start and stop control from TG (in the gated modes 2, 3 and 4). Restarting the timers causes the modulus to be reloaded for all gated timer modes (2, 3 and 4).

During software restarts of the timers (write to the STOP register and then to the START register) the modulus will be reloaded only if the internal clock signal (INTCLK) is in the high level or makes at least one transition to the high level between the time that the STOP and START registers are written. If INTCLK doesn't meet one of these criteria then the modulus will not be reloaded and the timer will continue to count down from where it was stopped.*

Since it is difficult, if not impossible, to know the level of INTCLK in non-gated modes the recommended practice for restart operation is to reload the modulus after stopping the timer using the 4 step programming procedure in the Timer Programming section of this datasheet. In gated modes INTCLK always stops high.

***NOTE:** INTCLK is coupled via the prescaler to TIN and reacts to the TIN clock input regardless of whether the timer is started or stopped.

— Start/Stop Timing

Figure 5 shows the relationships between the $\overline{\text{WR}}$ signal (start register), TIN and INTCLK for both the non-gated and gated modes. The TG signal is only sampled during the positive half of the TIN cycle. This means that when the gated modes are used the internal clock (INTCLK) is never stopped in the low state. Hence, when TG goes active high INTCLK is restarted on the next high-to-low transition of TIN. When TG goes inactive low INTCLK will stop as soon as TIN is high.

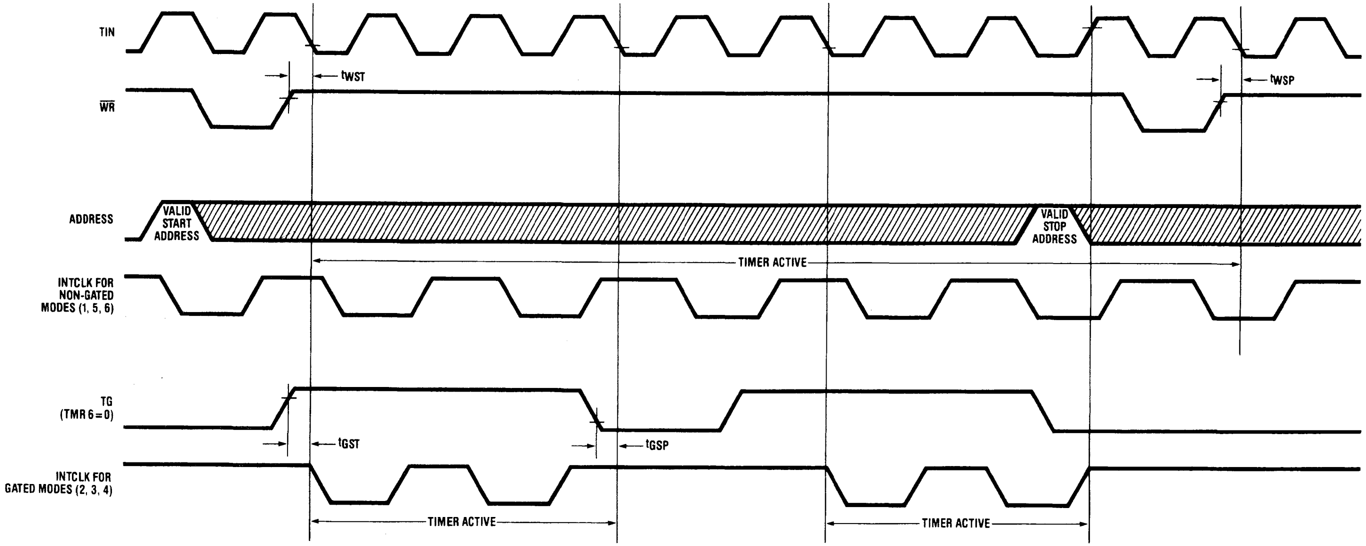
9.4.2 Timer Pins

TIN, TOUT, and TG

Timer 0 has dedicated pins for its clock, T0IN, and its output, T0OUT. Timer 1 must borrow its input and output pins from port C. This is accomplished by writing to the TMR for timer 1. If mode 1, 2, 3, 4, 5 or 6 is specified in TMR1, the pins from port C (PC3, PC4 and PC5) are automatically made available to the timer(s) for gating (TG), T1IN and T1OUT, respectively. These pins are also taken from port C any time timer 0 is in mode 2, 3, 4, so that it has a TG pin. In order to change pins PC3, PC4 and PC5 back to their original configuration as Basic I/O, the timer mode registers must be reset by selecting mode 0 or 7.

TG (PC3), the timer gate, is used for hardware control to start/stop (or trigger) the timers. The timer gate may be used individually by either timer or simultaneously by both timers.

For modes 2 and 3, the timer starts on the gate-active transition assuming the start address was previously written. If



TL/C/5517-16

FIGURE 5. Start/Stop Timing

Note: Diagonal lines indicate interval of invalid data.

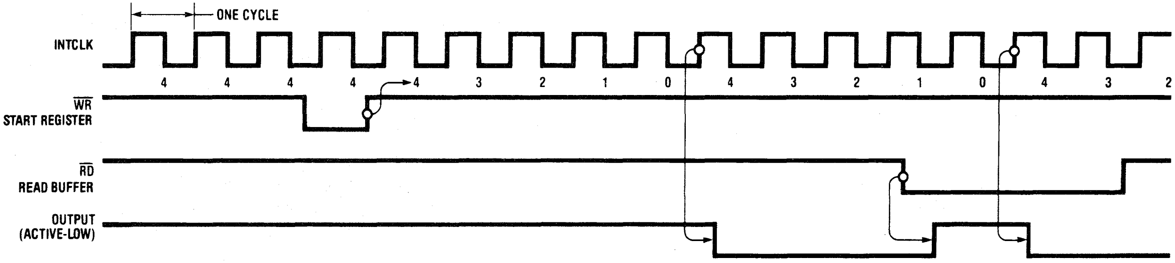
For mode 4 (one shot), only start-timing applies.

t_{WST}—$\overline{WR}$ set-up for starting timer 150 ns.

t_{WSP}—$\overline{WR}$ set-up for stopping timer 150 ns.

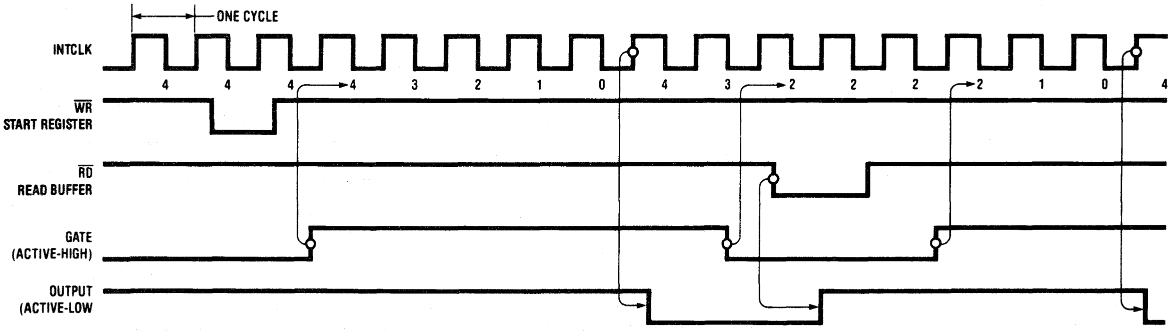
t_{GST}—TG (gate) set-up for starting timer 100 ns.

t_{GSP}—TG (gate) set-up for stopping timer 100 ns.



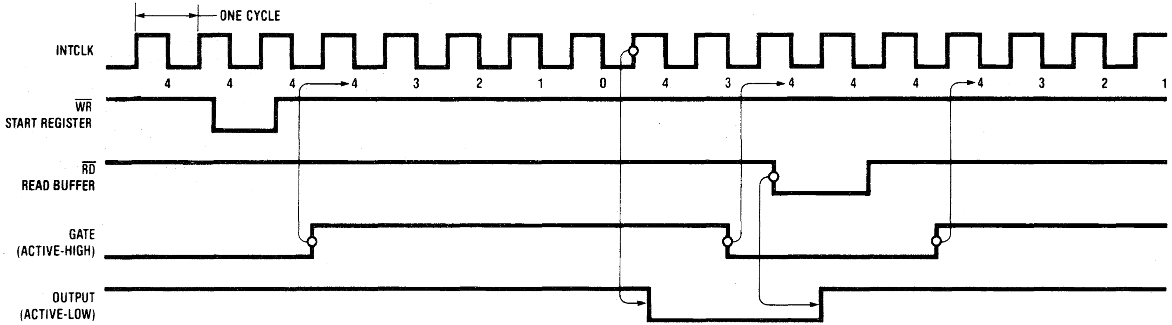
TL/C/5517-17

FIGURE 6a. Event Counter Mode (Mode 1)



TL/C/5517-18

FIGURE 6b. Accumulative Timer (Mode 2)



TL/C/5517-19

FIGURE 6c. Restartable Timing

9.0 Functional Description (Continued)

TABLE VI. Timer Programming Selection Example

Mode Register Bit (TMR)								Timer Output Polarity Active L/H	Timer Gate Polarity Active L/H	Mode Description Single/Double Precision S/D	Prescale Value	Timing Mode	Port C DDR 543210					
7	6	5	4	3	2	1	0						5	4	3	2	1	0
TIMER 0																		
x	x	x	x	x	0	0	0	x	x	x	x	0	x	x	x	x	x	x
0	x	0	0	0	0	0	1	L	x	D	÷1	1	x	x	x	x	x	x
1	x	0	1	1	1	1	0	H	x	D	÷64	6	x	x	x	x	x	x
1	0	0	0	1	1	0	0	H	H	D	÷2	4	1	0	0	x	x	x
0	1	1	0	0	0	1	0	L	L	S	÷1	2	1	0	0	x	x	x
TIMER 1																		
x	x	x	x	x	1	1	1	x	x	x	x	7	x	x	x	x	x	x
0	x	0	x	0	0	0	1	L	x	D	÷1	1	1	0	0	x	x	x
1	0	1	x	1	1	0	1	H	H	S	÷2	5	1	0	0	x	x	x
0	1	0	x	0	0	1	1	L	L	D	÷1	3	1	0	0	x	x	x

the timer gate makes an active transition prior to a write to the start register's address, the trailing edge of the $\overline{WR}$ strobe starts the timer. However, for mode 4 the timer always waits for an active gate edge following a write to the start address before it begins counting.

The DDR for port C must be programmed with the correct I/O direction for TG, T1IN and T1OUT of timer 1. See Table VI for programming examples.

9.4.3 Timer Modes

The low-order three bits (bits 0, 1, 2) of the timer mode registers (TMR) define the mode of operation for the timers. Each TMR may be written to, or read from, at any time. However, to ensure accurate timing, it is important to modify the mode of the timer only when the timer is stopped. Inputs of 000 or 111 define a NOP (no operation) mode. In either of these modes (0 or 7) the timer is stopped, INTCLK is high, and the output is inactive. Inputs of 001 through 110 will select one of six distinct timer functions.

In the explanations that follow, assume that the modulus register for the timer was loaded with the appropriate value (0004) by writing to the low and high bytes of each timer modulus register. Assume also, that the prescale is ÷1.

- **Event Counter (mode 1 TMR bits = 001)**

In this non-gated mode the count is decremented for each clock period (INTCLK) input to the timer block (see *Figure 6a*). When the count reaches zero, the output goes valid and remains valid, until the read buffer is read by the CPU or the timer stop register is written.

At the terminal count (0) the modulus is reloaded into the timer block and the count continues even when the output is valid. This mode can be used to cause periodic interrupts to the CPU.

- **Accumulative Timer (mode 2, TMR bits = 010)**

In this gated mode, the counter will decrement only when the gate input is active (see *Figure 6b*). If the gate becomes inactive, the counter will hold at its present value and continue to decrement when the gate again becomes active. When the count decrements to zero, the output becomes valid and remains valid until the count is read by the CPU or the timer is stopped.

At the terminal count the timer is reloaded and the count continues as long as the gate is active.

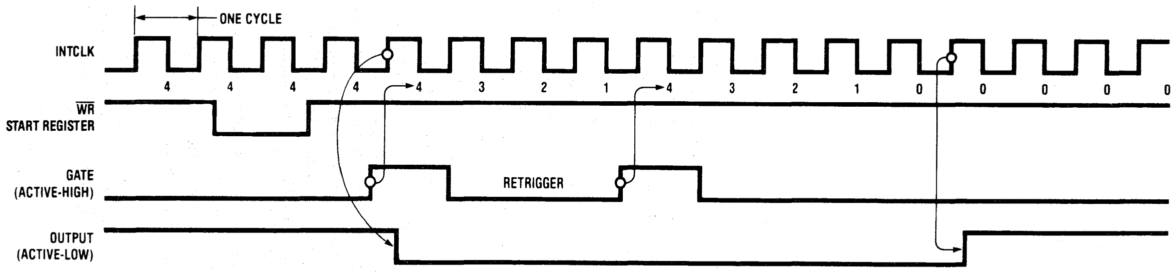
This mode can be used to time processor independent events and to interrupt the CPU when they occur. The prescale and modulus need to be longer than the expected event duration and the gate should go inactive at the event, to preserve the read buffer count for the CPU.

- **Restartable Timer (mode 3, TMR bits = 011)**

In this gated mode, the counter will decrement only when the gate input is active. If the gate becomes inactive, the counter will reload the modulus and hold this value until the gate again becomes active (see *Figure 6c*). If the timer is read when the gate is inactive, you will always read the value the timer has counted down to, not the value the timer has been reloaded with.

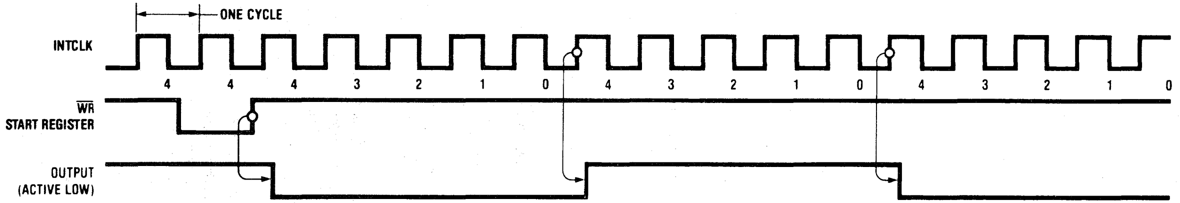
At terminal count the output becomes valid and the timer is reloaded. The timer will continue to run as normal, the only difference is the output is valid. The output remains valid until the count is read by the CPU or the timer stop register is written.

NOTE: The gate inactive time must be longer than the high time of the internal clock (INTCLK) on the chip. Therefore, with ÷64 prescale selected the gate inactive time must be 33 input clocks or greater.



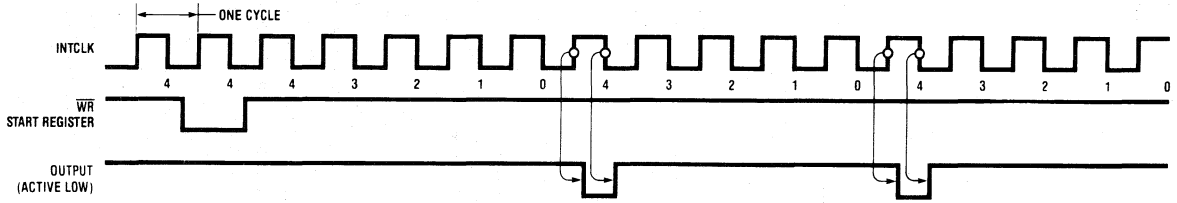
TL/C/5517-20

FIGURE 6d. One Shot (Mode 4)



TL/C/5517-21

FIGURE 6e. Square Wave (Mode 5)



TL/C/5517-22

FIGURE 6f. Pulse Generator (Mode 6)

9.0 Functional Description (Continued)

• One Shot Mode (mode 4, TMR bits = 100)

In this gated mode, the timer holds the modulus count until the active gate edge (see *Figure 6d*). The output immediately becomes valid and remains valid as the counter decrements. The gating signal may go inactive without affecting the count. If TG (the gate) becomes inactive and returns active prior to the terminal count, the modulus will be reloaded, retriggering the one shot period. When the timer reaches the terminal count, the output becomes inactive (see NOTE). The gate, in this mode, is edge sensitive; the active edge is defined by the TMR.

NOTE: The one shot cannot be retriggered during its last internal count (INTCLK) regardless of prescaler selected. Therefore, using the divide by 1 prescaler, it cannot be retriggered during the last clock (TIN), using the divide by 2 prescaler during the last two clocks (TIN) and using the divide by 64 prescaler during the last 64 clocks (TIN).

• Square Wave Mode (mode 5, TMR bits = 101)

In this non-gated mode, the output will go active as soon as the timer is started. The counter decrements for each clock period (INTCLK) and complements its output when zero is reached (see *Figure 6e*). The modulus is then reloaded and counting continues. Assuming a regular clock input, the output will then be a square wave with a period equal to twice the prescale value times the value loaded into the modulus +1 (see equation Timer section intro.). Therefore, varying the modulus will vary the period of the square wave.

• Pulse Generator (mode 6, TMR bits = 110)

In this non-gated mode, the counter decrements for each period of INTCLK (see *Figure 6f*). When the terminal count is reached the output becomes valid for ½ of the TIN clock width for a prescale of ÷1, for one full TIN clock width for a prescale of ÷2 and for 32 TIN clock widths for a prescale of ÷64. The modulus is then reloaded and the sequence is repeated. Varying the prescale and modulus varies the frequency of the pulse.

9.4.4 Timer Programming

The following is the proper sequence to program the timer and should always be used:

1. Write timer mode register selecting mode 0 or 7. This stops the timer, resets the prescaler, and sets internal clock high.
2. Write timer mode register again, this time loading it for your requirements.
3. Write the modulus values, low byte first, high byte second.
4. Start the timers.

The timer read buffer is only updated when the internal timer clock (INTCLK) makes a negative-going transition. Therefore, enough input clock cycles (TIN) must occur to cause a transition of INTCLK given the programmed pre-scaler. After the first transition, the new modulus will be loaded into the read buffer and it can then be read by the CPU.

To guarantee the integrity of the data during a read operation, updates to the timer read buffer are blocked out. If an update is blocked out due to a read, the read buffer will not be updated until the next active transition of INTCLK. Thus, it would appear as if a count was skipped between reads. For example, if the output latches were FF when a block out (read) occurred, the next update could occur at FD, thereby giving an appearance that the count FE was skipped. In actuality the correct number of clocks has occurred for the read buffer to hold FD.

Writing the modulus value when the timer is running does not update the timer immediately. The new value written will get into the timer when the timer reaches its terminal count and reloads its value. If the timer is stopped and a modulus is written the new modulus value will get into the timer when the internal clock is high during the modulus write or on the next low to high internal clock transition. The next time the timer reaches its terminal count it will load the new modulus into the timer. One way to guarantee the new modulus will get into the timer is to follow steps 1 through 4. Although this procedure guarantees that the data will get into the timer you will not be able to read it back until you get a negative-going transition on the internal clock.

Rewriting modulus does not reset the prescaler. The only way to reset the prescaler is to write the mode register and have the internal clock signal be high for some period between the write of the mode register and the start of the timer. Once again, steps 1 through 4 will reset the prescaler.

10.0 NSC810A/883 MIL-STD-883 Class B Screening

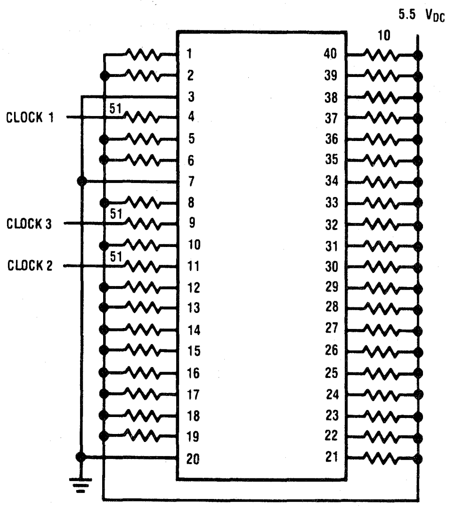
National Semiconductor offers the NSC810AD and NSC810AE with full class B screening per MIL-STD-883 for Military/Aerospace programs requiring high reliability. In addition, this screening is available for all of the key NSC800 peripheral devices.

Electrical testing is performed in accordance with RETS810AX, which tests or guarantees all of the electrical performance characteristics of the NSC810A data sheet. A copy of the current revision of RETS810AX is available upon request. The following table is the MIL-STD-883 flow as of the date of publication.

Test	MIL-STD-883 Method/Condition	Requirement
Internal Visual	2010 B	100%
Stabilization Bake	1008 C 24 Hrs. @ +150°C	100%
Temperature Cycling	1010 C 10 Cycles −65°C/ +150°C	100%
Constant Acceleration	2001 E 30,000 G's, Y1 Axis	100%
Fine Leak	1014 A or B	100%
Gross Leak	1014 C	100%
Burn-In	1015 160 Hrs. @ +125°C (using burn-in circuits shown below)	100%
Final Electrical	+25°C DC per RETS810AX	100%
PDA	5% Max	
	+125°C AC and DC per RETS810AX	100%
	−55°C AC and DC per RETS810AX	100%
	+25°C AC per RETS810AX	100%
QA Acceptance	5005	Sample per
Quality Conformance	5056	Method 5005
External Visual	2009	100%

11.0 Burn-In Circuit

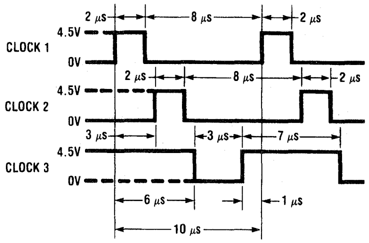
5242HR
NSC810AD/883B (Dual-In-Line)



TL/C/5517-23

12.0 Timing Diagram

Input Clocks



TL/C/5517-24

Note 1: All resistors ±5%, ¼ watt unless otherwise designated, 125°C operating life circuit.

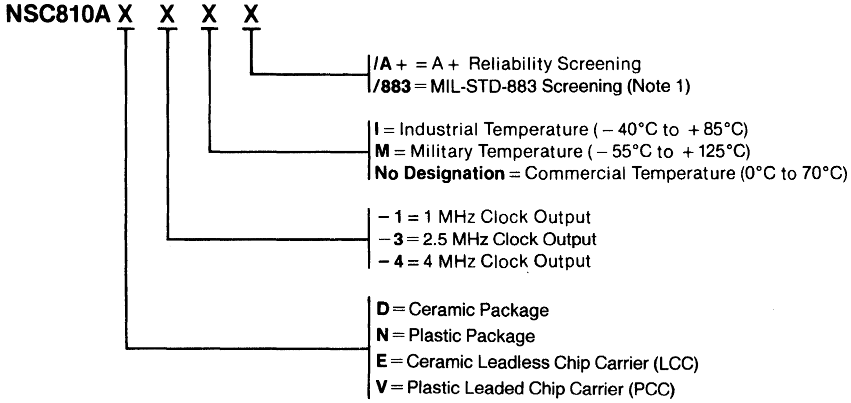
Note 2: E package burn-in circuit 5244HR is functionally identical to the D package.

Note 3: All resistors 2.7 kΩ unless marked otherwise.

Note 4: All clocks 0V to 4.5V.

Note 5: Device to be cooled down under power after burn-in.

13.0 Ordering Information


NSC810A X X X X
/A+ = A+ Reliability Screening
/883 = MIL-STD-883 Screening (Note 1)
I = Industrial Temperature (−40°C to +85°C)
M = Military Temperature (−55°C to +125°C)
No Designation = Commercial Temperature (0°C to 70°C)
−1 = 1 MHz Clock Output
−3 = 2.5 MHz Clock Output
−4 = 4 MHz Clock Output
D = Ceramic Package
N = Plastic Package
E = Ceramic Leadless Chip Carrier (LCC)
V = Plastic Leaded Chip Carrier (PCC)

TL/C/5517-25

Note 1: Do not specify a temperature option; all parts are screened to military temperature.

14.0 Reliability Information

Gate Count	4000
Transistor Count	14,000

National
Semiconductor
Corporation

microCMOS

NSC831 Parallel I/O

General Description

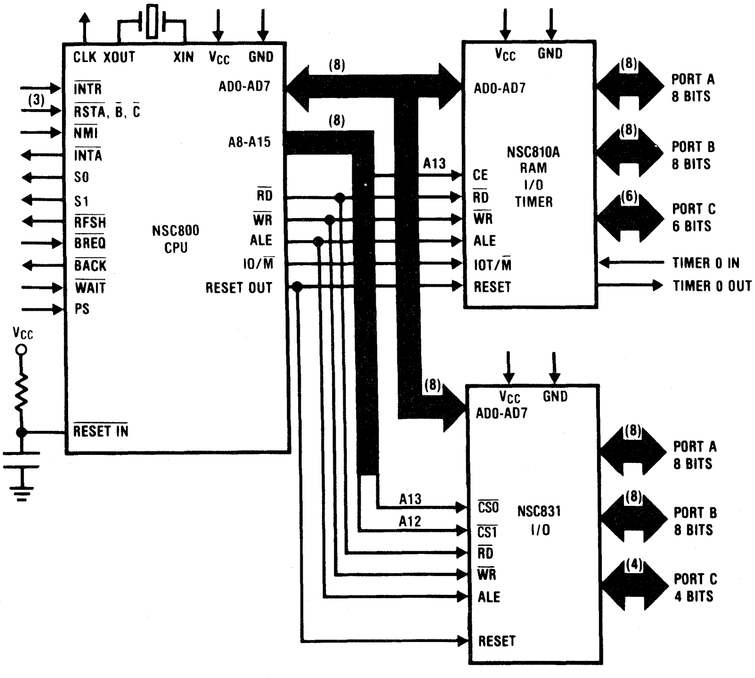
The NSC831 is an I/O device which is fabricated using microCMOS silicon gate technology, functioning as an input/output peripheral interface device. It consists of 20 programmable input/output bits arranged as three separate ports, with each bit individually definable as an input or output. The port bits can be set or cleared individually and can be written to or read from in bytes. Several types of strobed mode operations are available through Port A.

For military applications the NSC831 is available with class B screening in accordance with methods 5004 of MIL-STD-883.

Features

- Three programmable I/O ports
- Single 5V Power Supply
- Very low power consumption
- Fully static operation
- Single-instruction I/O bit operations
- Directly compatible with NSC800 family
- Strobed modes available on Port A

Microcomputer Family Block Diagram



TL/C/5594-1

Table of Contents

1.0 Absolute Maximum Ratings

If Military/Aerospace specified devices are required, contact the National Semiconductor Sales Office/ Distributors for availability and specifications.

Storage Temperature Range	−65°C to +150°C
Voltage at Any Pin With Respect to Ground	−0.3V to V_{CC} + 0.3V
V_{CC}	7V
Lead Temp. (Soldering, 10 seconds)	300°C
Power Dissipation	1W

Note: *Absolute maximum ratings are those values beyond which the safety of the device cannot be guaranteed. Continuous operation at these limits is not intended; operation should be limited to those conditions specified under DC Electrical Characteristics.*

2.0 Operating Range V_{CC} = 5V ±10%

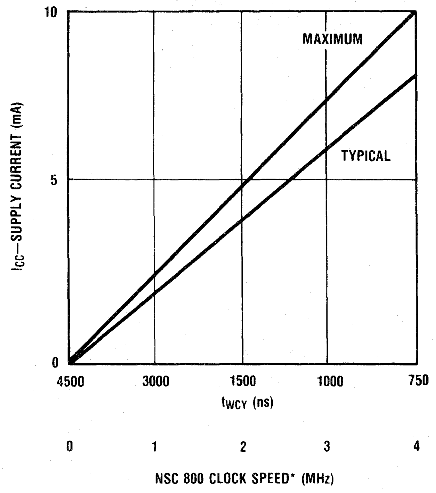
NSC831-1:	0°C to +70°C
	−40°C to +85°C
NSC831-3:	−40°C to +85°C
	−55°C to +125°C
NSC831-4:	0°C to +70°C
	−40°C to +85°C
	−55°C to +125°C

3.0 DC Electrical Characteristics V_{CC} = 5V ±10%, GND = 0V, unless otherwise specified

Symbol	Parameter	Test Conditions	Min	Typ	Max	Units
V_{IH}	Logical 1 Input Voltage		0.8 V_{CC}		V_{CC}	V
V_{IL}	Logical 0 Input Voltage		0		0.2 V_{CC}	V
V_{OH}	Logical 1 Output Voltage	I_{OH} = −1.0 mA	2.4			V
		I_{OUT} = −10 μA	4.0V			V
V_{OL}	Logical 0 Output Voltage	I_{OL} = 2 mA	0		0.4	V
		I_{OUT} = 10 μA	0		0.1	V
I_{IL}	Input Leakage Current	0 ≤ V_{IN} ≤ V_{CC}	−10.0		10.0	μA
I_{OL}	Output Leakage Current	0 ≤ V_{IN} ≤ V_{CC}	−10.0		10.0	μA
I_{CC}	Active Supply Current	I_{OUT} = 0, t_{WCY} = 750 ns		15	20	mA
I_Q	Quiescent Current	RESET = 0, $\overline{RD}$ = 1, WR = 1, ALE = X, V_{IN} = 0, or V_{IN} = V_{CC} No Input Switching, T_A = 25°C		10	100	μA
C_{IN}	Input Capacitance			4	7	pF
C_{OUT}	Output Capacitance			6	10	pF
V_{CC}	Power Supply Voltage	(Note 1)	2.4	5	6	V

Note 1: Operation at lower power supply voltages will reduce the maximum operating speed. Operation at voltages other than 5V ±10% is guaranteed by design, not tested.

I_{CC} vs. SPEED



TL/C/5594-2

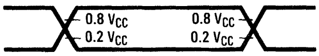
*When NSC831 is used with NSC800

4.0 AC Electrical Characteristics V_{CC} = 5V ±10%, GND = 0V

Symbol	Parameter	Test Conditions	NSC831-1		NSC831-3		NSC831-4		Units
			Min	Max	Min	Max	Min	Max	
t_{ACC}	Access Time from ALE	C_L = 150 pF		1000		400		250	ns
t_{AH}	AD0–AD7, CE, IO/$\overline{M}$ Hold Time		100		60		30		ns
t_{ALE}	ALE Strobe Width (High)		200		130		75		ns
t_{ARW}	ALE to $\overline{RD}$ or $\overline{WR}$ Strobe		150		120		75		ns
t_{AS}	AD0–AD7, CE, IO/$\overline{M}$ Setup Time		100		45		40		ns
t_{DH}	Data Hold Time		150		90		40		ns
t_{DO}	Port Data Output Valid			350		320		300	ns
t_{DS}	Data Setup Time		100		80		50		ns
t_{PE}	Peripheral Bus Enable			320		200		200	ns
t_{PH}	Peripheral Data Hold Time		150		125		100		ns
t_{PS}	Peripheral Data Setup Time		100		75		50		ns
t_{PZ}	Peripheral Bus Disable (TRI-STATE®)			150		150		150	ns
t_{RB}	$\overline{RD}$ to BF Output			300		300		300	ns
t_{RD}	Read Strobe Width		400		320		220		ns
t_{RDD}	Data Bus Disable		0	100	0	75	0	75	ns
t_{RI}	$\overline{RD}$ to $\overline{INTR}$ Output			320		300		300	ns
t_{RWA}	$\overline{RD}$ or $\overline{WR}$ to Next ALE		125		100		45		ns
t_{SB}	$\overline{STB}$ to BF Valid			300		300		300	ns
t_{SH}	Peripheral Data Hold With Respect to $\overline{STB}$		150		125		100		ns
t_{SI}	$\overline{STB}$ to INTR Output			300		300		300	ns
t_{SS}	Peripheral Data Setup With Respect to $\overline{STB}$		100		75		50		ns
t_{SW}	$\overline{STB}$ Width		400		320		220		ns
t_{WB}	$\overline{WR}$ to BF Output			340		300		300	ns
t_{WI}	$\overline{WR}$ to $\overline{INTR}$ Output			320		300		300	ns
t_{WR}	$\overline{WR}$ Strobe Width		400		320		220		ns
t_{WCY}	Width of Machine Cycle		3000		1200		750		ns

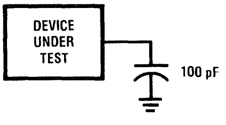
Note: Test conditions: t_{WCY} = 3000 ns for NSC831-1, 1200 ns for NSC831-3, 750 ns for NSC831-4

AC TESTING INPUT/OUTPUT WAVEFORM



TL/C/5594-3

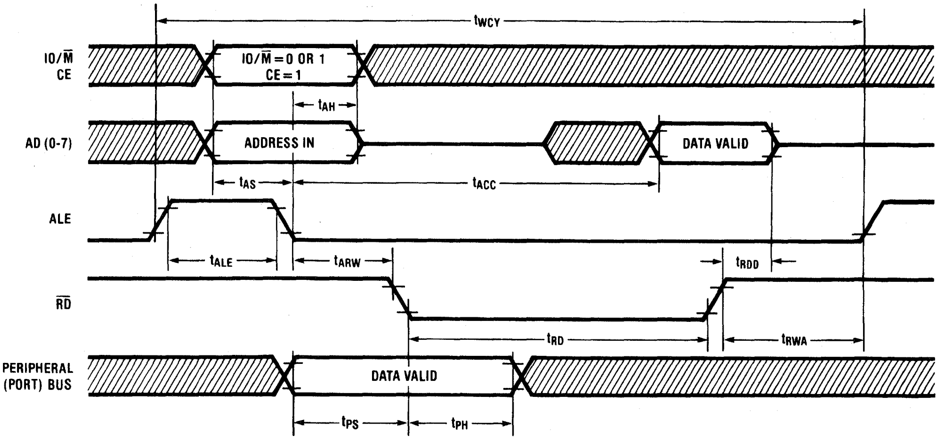
AC TESTING LOAD CIRCUIT



TL/C/5594-4

5.0 Timing Waveforms

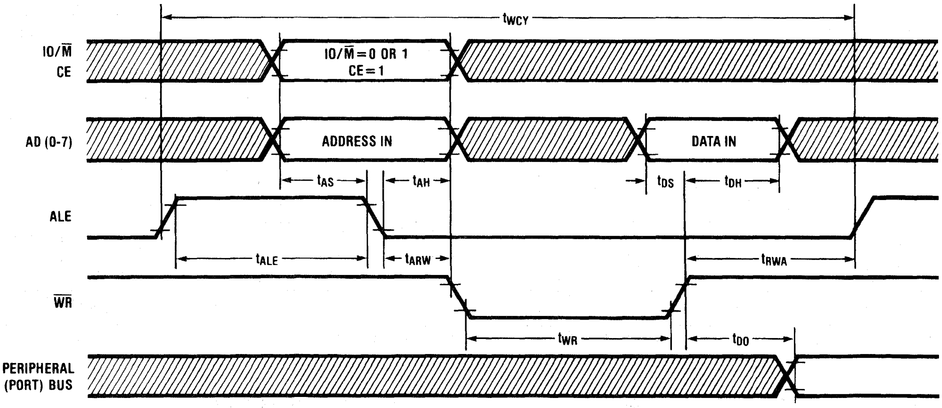
Read Cycle (Read from Port)



TL/C/5594–5

Note: Diagonal lines indicate interval of invalid data.

Write Cycle (Write to Port)

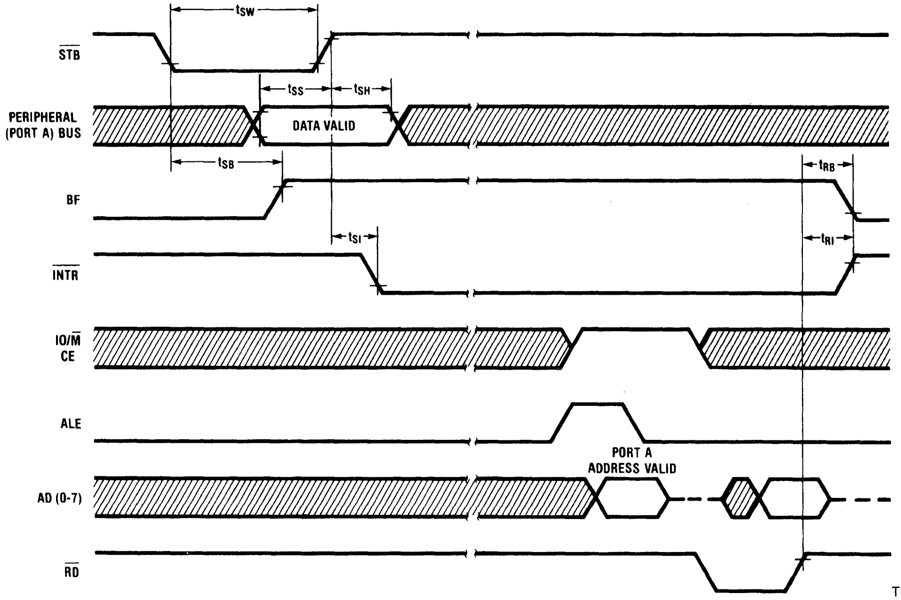


TL/C/5594–6

Note: Diagonal lines indicate interval of invalid data.

5.0 Timing Waveforms (Continued)

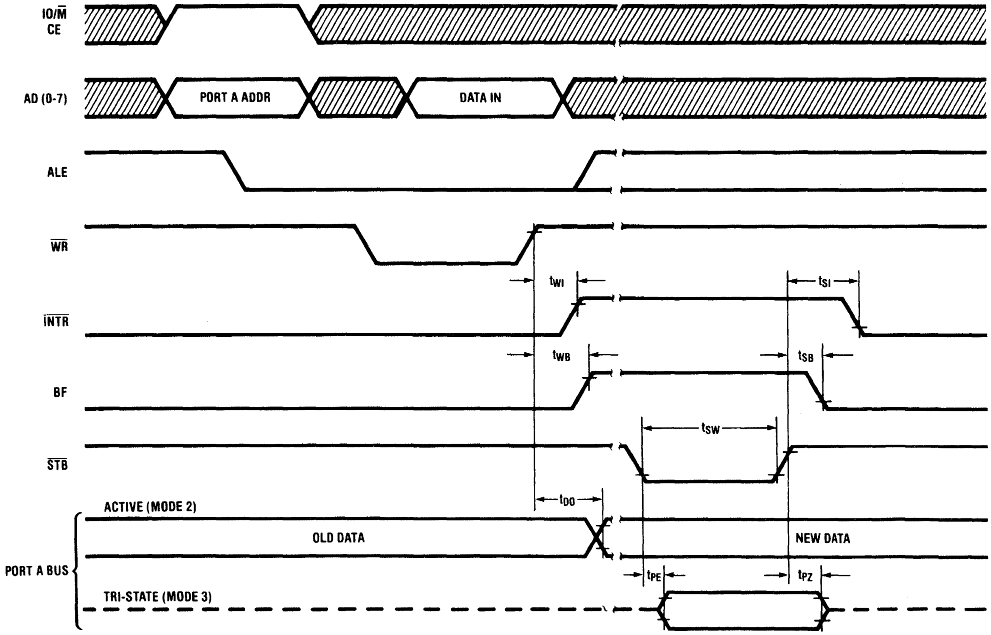
Strobed Mode Input



TL/C/5594-7

Note: Diagonal lines indicate interval of invalid data.

Strobed Mode Output



TL/C/5594-8

Note: Diagonal lines indicate interval of invalid data.

6.0 Pin Descriptions

The following describes the function of all NSC831 input/output pins. Some of these descriptions reference internal circuits.

6.1 INPUT SIGNALS

Master Reset (RESET): An active-high input on the RESET pin initializes the chip causing the three I/O ports (A, B and C) to revert to the input mode. The three ports, the three data direction registers and the mode definition register are reset to low (0).

Chip Enable ($\overline{CE_0}$, $\overline{CE_1}$): The CE inputs must be active at the falling edge of ALE. At ALE time, the CE inputs are latched to provide access to the NSC831.

Read ($\overline{\text{RD}}$): when the $\overline{\text{RD}}$ input is an active low, data is read from the AD0–AD7 bus.

Write ($\overline{\text{WR}}$): When the CE inputs are active an active low $\overline{\text{WR}}$ input causes the selected output port to be written with the data from the AD0–AD7 bus.

Address Latch Enable (ALE): The trailing edge (high to low transition) of the ALE input signal latches the address/data present on the AD0–AD7 bus, plus the input control signals on $\overline{CE_0}$ and $\overline{CE_1}$.

Power (V_{CC}): 5V power supply.

Ground (V_{SS}): Ground reference.

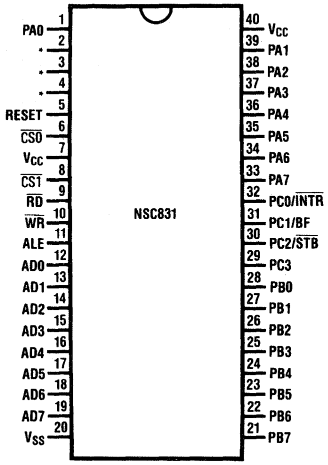
6.2 INPUT/OUTPUT SIGNALS

Bidirectional Address/Data Bus AD0–AD7: The lower 8 bits of the I/O address are applied to these pins, and latched by the trailing edge of ALE. During read operations, 8 bits are present on these pins, and are read when $\overline{\text{RD}}$ is low. During an I/O write cycle, Port A, B, or C is written with the data present on this bus at the trailing edge of the $\overline{\text{WR}}$ strobe.

Ports A, B, C (PA0–PA7, PB0–PB7, PC0–PC3): These are general purpose I/O pins. Their input/output direction is determined by the contents of the Data Direction Register (DDRs).

7.0 Connection Diagrams

Dual-In-Line Package



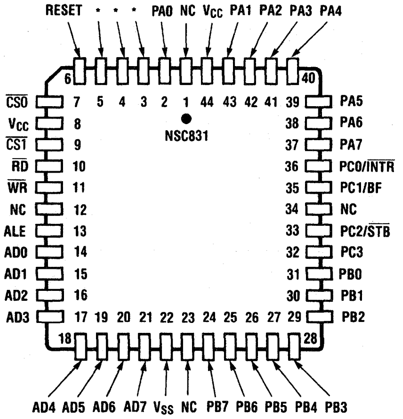
TL/C/5594-9

Top View

*Tie pins 2, 3, and 4 to either V_{CC} or V_{SS}.

Order Number NSC831D or N
See NS Package Number D40C or N40A

Leadless Chip Carrier


RESET * * * PA0 NC V_{CC} PA1 PA2 PA3 PA4
6 7 5 4 3 2 1 44 43 42 41 39 40
$\overline{\text{CS0}}$ 7
V_{CC} 8
$\overline{\text{CS1}}$ 9
$\overline{\text{RD}}$ 10
$\overline{\text{WR}}$ 11
NC 12
ALE 13
AD0 14
AD1 15
AD2 16
AD3 17
NSC831
39 PA5
38 PA6
37 PA7
36 PC0/$\overline{\text{INTR}}$
35 PC1/BF
34 NC
33 PC2/$\overline{\text{STB}}$
32 PC3
31 PB0
30 PB1
29 PB2
18 17 19 20 21 22 23 24 25 26 27 29 28
AD4 AD5 AD6 AD7 V_{SS} NC PB7 PB6 PB5 PB4 PB3

NC = NO CONNECT

TL/C/5594-10

Top View

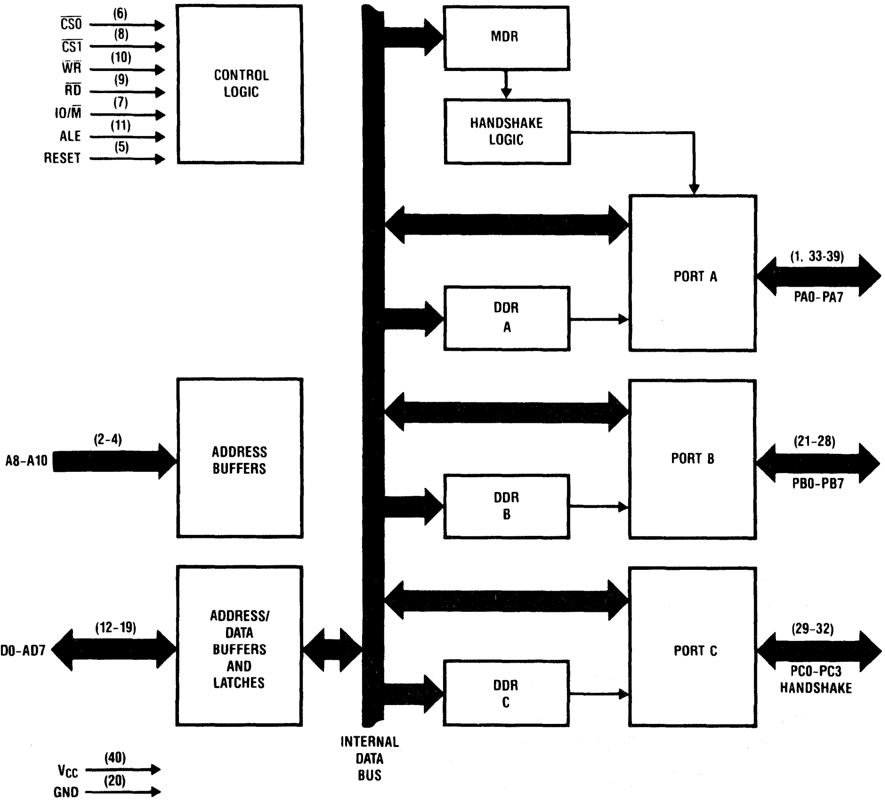
Order Number NSC831E
See NS Package Number E44A

8.0 Functional Description

Refer to *Figure 1* for a detailed block diagram of the NSC831, while reading the following paragraphs.

Input/Output (I/O): The I/O of the NSC831 contains three sets called Ports. There are two ports (A and B) which contain 8 bits each and one port (Port C) which has 4 bits. Any bit or combination of bits in a port may be addressed with Set or Clear commands. A port can also be addressed as an 8-bit word (4 bits for Port C). When reading Port C, bits 4–7 will be read as ones. All ports share common functions of Read, Write, Bit-Set and Bit-Clear. Additionally, Port A is programmable for strobed (handshake mode input or output. Port C has a programmable second function for each bit associated with strobed modes. Table I defines the address location of the ports and control registers.

8.1 BLOCK DIAGRAM



TL/C/5594-11

Note: Applicable pinout for 40 pin dual-in-line package within parentheses.

FIGURE 1

8.0 Functional Description (Continued)

8.2 I/O PORTS

There are three I/O ports (labeled A, B and C) on the NSC831. Ports A and B are 8-bits wide; port C is 4-bits wide. These ports transfer data between the CPU bus and the peripheral bus and vice versa. The way in which these transfers are handled depends upon the currently programmed operating mode.

The NSC831 can be programmed to operate in four different modes. One of these modes (Basic I/O) allows direct transfer of I/O data without any handshaking between the NSC831 and the peripheral. The other three modes (Strobed I/O) provide for timed transfers of I/O data with handshaking between the NSC831 and the peripheral.

Determination of the NSC831 port's mode, data direction and data is done by five registers which are under program control. The Mode Definition Register determines in which of the four I/O modes the chip will operate. Another register (Data Direction Register) establishes the data direction for each bit in that port. The Data Register holds data to be transferred or that which was received. The final two registers per port allow individual data register bits to be cleared (Bit-Clear Register) or data register bits to be set (Bit-Set Register).

Operation during Strobed I/O utilizes two of the port C pins for handshaking and one port C pin to interrupt the CPU.

8.3 REGISTERS

As indicated in the overview, programmable registers control the flow of data through the ports. Table I shows the registers of the NSC831. All registers affecting I/O transfers are in the first grouping of this table.

- **Mode Definition Register (MDR)**

The MDR determines the operating mode for port A and whether or not the lower 3-bits of port C will be used for handshaking (Strobed I/O). Port B always transfers data via the Basic I/O mode, regardless of how the MDR is programmed.

The four modes are as follows:

Mode 0—Basic I/O (Input or Output)

Mode 1—Strobed Mode Input

Mode 2—Strobed Mode Output (Active Peripheral Bus)

Mode 3—Strobed Mode Output (TRI-STATE Peripheral Bus)

The address assignment of the MDR is xxx00111 as shown in Table I. The upper 3 "don't care" bits are determined by the users decode logic (chip enable address). Table II specifies the data that must be loaded into the MDR to select the mode.

- **Data Direction Registers (DDR)**

Each port has a DDR that determines whether an individual port bit will be an input or an output. If DDR for the port bit is set to a 1, then that port bit is an output. If its DDR is reset to a 0, then it is an input. The DDR bits cannot be individually written to; the entire DDR register is affected by a write to the DDR. Thus, all data bits written must be consistent for all desired port bit directions.

TABLE I. I/O and Timer Address Designations

8-Bit Address Field Bits								Designation I/O Port, Timer, etc.	R (Read) W (Write)
7	6	5	4	3	2	1	0		
x	x	x	x	0	0	0	0	Port A (Data)	R/W
x	x	x	x	0	0	0	1	Port B (Data)	R/W
x	x	x	x	0	0	1	0	Port C (Data)	R/W
x	x	x	x	0	0	1	1	Not Used	**
x	x	x	x	0	1	0	0	DDR - Port A	W
x	x	x	x	0	1	0	1	DDR - Port B	W
x	x	x	x	0	1	1	0	DDR - Port C	W
x	x	x	x	0	1	1	1	Mode Definition Reg.	W
x	x	x	x	1	0	0	0	Port A - Bit-Clear	W
x	x	x	x	1	0	0	1	Port B - Bit-Clear	W
x	x	x	x	1	0	1	0	Port C - Bit-Clear	W
x	x	x	x	1	0	1	1	Not Used	**
x	x	x	x	1	1	0	0	Port A - Bit-Set	W
x	x	x	x	1	1	0	1	Port B - Bit-Set	W
x	x	x	x	1	1	1	0	Port C - Bit-Set	W
x	x	x	x	1	1	1	1	Not Used	**

x = don't care

LB = low-order byte

HB = high-order byte

* A write accesses the modulus register, a read the read buffer.

** A read from an unused location reads invalid data, a write does not affect any operation of NSC831.

TABLE II. Mode Definition Register Bit Assignments

Mode	Bit							
	7	6	5	4	3	2	1	0
0	x	x	x	x	x	x	x	0
1	x	x	x	x	x	x	0	1
2	x	x	x	x	x	0	1	1
3	x	x	x	x	x	1	1	1

8.0 Functional Description (Continued)

Any write or read to the port bits contradicting the direction established by the DDR will not affect the port bits output or input. However, a write to a port bit, defined as an input, will modify the output latch and a read to a port bit, defined as an output, will read this output latch. See *Figure 2*.

- **Data Registers**

These registers contain the actual data being transferred between the CPU and the peripheral. In Basic I/O, data presented by the peripheral (read cycle) will be latched on the falling edge of $\overline{RD}$. Data presented by the CPU (write cycle) will be valid after the rising edge of $\overline{WR}$ (see AC characteristics for exact timing).

During Strobed I/O, data presented by the peripheral must be valid on the rising edge of $\overline{STB}$. Data received by the peripheral will be valid on the rising edge of $\overline{STB}$. Data latched by the port on the rising edge of $\overline{STB}$ will be preserved until the next CPU read or $\overline{STB}$ signal.

- **Bit Set-Clear Registers**

The I/O features of the RAM-I/O-timer allow modification of a single bit or several bits of a port with the Bit-Set and Bit-Clear commands. The address selected indicates whether a Bit-Set or Clear will take place. The incoming data on the address/data bus is latched at the trailing edge of the $\overline{WR}$ strobe and is treated as a mask. All bits containing 1s will cause the indicated operation to be performed on the corresponding port bit. All bits of the mask with 0s cause the corresponding port bits to remain unchanged. Three sample operations are shown in Table III using port B as an example.

TABLE III. Bit-Set and Clear Examples

Operation Port B	Set B7	Clear B2 and B0	Set B4, B3 and B1
Address	xxx01101	xxx01001	xxx01101
Data	10000000	00000101	00011010
Port Pins			
Prior State	00001111	10001111	10001010
Next State	10001111	10001010	10011010

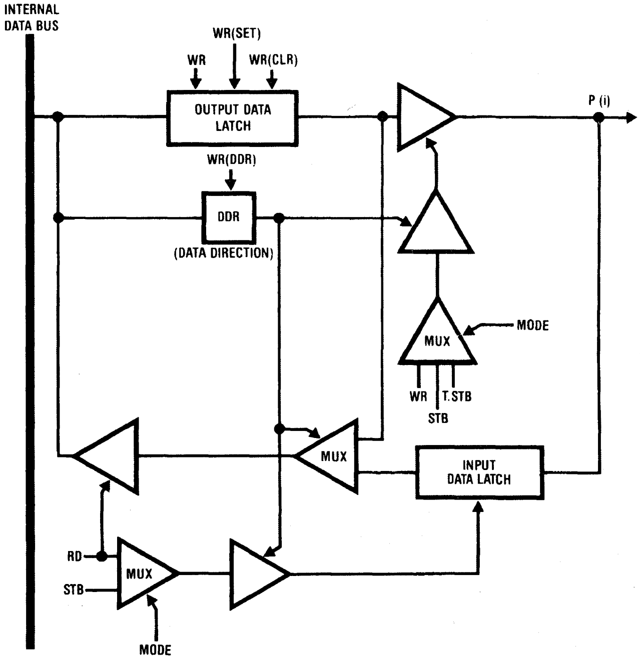
8.4 MODES

Two data transfer modes are implemented: Basic I/O and Strobed I/O. Strobed I/O can be further subdivided into three categories: Strobed Input, Strobed Output (active peripheral bus) and Strobed Output (TRI-STATE peripheral bus). The following descriptions detail the functions of these categories.

- **Basic I/O**

Basic I/O mode uses the $\overline{RD}$ and $\overline{WR}$ CPU bus signals to latch data at the peripheral bus. This mode is the permanent mode of operation for ports B and C. Port A is in this mode if the MDR is set to mode 0. Read and write byte operations and bit operations can be done in Basic I/O. Timing for these modes is shown in the AC Characteristics Table and described with the data register definitions.

When the NSC831 is reset, all registers are cleared to zero. This results in the basic mode of operation being selected, all port bits are made inputs and the output latch for each port bit is cleared to zero. The NSC831, at this point, can read data from any peripheral port without further set-up. If outputs are desired, the CPU merely has to program the appropriate DDR and then send data to the data ports.



TL/C/5594-12

FIGURE 2

8.0 Functional Description (Continued)

• Strobed I/O

Strobed I/O Mode uses the $\overline{\text{STB}}$, BF and $\overline{\text{INTR}}$ signals to latch the data and indicate that new data is available for transfer. Port A is used for the transfer of data when in any of the Strobed modes. Port B can still be used for Basic I/O and the lower 3-bits of port C are now the three handshake signals for Strobed I/O. Timing for this mode is shown in the AC Characteristic Tables.

Initializing the NSC831 for Strobed I/O Mode is done by loading the data shown in Table IV into the specified register. The registers should be loaded in the order (left to right) that they appear in Table IV.

TABLE IV. Mode Definition Register Configurations

Mode	MDR	DDR Port A	DDR Port C	Port C Output Latch
Basic I/O	xxxxxxx0	Port bit directions are determined by the bits of each port's DDR		
Strobed Input	xxxxxx01	00000000	xxx011	xxx1xx
Strobed Output (Active)	xxxxx011	11111111	xxx011	xxx1xx
Strobed Output (TRI-STATE)	xxxxx111	11111111	xxx011	xxx1xx

• Strobed Input (Mode 1)

During strobed input operations, an external device can load data into port A with the $\overline{\text{STB}}$ signal. Data is input to the PA0–7 input latches on the leading (negative) edge of $\overline{\text{STB}}$, causing BF to go high (true). On the trailing (positive) edge of $\overline{\text{STB}}$ the data is latched and the interrupt signal, $\overline{\text{INTR}}$, becomes valid indicating to the CPU that new data is available. $\overline{\text{INTR}}$ becomes valid only if the interrupt is enabled, that is the output data latch for PC2 is set to 1.

When the CPU reads port A, address x'00, the trailing edge of the $\overline{\text{RD}}$ strobe causes BF and $\overline{\text{INTR}}$ to become inactive, indicating that the strobed input cycle has been completed.

• Strobed Output—Active (Mode 2)

During strobed output operations, an external device can read data from port A using the $\overline{\text{STB}}$ signal. Data is initially loaded into port A by the CPU writing to I/O address x'00. On the trailing edge of $\overline{\text{WR}}$, $\overline{\text{INTR}}$ is set inactive and BF becomes valid indicating new data is available for the external device. When the external device is ready to accept the data in port A it pulses the $\overline{\text{STB}}$ signal. The rising edge of $\overline{\text{STB}}$ resets BF and activates the $\overline{\text{INTR}}$ signal. $\overline{\text{INTR}}$ becomes valid only if the interrupt is enabled, that is the output latch for PC2 is set to 1. $\overline{\text{INTR}}$ in this mode indicates a condition that requires CPU intervention (the output of the next byte of data).

• Strobed Output—TRI-STATE (Mode 3)

The Strobed Output TRI-STATE Mode and the Strobed Output active (peripheral) bus mode function in a similar manner with one exception. The exception is that the data signals on PA0–7 assume the high impedance state at all times except when accessed by the $\overline{\text{STB}}$ signal. Thus, in addition to its timing function, $\overline{\text{STB}}$ enables port A outputs to active logic levels. This Mode 3 operation allows other data sources, in addition to the NSC831, to access the peripheral bus. Strobed Mode 3 is identical to Strobed Mode 2, except as indicated above.

Example Mode 1 (Strobed Input):

Action Taken	$\overline{\text{INTR}}$	BF	Results of Action
INITIALIZATION			
Reset NSC831	H	L	Basic input mode all ports.
Load 01'H into MDR	H	L	Strobed input mode entered; no byte loads to port C after this step; bit-set and clear commands to $\overline{\text{INTR}}$ and BF no longer work.
Load 00'H into DDR A	H	L	Sets data direction register for port A to input; data from port A peripheral bus is available to the CPU if the $\overline{\text{STB}}$ signal is used, other handshake signals aren't initialized, yet.
Load 03'H into DDR C	H	L	Sets data direction register of port C; buffer full signal works after this step and it is unaffected by the bit-set and clear registers.
Load 04'H into Port C Bit-Set Register	H	L	Sets output latch (PC2) to enable $\overline{\text{INTR}}$; $\overline{\text{INTR}}$ will latch active whenever $\overline{\text{STB}}$ goes low; $\overline{\text{INTR}}$ can be disabled by a bit-clear to PC2.*
OPERATION			
$\overline{\text{STB}}$ pulses low	L	H	Data on peripheral bus is latched into port A; $\overline{\text{INTR}}$ is cleared by a CPU read of port A or a bit-clear of $\overline{\text{STB}}$.
CPU reads Port A	H	L	CPU gets data from port A; $\overline{\text{INTR}}$ is cleared; peripheral is signalled to send next byte via an inactive BF signal. Repeat last two steps until EOT at which time CPU sends bit-clear to the output latch (PC2).

*Port C can be read by the CPU at anytime, allowing polled operation instead of interrupt driven operation.

8.0 Functional Description (Continued)

Example Mode 2 (Strobed Output—active peripheral bus):

Action Taken	$\overline{\text{INTR}}$	BF	Results of Action
INITIALIZE			
Reset NSC831	H	L	Basic input mode all ports.
Load 03'H into MDR	H	L	Strobed output mode entered; no byte loads to port C after this step; bit-set and clear commands to $\overline{\text{INTR}}$ and BF no longer work.
Load FF'H into DDR A	H	L	Sets data direction register for port A to output; data from port A is available to the peripheral if the $\overline{\text{STB}}$ signal is used other handshake signals aren't initialized, yet.
Load 03'H into DDR C	H	L	Sets data direction register of port C; buffer full signal works after this step and it is unaffected by the bit-set and clear registers
Load 04'H into Port C Bit-Set Register	L	L	Sets output latch (PC2) to enable $\overline{\text{INTR}}$; active $\overline{\text{INTR}}$ indicates that CPU should send data; $\overline{\text{INTR}}$ becomes inactive whenever the CPU loads port A; $\overline{\text{INTR}}$ can be disabled by a bit-clear to $\overline{\text{STB}}$.*
OPERATION			
CPU writes to Port A	H	H	Data on CPU bus is latched into port A; $\overline{\text{INTR}}$ is set by the CPU write to port A; active BF indicates to peripheral that data is valid; Peripheral gets data from port A; $\overline{\text{INTR}}$ is reset active; The active $\overline{\text{INTR}}$ signals the CPU to send the next byte. Repeat last two steps until EOT at which time CPU sends bit-clear to the output latch (PC2).
$\overline{\text{STB}}$ pulses low	L	L	

*Port C can be read by the CPU at any time, allowing polled operation instead of interrupt driven operation.

- **Handshaking Signals**

In the Strobed mode of operation, the lower 3-bits of port C transmit/receive the handshake signals (PC0 = $\overline{\text{INTR}}$, PC1 = BF, PC2 = $\overline{\text{STB}}$).

$\overline{\text{INTR}}$ (Strobe Mode Interrupt) is an active-low interrupt from the NSC831 to the CPU. In strobed input mode, the CPU reads the valid data at port A to clear the interrupt. In strobed output mode, the CPU clears the interrupt by writing data to port A.

The $\overline{\text{INTR}}$ output can be enabled or disabled, thus giving it the ability to control strobed data transfer. It is enabled or disabled, respectively, by setting or clearing bit 2 of the port C output data latch ($\overline{\text{STB}}$).

PC2 is always an input during strobed mode of operation, its output data latch is not needed. Therefore, during strobed mode of operation it is internally gated with the interrupt signal to generate the $\overline{\text{INTR}}$ output. Reset clears this bit to zero, so it must be set to one to enable the $\overline{\text{INTR}}$ pin for strobed operation.

Once the strobed mode of operation is programmed, the only way to change the output data latch of PC2 is by using the Bit-Set and Clear registers. The port C byte write command will not alter the output data latch of PC2 during the strobed mode of operation.

$\overline{\text{STB}}$ (Strobe) is an active low input from the peripheral device, signalling a data transfer. The NSC831 latches data on the rising edge of $\overline{\text{STB}}$ if the port bit is an input and the peripheral should latch data on the rising edge of $\overline{\text{STB}}$ if the port bit is an output.

BF (Buffer Full) is a high active output from the NSC831. For input port bits, it indicates that new data has been received from the peripheral. For output port bits, it indicates that new data is available for the peripheral.

Note: In either input or output mode the BF may be cleared by rewriting the MDR.

9.0 NSC831/883B MIL-STD-883 Class B Screening

National Semiconductor offers the NSC831D and NSC831E with full class B screening per MIL-STD-883 for Military/Aerospace programs requiring high reliability. In addition, this screening is available for all of the key NSC800 peripheral devices.

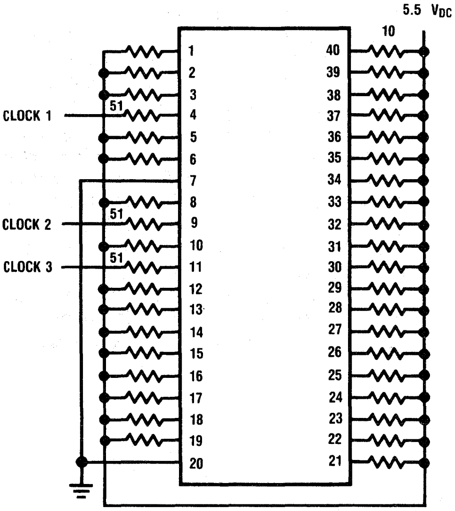
Electrical testing is performed in accordance with RETS831X, which tests or guarantees all of the electrical performance characteristics of the NSC831 data sheet. A copy of the current revision of RETS831X is available upon request. The following table is the MIL-STD-883 flow as of the date of publication.

100% Screening Flow

Test	MIL-STD-883 Method/Condition	Requirement
Internal Visual	2010 B	100%
Stabilization Bake	1008C 24 Hrs. @ +150°C	100%
Temperature Cycling	1010C 10 Cycles −65°C/ +150°C	100%
Constant Acceleration	2001E 30,000 Gs, Y1 Axis	100%
Fine Leak	1014 A or B	100%
Gross Leak	1014C	100%
Burn-In	1015 160 Hrs. @ +125°C (using burn-in circuits shown below)	100%
Final Electrical	+25°C DC per RETS831X	100%
PDA	5% Max	
	+125°C AC and DC per RETS831X	100%
	−55°C AC and DC per RETS831X	100%
	+25°C AC per RETS831X	100%
QA Acceptance	5005	Sample per
Quality Conformance		Method 5005
External Visual	2009	100%

10.0 Burn-In Circuit

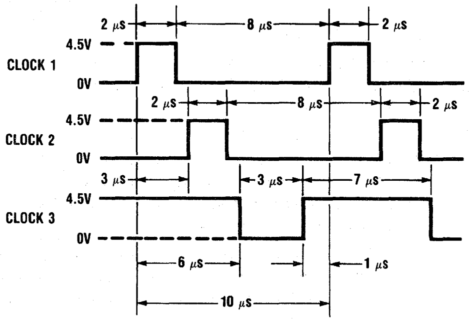
5242HR
NSC831AD/883B (Dual-In-Line)



TL/C/5594–13

11.0 Timing Diagram

Input Clocks



TL/C/5594–14

Note 1: All resistors ±5%, ¼ watt unless otherwise designated, 125°C operating life circuit.

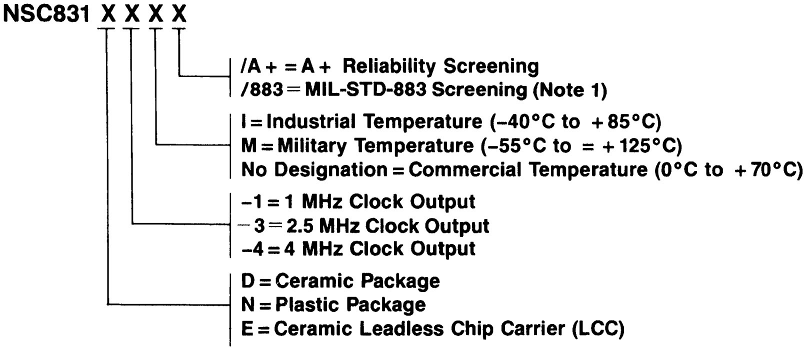
Note 2: E package burn-in circuit 5244HR is functionally identical to the D package.

Note 3: All resistors 2.7 kΩ unless marked otherwise.

Note 4: All clocks 0V to 4.5V.

Note 5: Device to be cooled down under power after burn-in.

12.0 Ordering Information



NSC831 X X X X

/A+ = A+ Reliability Screening
/883 = MIL-STD-883 Screening (Note 1)

I = Industrial Temperature (−40°C to +85°C)
M = Military Temperature (−55°C to = +125°C)
No Designation = Commercial Temperature (0°C to +70°C)

−1 = 1 MHz Clock Output
−3 = 2.5 MHz Clock Output
−4 = 4 MHz Clock Output

D = Ceramic Package
N = Plastic Package
E = Ceramic Leadless Chip Carrier (LCC)

TL/C/5594-15

Note 1: Do not specify a temperature option: all parts are screened to military temperature.

13.0 Reliability Information (NSC831)

Gate Count	1900
Transistor Count	7400

National
Semiconductor
Corporation

microCMOS

NSC858 Universal Asynchronous Receiver/Transmitter

General Description

The NSC858 is a CMOS programmable Universal Asynchronous Receiver/Transmitter (UART). It has an on chip programmable baud rate generator. The UART, which is fabricated using microCMOS silicon gate technology, functions as a serial receiver/transmitter interface for your microcomputer system.

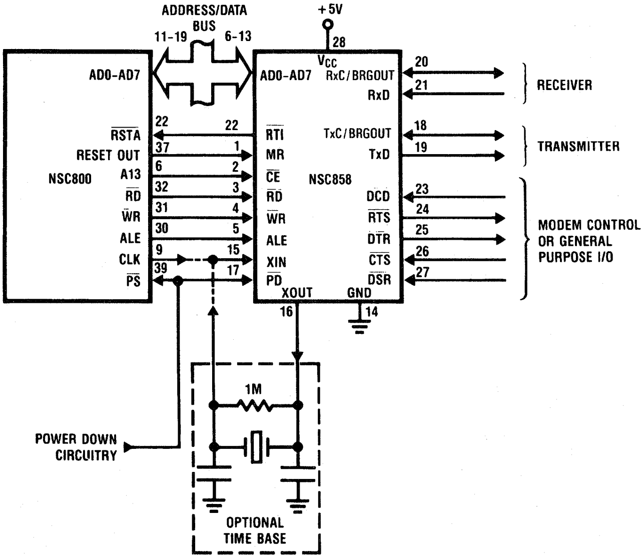
The transmitter converts parallel data from the CPU to serial form and shifts it out in the standard asynchronous communication data format. Appropriate start, parity, and stop bits are added to the outgoing serial stream. Incoming serial data is converted to parallel form by the receiver. The receiver checks incoming data for errors (parity, overrun, framing or break interrupt) and then converts it from serial to parallel for transfer to the CPU. Five pins on the chip are available for modem control functions or general purpose I/O.

The NSC858 has a programmable baud generator that is capable of dividing the timing reference clock input by divisors of 1 to ($2^{16}-1$), and producing a 1X, 16X, 32X, 64X clock for driving the transmitter and/or receiver logic. Both the transmitter and receiver can either be driven by an external clock or the internal baud rate generator. The NSC858 has an interrupt system that can be tailored to the user's requirements. In addition to the CMOS power consumption levels there are hardware and software power down modes which further reduce power consumption levels.

Features

- Maximum baud rate 256k BPS (16X), 1M BPS (1X)
- Programmable baud rate generator
- Double buffered receiver and transmitter
- Independently configured receiver and transmitter
  - — 5-, 6-, 7-, 8-bit characters
  - — Odd, even, force high, force low, or no parity
  - — 1, 1½, 2 stop bits
- Five bits modem I/O or general purpose I/O (3 input, 2 output)
- Programmable auto enables for $\overline{\text{CTS}}$ and $\overline{\text{DCD}}$
- Local and remote loopback diagnostics
- False start bit detection
- Break condition detection and generation
- Program polled, or interrupt driven operation
  - — 8 maskable status conditions for receiver and transmitter interrupt
  - — 4 maskable status conditions for modem interrupt
- Variable power supply (2.4V–6.0V)
- Low power consumption with software and hardware power down modes
- 8-bit multiplexed address/data bus directly compatible with NSC800™

System Configuration



TL/C/5593–1

Table of Contents

1.0 Absolute Maximum Ratings

(Note 1)

If Military/Aerospace specified devices are required, contact the National Semiconductor Sales Office/ Distributors for availability and specifications.

Storage Temperature	−65°C to +150°C
Voltage on Any Pin with Respect to Ground	−0.3V to V_{CC} +0.3V
Maximum V_{CC}	7V
Power Dissipation	1W
Lead Temp. (Soldering, 10 seconds)	300°C

2.0 Operating Conditions V_{CC} = 5V ± 10%

Ambient Temperature

Industrial	−40°C to +85°C
Commercial	0°C to +70°C

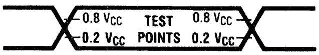
3.0 DC Electrical Characteristics V_{CC} = 5V ± 10%, GND = 0V, unless otherwise specified.

Symbol	Parameter	Conditions	Min	Typ	Max	Units
V_{IH}	Logical 1 Input Voltage		0.8 V_{CC}		V_{CC}	V
V_{IL}	Logical 0 Input Voltage		0		0.2 V_{CC}	V
V_{HY}	Hysteresis at RESET IN Input	V_{CC}=5V	0.25	0.5		V
V_{OH1}	Logical 1 Output Voltage	I_{OUT} = −1.0 mA	2.4			V
V_{OH2}	Logical 1 Output Voltage	I_{OUT} = −10 μA	V_{CC}−0.5			V
V_{OL1}	Logical 0 Output Voltage	I_{OL} = 2 mA except X_{OUT}	0		0.4	V
V_{OL2}	Logical 0 Output Voltage	I_{OUT} = 10 μA	0		0.1	V
I_{IL}	Input Leakage Current	$0 \leq V_{IN} \leq V_{CC}$	−10.0		10.0	μA
I_{OL}	Output Leakage Current	$0 \leq V_{IN} \leq V_{CC}$	−10.0		10.0	μA
I_{CC}	Active Supply Current	T_A = 25°C		2	10	mA
I_{HPD}	Current Hardware Power Down	Pin $\overline{PD}$=0, No Resistive Output Loads, V_{IN}=0V or V_{IN}=V_{CC}, T_A = 25°C		100		μA
I_{SPD}	Current Software Power Down	Power Down Reg Bit 0=1, No Resistive Output Loads, V_{IN}=0V or V_{IN}=V_{CC}, T_A = 25°C		300		μA
C_{IN}	Input Capacitance			6	10	pF
C_{OUT}	Output Capacitance			8	12	pF
V_{CC}	Power Supply Voltage	(Note 2)	2.4	5	6	V

Note 1: Absolute Maximum Ratings indicate limits beyond which permanent damage may occur. Continuous operation at these limits is not intended and should be limited to those conditions specified under DC Electrical Characteristics.

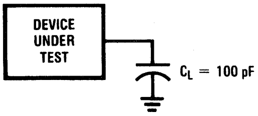
Note 2: Operation at lower power supply voltages will reduce the maximum operating speed. Operation at voltages other than 5V ±10% is guaranteed by design, not tested.

AC Testing Input/Output Waveform



TL/C/5593-2

AC Testing Load Circuit



TL/C/5593-3

4.0 AC Electrical Characteristics $V_{CC} = 5V \pm 10\%$, GND = 0V, $C_L = 100$ pF

Symbol	Parameter	Test Conditions	Min	Typ	Max	Units
BUS						
t_{AS}	Address 0–7 Set-Up Time		40			ns
t_{AH}	Address 0–7 Hold Time		30			ns
t_{ALE}	ALE Strobe Width (High)		100			ns
t_{ARW}	ALE to Read or Write Strobe		75			ns
t_{CRW}	Chip Enable to Read or Write		100			ns
t_{RD}	Read Strobe Width		250			ns
t_{DDR}	Data Delay from Read			180	200	ns
t_{RDD}	Data Bus Disable				75	ns
t_{CH}	Chip Enable Hold After Read or Write		60			ns
t_{RWA}	Read or Write to Next ALE		45			ns
t_{WR}	Write Strobe Width		200	250		ns
t_{DS}	Data Set-Up Time		100			ns
t_{DH}	Data Hold Time		75			ns
MODEM						
t_{MD}	$\overline{\text{WR}}$ Command Reg. to Modem Outputs Delay			180		ns
t_{SIM}	Delay to Set Interrupt from Modem Input			200		ns
t_{RIM}	Delay to Reset Modem Status Interrupt from $\overline{\text{RD}}$			240		ns
t_{SMI}	$\overline{\text{WR}}$ to Status Mask Reg., Delay to $\overline{\text{RTI}}$				230	ns
POWER DOWN						
t_{PCS}	Power Down to All Clocks Stopped			1	2	$t_{BIT} + t_{XC}$
t_{PCR}	Power Down Removed to Clocks Running			1	2	$t_{BIT} + t_{XC}$
t_{PXS}	Power Down Removed to XTAL Oscillator Stable	When Using On Chip Inverter for Oscillator Circuit		100		ms
t_{PSE}	Power Down Set-Up to $\overline{\text{RD}}$ or $\overline{\text{WR}}$ Edge		160	260		ns
t_{EPI}	$\overline{\text{WR}}$ or $\overline{\text{RD}}$ Edge Following $\overline{\text{PD}}$ to Internal Signals	Enable or Disable		100		ns
BAUD GENERATOR						
t_{XH}	XTAL In High		100			ns
t_{XL}	XTAL In Low		100			ns
f_{BRC}	Baud Rate Clock Input Frequency				4.1	MHz
t_{BD1}	Baud Out Delay ÷ 1			160		ns
t_{BD2}	Baud Out Delay ÷ 2			200		ns
t_{BD3}	Baud Out Delay ÷ 3			200		ns
t_{BDN}	Baud Out Delay ÷ N > 3			200		ns
t_{XC}	Baud Clock Cycle	$t_{XC} = \frac{1}{f_{BRC}}$	243			ns

4.0 AC Electrical Characteristics (Continued)

Symbol	Parameter	Test Conditions	Min	Typ	Max	Units
TRANSMITTER						
t_{TCD}	TxD Delay from $\overline{\text{TxC}}$	External Clock		220		ns
		Internal Clock		140		ns
t_{TXC}	Cycle Time $\overline{\text{TxC}}$	16X, 32X, 64X Clock Factor	243			ns
		1X Clock Factor	1000			ns
t_{TCH}	$\overline{\text{TxC}}$ High		100			ns
t_{TCL}	$\overline{\text{TxC}}$ Low		100			ns
t_{HRI}	$\overline{\text{WR}}$ TxHR to Reset TxBE $\overline{\text{RTI}}$			260		ns
t_{HTS}	$\overline{\text{WR}}$ TxHR to TxD Start		2	3	4	t_{BIT}
t_{TSI}	Skew Start Bit to $\overline{\text{RTI}}$		−100	+20	+120	ns
t_{ETS}	Enable Tx to Start Bit		3	4	5	t_{BIT}
t_{BIT}[1]	One Bit Time	1X	1000			ns
		16X	3.88			μs
		32X	7.77			μs
		64X	15.55			μs
RECEIVER						
t_{RS}	RxD Set-Up	1X Clock Factor		160		ns
t_{RH}	RxD Hold	1X Clock Factor		100		ns
t_{RXC}	Cycle time $\overline{\text{RxC}}$	16X, 32X, 64X Clock Factor	243			ns
		1X Clock Factor	1000			ns
t_{RCH}	$\overline{\text{RxC}}$ High		100			ns
t_{RCL}	$\overline{\text{RxC}}$ Low		100			ns
t_{RRI}	$\overline{\text{RD}}$ to Reset $\overline{\text{RTI}}$			300		ns
t_{BIT}[1]	One Bit Time	1X	1000			ns
		16X	3.88			μs
		32X	7.77			μs
		64X	15.55			μs
t_{ERS}	Enable Rx to Correctly Detect Start Bit	All Clock Factors	2	3	4	t_{RXC}
t_{RNO}	Read RxHR Before Next Data; No OE		240			ns
t_{BI}	$\overline{\text{RxC}}$, Break to $\overline{\text{RTI}}$			340		
t_{REI}	Receiver Error Int			½ Clock Factor		t_{RXC}
t_{RDI}	Receiver Ready Int			$t_{REI}+1$		t_{RXC}
t_{RSI}	$\overline{\text{RxC}}$ to $\overline{\text{RTI}}$			300		ns
RESET TIMING						
t_{MR}	MR Pulse Width			100		ns
t_{RA}	MR to ALE if Valid $\overline{\text{WR}}$ or $\overline{\text{RD}}$ Cycle			100		ns

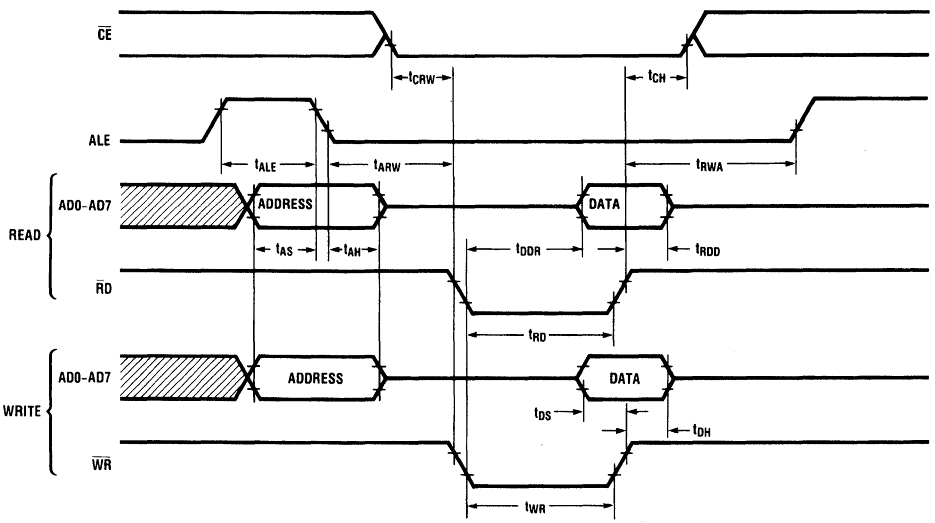
Note 1: $t_{BIT} = t_{TXC} \times$ Clock Factor (1, 16, 32, 64), transmitter

$t_{BIT} = t_{RXC} \times$ Clock Factor (1, 16, 32, 64), receiver

$$t_{BIT} = \frac{1}{\text{Baud Rate}}$$

5.0 Timing Waveforms

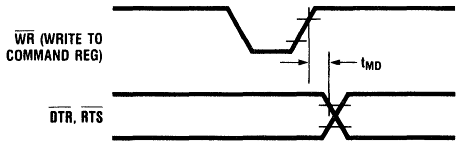
Read and Write Cycles



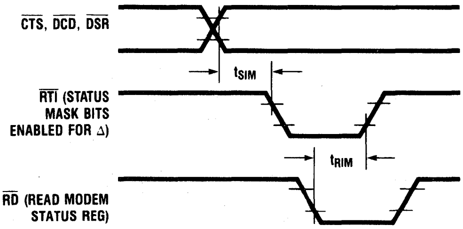
TL/C/5593-4

Note: The internal write is made inactive by either the next ALE or $\overline{\text{CE}}$ going invalid

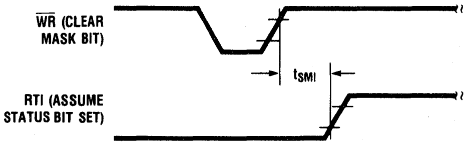
Modem Timing



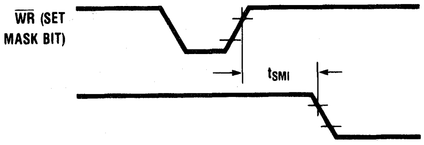
TL/C/5593-5


CTS, DCD, DSR
tSIM
RTI (STATUS MASK BITS ENABLED FOR Δ)
tRIM
RD (READ MODEM STATUS REG)

TL/C/5593-6



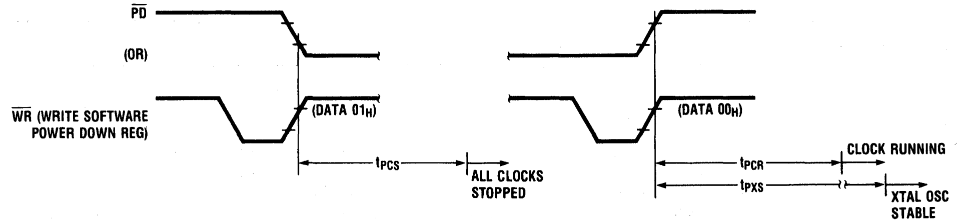
TL/C/5593-7



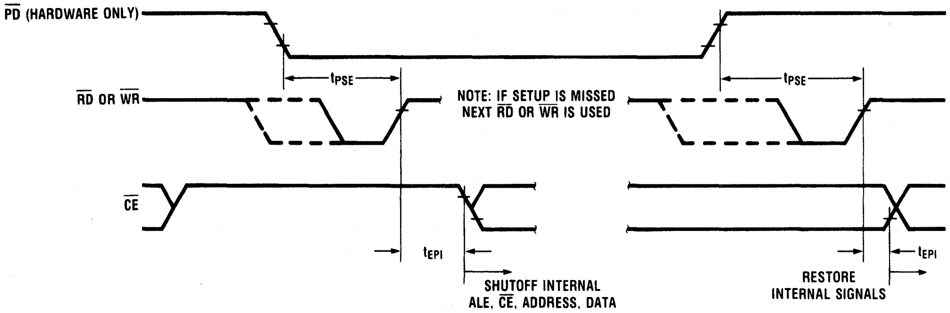
TL/C/5593-8

5.0 Timing Waveforms (Continued)

Power Down Timing

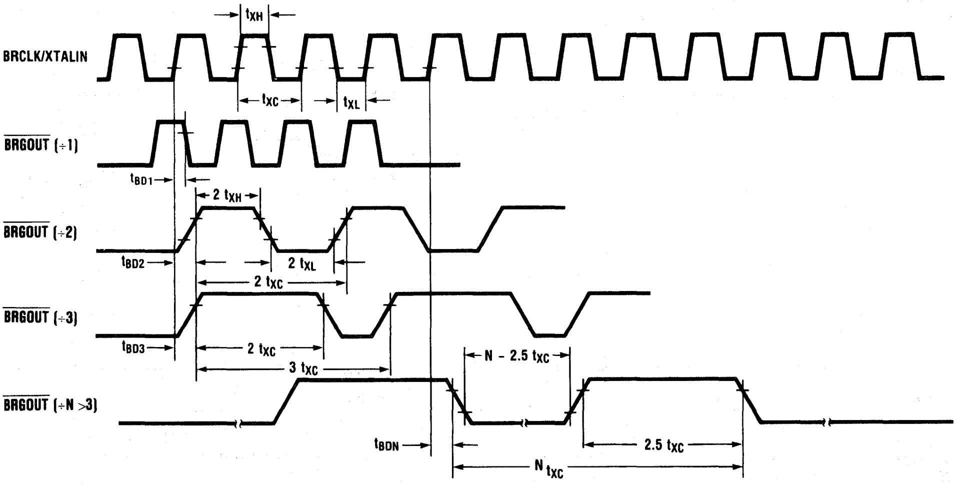


TL/C/5593-9


$\overline{PD}$ (HARDWARE ONLY)
t_{PSE}
t_{PSE}
$\overline{RD}$ OR $\overline{WR}$
NOTE: IF SETUP IS MISSED NEXT $\overline{RD}$ OR $\overline{WR}$ IS USED
$\overline{CE}$
t_{EPI}
SHUTOFF INTERNAL ALE, $\overline{CE}$, ADDRESS, DATA
t_{EPI}
RESTORE INTERNAL SIGNALS

TL/C/5593-10

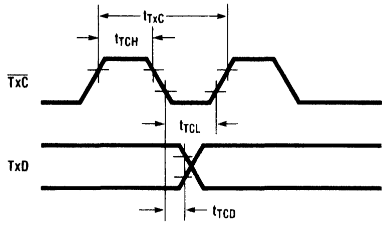
Baud Out Timing



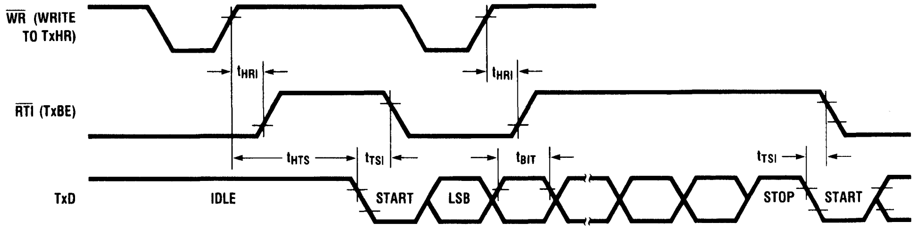
TL/C/5593-11

5.0 Timing Waveforms (Continued)

Transmitter Timing



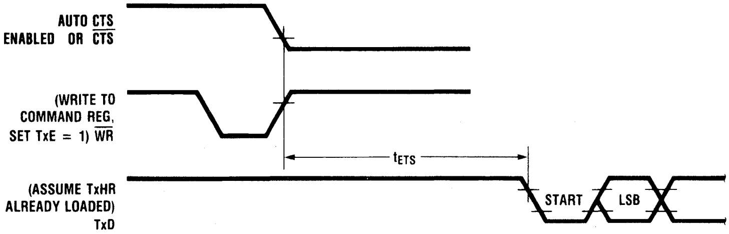
TL/C/5593-12



$$t_{BIT} = \frac{1}{\text{BAUD RATE}} = t_{TxC} \times \text{CLOCK FACTOR (1, 16, 32, 64)}$$

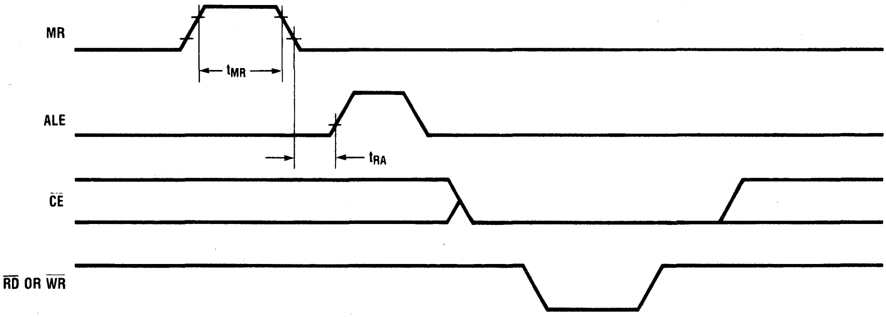
TL/C/5593-13

Note: The AC Timing Spec for RTI due to TXU or TBK will be published in the next data sheet.


AUTO CTS ENABLED OR CTS
(WRITE TO COMMAND REG, SET TxE = 1) WR
tETS
(ASSUME TxHR ALREADY LOADED) TxD
START
LSB

TL/C/5593-14

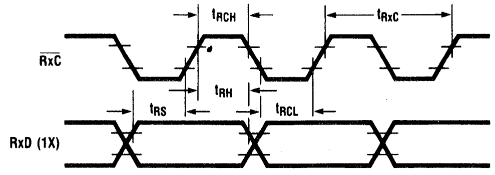
Reset Timing



TL/C/5593-15

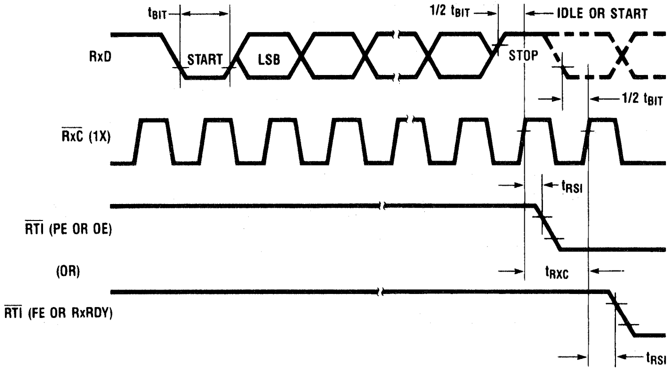
5.0 Timing Waveforms (Continued)

Receiver Timing

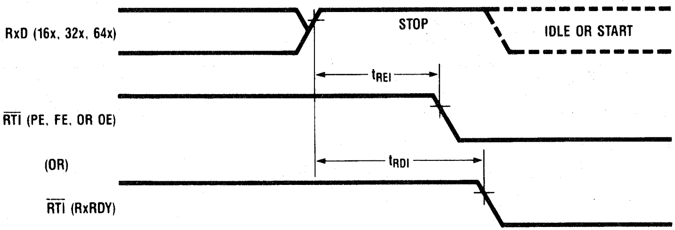


TL/C/5593-16

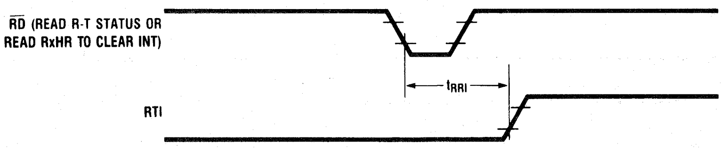
$$t_{BIT} = \frac{1}{\text{BAUD RATE}} = t_{RxC} \times \text{CLOCK FACTOR (1, 16, 32, 64)}$$



TL/C/5593-17



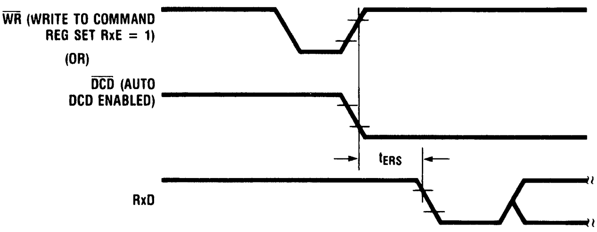
TL/C/5593-18



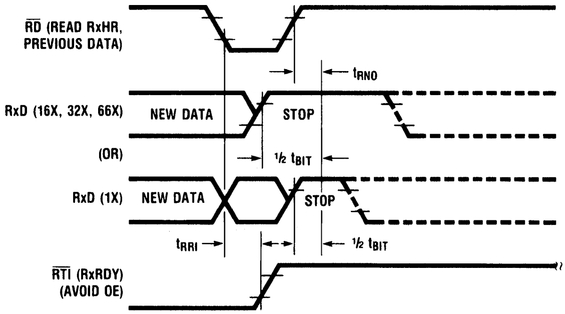
TL/C/5593-19

5.0 Timing Waveforms (Continued)

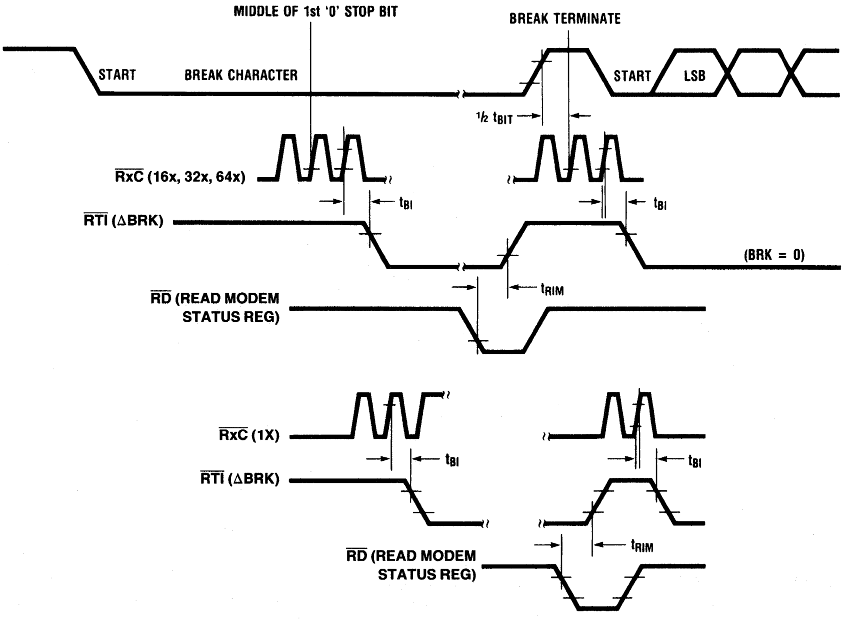
Receiver Timing (Continued)



TL/C/5593-20



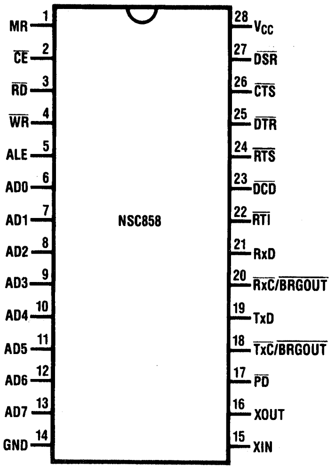
TL/C/5593-21



TL/C/5593-22

6.0 Connection Diagrams

Dual-In-Line Package

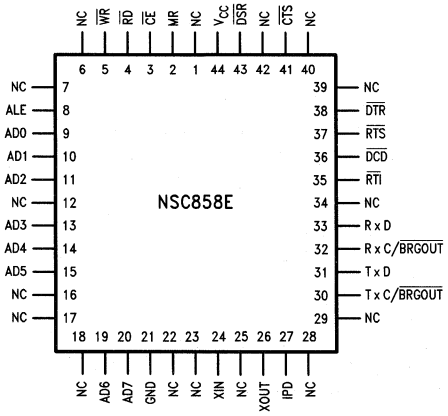


TL/C/5593–23

Top View

Order Number NSC858D or N
See NS Package D28C or N28B

Leadless Chip Carrier

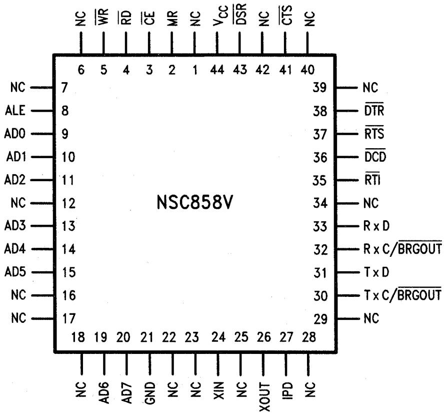


TL/C/5593–24

Top View

Order Number NSC858E
See NS Package Number E44A

Plastic Chip Carrier



TL/C/5593–25

Top View

Order Number NSC858V
See NS Package Number V44A

7.0 Pin Descriptions

7.1 INPUT SIGNALS

Master Reset (MR): active high, Pin 1. This Schmitt trigger input has a 0.5V typical hysteresis. When high, the following registers are cleared: receiver mode, transmitter mode, global mode, R-T status (except for TxBE which is set to one), R-T status mask, modem mask, command (which disables receiver "Rx" and the transmitter "Tx"), power down, and receiver holding. In the modem status register, ΔCTS, ΔDCD, ΔDSR, BRK and ΔBRK are cleared.

Chip Enable ($\overline{\text{CE}}$): active low, Pin 2. Chip enable must be low during a valid read or write pulse in order to select the device. Chip enable is not latched.

Read ($\overline{\text{RD}}$): active low, Pin 3. While the chip is enabled the CPU latches data from the selected register on the rising edge of $\overline{\text{RD}}$.

Write ($\overline{\text{WR}}$): active low, Pin 4. While the chip is enabled it latches data from the CPU on the rising edge of $\overline{\text{WR}}$.

Address Latch Enable (ALE): negative edge sensitive, Pin 5. The negative edge (high to low) of ALE latches the address for the register select during a read or write operation.

7.0 Pin Descriptions (Continued)

Power Down ($\overline{\text{PD}}$): active low, Pin 17. When active it disables all internal clocks, shuts off the oscillator, clears RxE, TxE, and break control bits in the command register. All other registers retain their data. Unlike software power down, $\overline{\text{PD}}$ also disables the internal ALE, $\overline{\text{CE}}$, $\overline{\text{RD}}$, $\overline{\text{WR}}$, address and data paths for minimum power consumption. Registers cannot be accessed in hardware power down; they may be in software power down.

Receiver Data (RxD): Pin 21. This accepts serial data input from the communications link (peripheral device, modem, or data set). Serial data is received least significant bit (LSB) first. "Mark" is high (1), "space" is low (0).

Data Carrier Detect ($\overline{\text{DCD}}$): active low, Pin 23. Can be used as a modem or general purpose input. When this modem input is low it indicates that the data carrier has been detected by the modem or data set. The $\overline{\text{DCD}}$ signal is a modem control function input whose complement value can be tested by the CPU by reading bit 5 (DCD) of the modem status register. Bit 1 (ΔDCD) of the modem status register indicated whether the $\overline{\text{DCD}}$ input has changed state since the previous reading of the modem status register. $\overline{\text{DCD}}$ can also be programmed to become an auto enable for the receiver.

NOTE: Whenever the DCD bit of the modem status register changes state, an interrupt is generated if the ΔDCD mask and the DSCHG mask bits are set.

Clear to Send ($\overline{\text{CTS}}$): active low, Pin 26. Can be used as a modem or a general purpose input. The $\overline{\text{CTS}}$ inputs complement can be tested by the CPU by reading bit 4 (CTS) of the modem status register. Bit 0 (ΔCTS) of the modem status register indicates whether the $\overline{\text{CTS}}$ input has changed state since the previous reading of the modem status register. $\overline{\text{CTS}}$ can be programmed to automatically enable the transmitter. Note: Whenever the CTS bit of the modem status register changes state, an interrupt is generated if the ΔCTS mask and the DSCHG mask bits are set.

Data Set Ready ($\overline{\text{DSR}}$): active low, Pin 27. Can be used as a modem or a general purpose input. When this modem input is low it indicates that the modem or data set is ready to establish the communication link and transfer data with the NSC858. The $\overline{\text{DSR}}$ is a modem-control function input whose complement value can be tested by the CPU by reading bit 6 (DSR) of the modem status register. Bit 2 (ΔDSR) of the modem status register indicates whether the ($\overline{\text{DSR}}$) input has changed state since the previous reading of the modem status register.

NOTE: Whenever the DSR bit of the modem status register changes state, an interrupt is generated if ΔDSR mask and the DSCHG mask bits are set.

Power (V_{CC}): Pin 28. +5V supply.

Ground (GND): Pin 14. Ground (0V) supply.

7.2 OUTPUT SIGNALS

Transmit Data (TxD): Pin 19: Composite serial data output to the communication link (peripheral, modem or data set) least significant bit first. The TxD signal is set to the marking (logic 1) state upon a master reset. In hardware or software power down this pin will always be a one.

Receiver-Transmitter Interrupt ($\overline{\text{RTI}}$): active low, Pin 22. Goes low when any R-T status register bit and its corresponding mask bit are set. This bit can change states during either hardware or software power down due to a change in modem status information.

Request to Send ($\overline{\text{RTS}}$): active low, Pin 24. Can be used as a modem or a general purpose output. When this modem output is low it informs the modem or data set that the NSC858 is ready to transmit data. The $\overline{\text{RTS}}$ output or general purpose output signal can be set to an active low by programming bit 6 of the command register with a 1. The $\overline{\text{RTS}}$ signal is set high upon a master reset operation. During remote loopback $\overline{\text{RTS}}$ signal reflects the complement of bit 6 of the command register. During local loopback the $\overline{\text{RTS}}$ signal is forced to its inactive state (high). $\overline{\text{RTS}}$ cannot change states during hardware power down; it can during software power down.

Data Terminal Ready ($\overline{\text{DTR}}$): active low, Pin 25. Can be used as a modem or general purpose output. When this modem output is low it informs the modem or data set that the NSC858 is ready to communicate. The $\overline{\text{DTR}}$ output or the general purpose output signal can be set to an active low by programming bit 7 of the command register with a 1. The $\overline{\text{DTR}}$ signal is set high upon a master reset operation. During remote loopback $\overline{\text{DTR}}$ signal reflects the complement of bit 7 of the command register. During local loopback the $\overline{\text{DTR}}$ signal is forced to its inactive state (high). $\overline{\text{DTR}}$ signal cannot change state during hardware power down; it can during software power down.

7.3 INPUT/OUTPUT SIGNALS

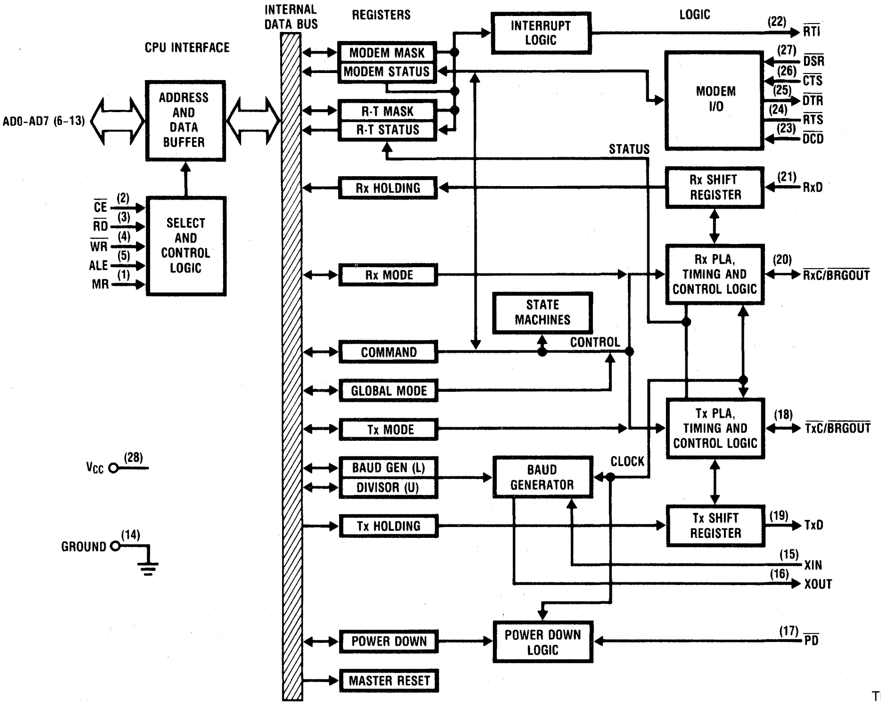
Address/Data Bus (AD0–AD7): Pins 6–13. The multiplexed bidirectional address/data bus, AD0–AD7 pins, are in the high impedance state when the NSC858 is not selected or whenever it is in hardware power down. AD0–AD3 are latched on the trailing edge of ALE, providing the four address inputs. The rising edge of the $\overline{\text{WR}}$ input enables 8 bits to be written in, through AD0–AD7, to the addressed register. $\overline{\text{RD}}$ input enables 8 bits to be read from a register out through AD0–AD7.

Transmitter Clock/Baud Rate Generator Output ($\overline{\text{TxC}}$/$\overline{\text{BRGOUT}}$): Pin 18. If the transmitter is programmed for an external clock, $\overline{\text{TxC}}$ is an input. If the transmitter is programmed for an internal clock, then the Baud Rate Generator is used for the transmitter, and it is output at $\overline{\text{TxC}}$/$\overline{\text{BRGOUT}}$. In either case, $\overline{\text{TxC}}$/$\overline{\text{BRGOUT}}$ signal is running at 1X, 16X, 32X, 64X the data rate, as selected by the clock factor. If this pin is used as an output it will be set to a zero (0) in both hardware and software power down.

Receiver Clock/Baud Rate Generator Output ($\overline{\text{RxC}}$/$\overline{\text{BRGOUT}}$): Pin 20. If the receiver is programmed for an external clock, RxC is an input. If the receiver is programmed for an internal clock, the Baud Rate Generator is used for the receiver, and it is output at $\overline{\text{RxC}}$/$\overline{\text{BRGOUT}}$. In either case, $\overline{\text{RxC}}$/$\overline{\text{BRGOUT}}$ signal is running at 1X, 16X, 32X, 64X, the data rate as selected by the clock factor. If this pin is programmed as an output it will be set to one (1) in both hardware and software power down.

Crystal (XIN, XOUT): Pins 15, 16. These two pins connect the main timing reference. A crystal network can be connected across these two pins, or a square wave can be driven into XIN with XOUT left floating. In hardware and software power down XOUT is set to a 1. Ground XIN when using both RxC and TxC to supply external clocks to the UART.

8.0 Block Diagram



TL/C/5593-26

FIGURE 1. NSC858 Functional Block Diagram

9.0 Registers

The system programmer may access control of any of the NSC858 registers summarized in Table I via the CPU. These 8-bit registers are used to control NSC858 operation and to transmit and receive data.

TABLE I. Register Address Designations

Address				Register	Read/ Write
A_3	A_2	A_1	A_0		
0	0	0	0	Rx Holding	R
0	0	0	0	Tx Holding	W
0	0	0	1	Receiver Mode	R/W
0	0	1	0	Transmitter Mode	R/W
0	0	1	1	Global Mode	R/W
0	1	0	0	Command	R/W
0	1	0	1	Baud Rate Generator Divisor Latch (Lower)	R/W
0	1	1	0	Baud Rate Generator Divisor Latch (Upper)	R/W
0	1	1	1	R-T Status Mask	R/W
1	0	0	0	R-T Status	R
1	0	0	1	Modem Status Mask	R/W
1	0	1	0	Modem Status	R
1	0	1	1	Power Down	R/W
1	1	0	0	Master Reset	W

Note: Offset address OD, OE, OF are unused.

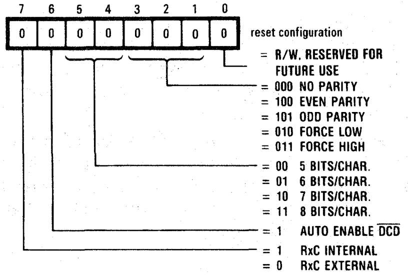
9.1 RECEIVER AND TRANSMITTER HOLDING REGISTER

A read to offset location 00 will access the Receiver holding register; a write will access the Transmitter holding register.

9.2 RECEIVER MODE REGISTER

The system programmer specifies the data format of the receiver (which may differ from the transmitter) by programming the Receiver mode register at offset location "01." This read/write register programs the parity, bits/character, auto enable option, and clock source. When bit 6 of this register is set high the receiver will be enabled any time the $\overline{\text{DCD}}$ signal input is low (provided CR0 = 1). When bit 7 is set to a "1" the receiver clock source is the internal baud rate generator and $\overline{\text{RxC}}$ is then an output. After reset this register is set to "00."

TABLE II. Receiver Mode Register (Address "01") (Bits RMO-7)



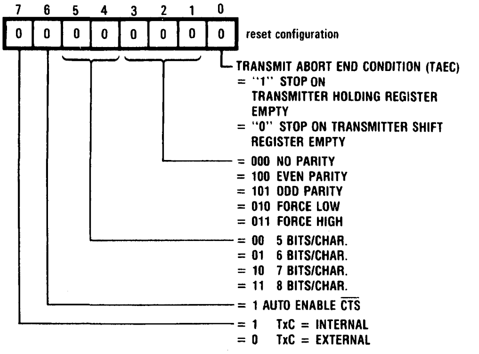
TL/C/5593-27

9.0 Registers (Continued)

9.3 TRANSMITTER MODE REGISTER

The system programmer specifies the data format of the transmitter (which may differ from the receiver) by programming the transmitter mode register at offset location "02."

TABLE III. Transmit Mode Register (Address "02") (Bits TM0–7)



TL/C/5593–28

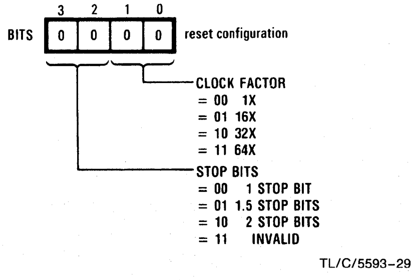
The transmitter mode register is similar in operation to the receiver mode register except for the addition of the Transmit Abort End Condition (TAEC). If this bit is set to a one when a request to disable the transmitter or send a break is pending then the data in the shift register and holding register will be transmitted prior to such action occurring. If TAEC equals 0 then the action will take place after the shift register has been emptied. When bit 6 of this register is set high the transmitter will be enabled any time the $\overline{CTS}$ signal is low (provided CR1 = 1). When bit 7 is set to a "1" the transmitter clock source is the internal baud rate generator, and $\overline{TxC}$ is then an output. After reset this register is set to "00."

9.4 GLOBAL MODE REGISTER

This register is used to program the number of stop bits and the clock factor for both the receiver and transmitter. Only the lower four bits of this register are used, the upper four can be programmed as don't cares and they will be read back as zeros. Programming the number of stop bits is for the transmitter only; the receiver always checks for one stop bit. If a 1X clock factor with 1.5 stop bits is selected for the transmitter the number of stop bits will default to 1. After reset this register is set to "00."

Note: Selecting the 1x clock requires that the clock signal be sent or received along with the data.

TABLE IV. Global Mode Register (Address "03") (Bits GM0–3)



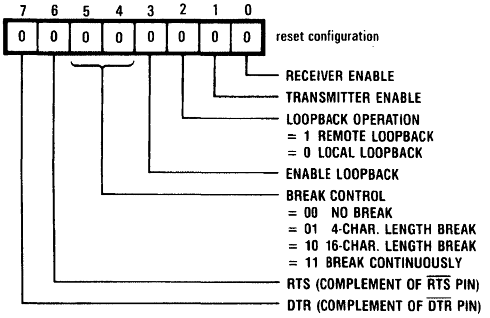
TL/C/5593–29

Bits 4–7 are don't care, read as 0s.

9.5 COMMAND REGISTER

The Command register is an eight bit read/write register which is accessed at offset location "04." After reset the command register equals "00."

TABLE V. Command Register (Address "04") (Bits CR0–7)



TL/C/5593–30

Bit 0: Receive Enable, when set to a one the receiver is enabled. If auto enable for the receiver has been programmed then in addition to CR0 = 1, the $\overline{DCD}$ input must be low to enable receiver.

Bit 1: Transmitter Enable, when set to a one the transmitter is enabled. If auto enable for the transmitter is programmed then in addition to CR1 = 1, the $\overline{CTS}$ input must be low to enable transmitter.

Bit 2: A zero selects local loopback and a one selects remote loopback.

Bit 3: A one enables either of the diagnostic modes selected in bit 2 of the command register.

Bits 4 and 5: Bits 4 and 5 of the command register are used to program the length of a transmitted break condition. A continuous break must be terminated by the CPU, but the 4 and 16 character length breaks are self clearing. (At the beginning of the last break character bits 4 and 5 will automatically be reset to 0.) Break commands affect the status of bit 6 (TBK) of the R-T Status register (see R-T Status register). Break control bits are cleared by software or hardware power down.

Bits 6 and 7: These two bits control the status of the output pins $\overline{RTS}$ (pin 24) and $\overline{DTR}$ (pin 25) respectively. They may be used as modem control functions or be used as general purpose outputs. The output pins will always reflect the complement of the register bits.

9.6 R-T STATUS REGISTER

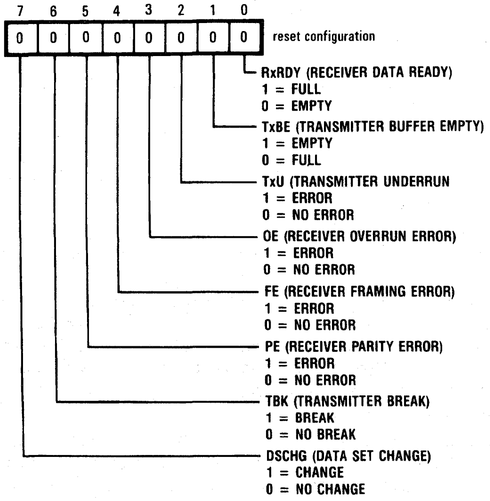
This 8-bit register contains status information of the NSC858 and therefore is a read only register at offset location "08." Each bit in this register can generate an interrupt ($\overline{RTI}$). If any bit goes active high and its associated mask bit is set then the $\overline{RTI}$ will go low. $\overline{RTI}$ will be cleared when all unmasked R-T Status bits are cleared. Bits 0 and 1, receiver ready and transmitter empty are cleared by reading the receiver holding register or writing the transmitter holding register respectively. Bits 2 through 5, transmit underrun, receiver overrun, framing error, parity error are cleared by reading the R-T Status register. Bit two, transmitter underrun will occur when both the transmit holding register and the transmit shift register are empty.

9.0 Registers (Continued)

Bit three, overrun error, will occur when the CPU does not read a character before the next one becomes available. The OE bit informs the programmer or CPU that RXHR data has been overrun or overwritten. The byte in the shift register is always transferred to the holding register, even after an overrun occurs. If an OE occurs, it is standard protocol to request a re-transmission of that block of data. A read of RXHR, when a subsequent read of R-T status shows that no OE is present, indicates current receiver data is available. Bit four, framing error, occurs when a valid stop bit is not detected. Bit 5 is set when a parity error is detected. Bits three, four and five are affected by the receiver only.

Bit 6, Transmit Break (TBK) is set at the beginning of each break character during a break continuously command, or at the beginning of the final break character in a 4 or 16 character programmed break length. It is cleared by reading the R-T Status register. Bit 7, Data Set Change (DSCHG) will be set whenever any of the bits 0–3 of the Modem Status register and their associated mask bit are set. Data Set Change bit is cleared by reading the Modem Status register or is masked off by writing "0" to all modem register bits. After reset the R-T Status register equals '02', i.e. all bits except TxBE are reset to zero.

TABLE VI. R-T Status Register (Address "08") (Bits SR0–7)



TL/C/5593–31

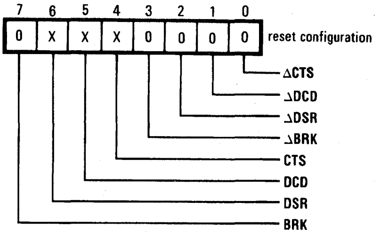
9.7 R-T STATUS MASK REGISTER (SM0–7)

This register is used in conjunction with the R-T Status register to enable or disable conditional interrupts A one in any bit unmasks its associated bit in the R-T Status register, and allows it to generate an interrupt out through $\overline{\text{RTI}}$. The mask affects only the interrupt and not the R-T Status bits. This eight bit register is both read and writable at offset location "07." After reset it is set to "0" which disables all interrupts. Each bit in the R-T Status mask register is associated with that bit in the R-T Status register (e.g., SM0 is SR0's mask).

9.8 MODEM STATUS

This eight bit read only register which is addressed at offset location "0A" contains modem or general purpose input and receiver break information.

TABLE VII. Modem Status Register (Address "0A") (Bits MS0–7)



TL/C/5593–32

Each of the four status signals in this register also have an associated delta bit in this register. Each delta bit (bits MS0–3) will be set when its corresponding bit changes states. These four delta bits are cleared when the Modem Status register is read. If any of these four delta bits and associated mask bits are set they will force DSCHG (bit 7) of the R-T Status register high. Bits 4–6, CTS, DCD, DSR can be used as modem signals or general purpose inputs. In either case the value in the register represents the complements of the input pins $\overline{\text{CTS}}$ (pin 26), $\overline{\text{DCD}}$ (pin 23), and $\overline{\text{DSR}}$ (Pin 27). Bit 7 (BRK) when set to a one indicates that the receiver has detected a break condition. It is cleared when break terminates. After reset ΔCTS, ΔDCD, ΔDSR, ΔBRK and BRK are cleared.

9.9 MODEM MASK REGISTER (MM0–3)

This 4-bit read/write register, which is addressed at offset location "09," contains mask bits for the four delta bits of the Modem Status register (MS0–3). A one ("1") in any of three bits and a one in the associated delta bit of the Modem Status register will set the DSCHG bit of the R-T Status register. Modem Mask bit 0 is associated with Modem Status bit 0, etc. The four (4) most significant bits of this register will read as zeros. After reset the register equals '00'.

9.10 POWER DOWN REGISTER (PD0)

This one bit register can both be read and written at offset location "0B." When bit zero is set to a one the NSC858 will be put into software power down. This disables the receiver and transmitter clocks, shuts off the baud rate generator and crystal oscillator, and clears the RxE, TxE, and break control bits in the command register. Registers on chip can still be accessed by the CPU during software power down. Bits 1 through 7 will always read as 0.

9.11 MASTER RESET REGISTER

This write only register is addressed at offset location "0C." When writing to this register the data can be any value (don't cares). Resetting the NSC858 by way of the reset register is functionally identical to resetting it by the MR pin.

9.12 BAUD RATE GENERATOR DIVISOR LATCH

These two 8-bit read/write registers which are accessed at offset locations "05" (lower) and "06" (upper) are used to program the baud rate divisor. These registers are not affected by the reset function and are powered up in a random state.

10.0 Functional Description

10.1 PROGRAMMABLE BAUD GENERATOR

The NSC858 contains a programmable Baud Generator that is capable of taking any clock input (DC to 4.1 MHz) and dividing it by any divisor from 1 to ($2^{16}-1$). The output frequency of the Baud Generator (available at $\overline{\text{TxC}}/\overline{\text{BRGOUT}}$ or $\overline{\text{RxC}}/\overline{\text{BRGOUT}}$, if internal $\overline{\text{TxC}}$ or $\overline{\text{RxC}}$ is selected) is equal to the clock factor (1X, 16X, 32X, 64X) times the baud rate. The divisor number is determined by the following equation:

$$\text{divisor \#} = \frac{\text{Frequency Input } (f_{BRC})}{[\text{Baud Rate} \times \text{Clock Factor } (1, 16, 32, 64)]}$$

Two 8-bit latches store the divisor in a 16-bit binary format. These Divisor Latches must be loaded during initialization in order to ensure desired operation of the Baud Generator. Upon loading either of the Divisor Latches, a 16-bit Baud counter is immediately loaded. This prevents long counts on initial load.

Tables VIII and IX illustrate the use of the Baud Generator with crystal frequencies of 1.8432 MHz and 3.072 MHz respectively. For baud rates of 38400 and below, the error obtained is minimal. The accuracy of the desired baud rate is dependent on the crystal frequency chosen.

TABLE VIII. Baud Rates Using 1.8432 MHz Crystal

Desired Baud Rate	Divisor Used To Generate 16 x Clock	Percent Error Difference Between Desired and Actual
50	2304	—
75	1536	—
110	1047	0.026
134.5	857	0.058
150	768	—
300	384	—
600	192	—
1200	96	—
1800	64	—
2000	58	0.69
2400	48	—
3600	32	—
4800	24	—
7200	16	—
9600	12	—
19200	6	—
38400	3	—
56000	2	2.86

TABLE IX. Baud Rates Using 3.072 MHz Crystal

Desired Baud Rate	Divisor Used To Generate 16 x Clock	Percent Error Difference Between Desired and Actual
50	3840	—
75	2560	—
110	1745	0.026
134.5	1428	0.034
150	1280	—
300	640	—
600	320	—
1200	160	—
1800	107	0.317
2000	96	—
2400	80	—
3600	53	0.628
4800	40	—
7200	27	1.23
9600	20	—
19200	10	—
38400	5	—

10.2 RECEIVER AND TRANSMITTER OPERATION

The NSC858 transmits and receives data in an asynchronous communications mode. The CPU must set up the appropriate mode of operation, number of bits per character, parity, number of stop bits, etc. Separate mode registers exist for the independent specification of receiver and transmitter operation. These independent specifications include parity, character length, and internal or external clock source. Only the Global Mode Register, which controls the number of stop bits and the clock factor, exercises common control over the receiver and transmitter (receiver looks for only one stop bit).

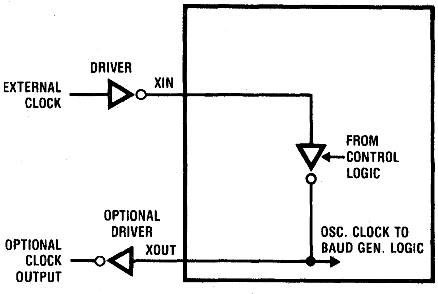
10.3 TRANSMITTER OPERATION

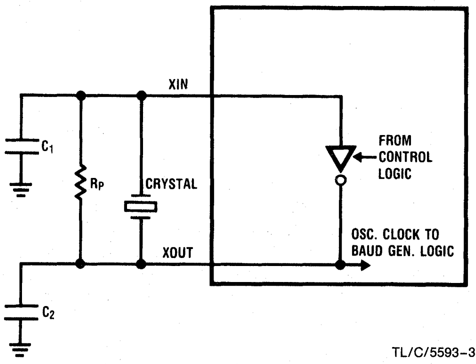
The Transmitter Holding register is loaded by the CPU. To enable the transmitter, TxE must be set in the Command register. $\overline{\text{CTS}}$ must be low if the auto enable is set in the Tx Mode register. The Transmitter Holding register is then parallel loaded into the Transmitter Shift register, and the start bit, parity bit and the specified number of stop bits are inserted. This serialized data is available at the TxD output pad, and changes on the rising edge of $\overline{\text{TxC}}$, or equivalently the falling edge of $\overline{\text{TxC}}$. The TxD output remains in a mark ("1") condition when no data is being transmitted, with the exception of sending a break ("0").

A break condition is initiated by writing either a continuous or specified length break request to the Command Register. A finite break specification of either 4 or 16 character lengths can be extended by re-writing the break command before the specified break length is completed. Each break character is transmitted as a start bit, logical zero data, logical zero parity (if specified) and logical zero stop bit(s). The number of data and stop bits, plus the presence of a parity bit are determined by the Transmitter and Global Mode registers. Thus, the total number of (all zero) bits in a break character is the same as that for data. The break is terminated by writing "00" to the Break Control bits in the Command Register. The Set Break bits in the Command register are always reset to "00" after the termination of the specified break transmission or if the transmitter is disabled during a break transmission. The TxD output will always return to a mark condition for at least one bit time before transmitting a character after a break condition. Data in the Transmitter Holding register, whether loaded before (on TAEC=0) or during the break will be transmitted after the break is terminated.

10.0 Functional Description (Continued)

10.4 TYPICAL CLOCK CIRCUITS

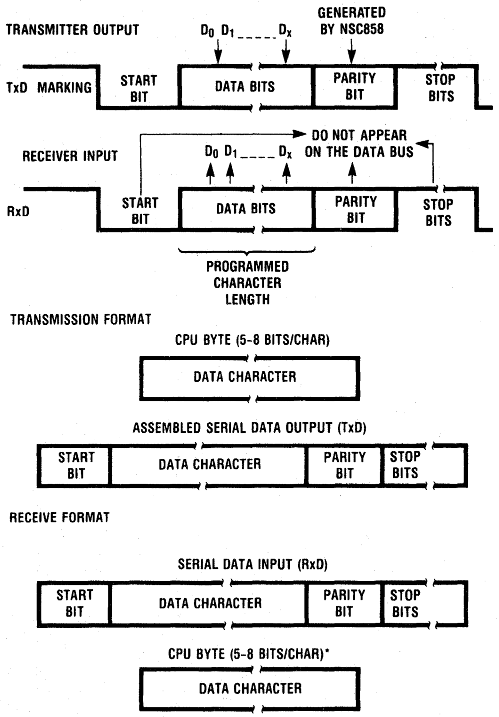




TL/C/5593-33

CRYSTAL	R_P	C_1	C_2
3.1 MHz	1 MΩ	10–30 pF	40–60 pF
1.8 MHz	1 MΩ	10–30 pF	40–60 pF

FIGURE 2. Typical Crystal Oscillator Network



TL/C/5593-34

Note: If character length is defined as 5, 6 or 7 bits, the unused bits are set to "0").

FIGURE 3

10.5 RECEIVER OPERATION

The NSC858 receives serial data on the RxD input. To enable the receiver, DCD must be low if the DCD Auto Enable bit in the Receiver Mode register is set ("1"). RxE must be set in the Command register. RxD is sampled on the falling edge of RxC or equivalently on the rising edge of RxC. If a high ("1") to low ("0") transition of RxD is detected, RxD is sampled again, for all except the 1X clock factor, at ½ of a bit time later. If RxD is still low, then a valid start bit has been received and character assembly proceeds. If RxD has returned high, then a valid start bit has not been received, and the search for a valid start bit continues. When a character has been assembled in the Receiver Shift Register and transferred to the Receiver Holding Register, the RxRDY bit (and any error bits that may have occurred) in the R-T Status register will be set and RTI will go low (if the proper mask bits are set). After the CPU reads the Receiver Holding register, the RxRDY will go low and the RTI will go inactive ("1").

The receiver will detect a break condition on RxD if an all zero character with zero parity bit (if parity is specified) and a zero stop bit is received. For the break condition to terminate, RxD must be high for one half a bit time. If a break condition is detected, bits 3 and 7 in the Modem Status register (ΔBRK and BRK respectively) will be set. Bit 3 (ΔBRK) will then cause bit 7 (DSCHG) in the R-T Status register to be set which in turn forces RTI to an asserted state ("0"). These interrupts will occur only if the appropriate mask bits are set for the registers in question.

When the 1x clock factor is selected:

The RxC pin on the NSC858 should be connected to the clock signal of the incoming data stream and bit 7 of the receiver mode register should be cleared to A0.

The TxC output of the NSC858 does not have to be sent to the remote receiver unless the receiver is using a 1x clock factor.

10.6 PROGRAMMING THE NSC858

There are two distinct steps in programming the 858. During initialization, the modes, clocks, masks and commands are set up. Then, in operation, Modem I/O takes place, status is monitored, the receiver and transmitter are run as needed.

To initialize the 858, first pulse the MR line or write to the Master Reset register. Then, write to the following registers in any order, except for enabling the Rx and Tx, which must

10.0 Functional Description (Continued)

be at the end of the set up procedure. The Global, Receiver and Transmitter Mode registers determine the modes for the Rx and Tx. These latter two registers often will have the same data byte written to them, but are kept independent for flexibility. If the mode registers indicate that the receiver and/or the transmitter use an internal clock, then data (determined by the crystal frequency and desired bit time and clock factor) should be written to the upper and lower Baud Rate Generator Divisor Latches. The Modem Status Mask register enables Data Set change in R-T Status. If interrupts are required, the R-T Status Mask register allows $\overline{\text{RTI}}$ to occur. Write to the Command register to enable the receiver and/or transmitter only when all else is set up.

In operation, the 858 can transmit, receive and handle I/O simultaneously. Modem outputs are written to at the Command register, while the inputs are read at the Modem Status register. Data flow and errors are read at the R-T Status register. When serial data has been shifted in and assembled, the receiver is ready, and the word can be read at the Rx Holding register. When the transmitter buffer is empty, the Tx Holding register can be written to, and the word will be shifted out as serial asynchronous data.

Once the 858 is running, several options may be exercised. Masks can be changed at any time. The Rx and Tx are disabled or enabled, as needed, by writing to the Command register, or toggling the auto enable modem inputs (if used). Both the Rx and Tx should be disabled before either altering any mode or engaging a loopback diagnostic, and they can be re-enabled then or at a later time. Power down is allowed at any time except during loopback, although data may be lost if PD occurs in the middle of a word.

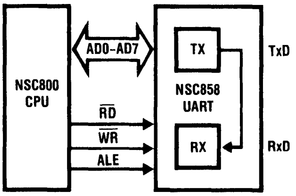
Thus, software for the NSC858 is of two types. The initialization routine is performed once. The operation routines, usually incorporating polling or interrupts, are then run continuously or on demand, depending upon the system or application.

10.7 DIAGNOSTIC CAPABILITIES

The NSC858 offers both remote and local loopback diagnostic capabilities. These features are selected through the Command register.

Local Loopback Mode (see *Figure 4*)

1. The transmitter output is internally connected to the receiver input.
2. $\overline{\text{DTR}}$ is internally connected to $\overline{\text{DCD}}$, and $\overline{\text{RTS}}$ is internally connected to $\overline{\text{CTS}}$.
3. $\overline{\text{TxC}}$ is internally connected to $\overline{\text{RxC}}$.
4. The DSR is internally held low (inactive).
5. The TxD, $\overline{\text{DTR}}$ and $\overline{\text{RTS}}$ outputs are held high.
6. The $\overline{\text{CTS}}$, $\overline{\text{DCD}}$, $\overline{\text{DSR}}$ and $\overline{\text{RxD}}$ inputs are ignored.
7. Except as noted, all other Status, Mode and Command Register bits and interrupts retain their functions and settings.



TL/C/5593-35

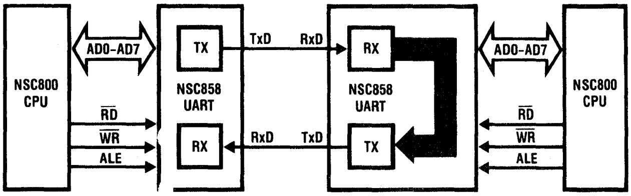
FIGURE 4. Local Loopback

Remote Loopback Mode (see *Figure 5*)

1. The contents of the Receiver Holding Register, when RxRDY = 1 indicates it is full, are transferred to the Transmitter Holding register, when TxBE = 1 indicates it is empty. After this action, both RxRDY and TxBE are cleared.
2. $\overline{\text{RxC}}$ is connected internally to $\overline{\text{TxC}}$.
3. Setting the Remote Loopback Mode places all receiver and transmitter flags under control of the remote loopback sequencer. RxRDY and TxBE can be monitored to follow automatic remote loopback data flow, while OE and TxU can indicate system problems.
4. The CPU can read the Receiver Holding register if desired, but this is not necessary. The CPU cannot load the Transmitter Holding Register.
5. Modem Status, all Mode and Command register bits retain their functions and interrupts are generated.

Under certain conditions entering the remote loopback mode causes a character in the receiver or transmitter holding registers to be sent, even though, the transmitter is disabled.

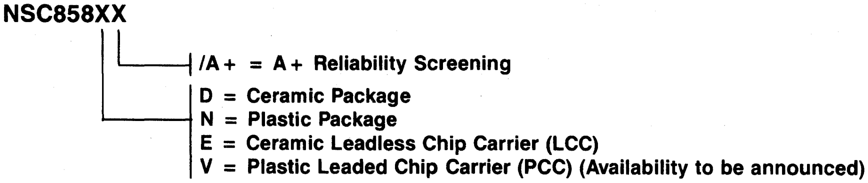
1. If the UART enters the remote loopback mode immediately after receiving a break character in the normal receive mode, it will then automatically transmit that character.
2. If the UART enters the remote loopback mode before the CPU has read the latest character in the receiver holding register, it will then automatically transmit that character.
3. If the UART enters the remote loopback mode before the last character written to the transmitter holding register is transmitted, then it will automatically transmit this character.



TL/C/5593-36

FIGURE 5. Remote Loopback

11.0 Ordering Information



NSC858XX

/A+ = A+ Reliability Screening

D = Ceramic Package

N = Plastic Package

E = Ceramic Leadless Chip Carrier (LCC)

V = Plastic Leaded Chip Carrier (PCC) (Availability to be announced)

TL/C/5593-37

12.0 Reliability Information

Gate Count 4280

Transistor Count 8450

National Semiconductor Corporation

NS16550A Universal Asynchronous Receiver/Transmitter with FIFOs†

General Description

The NS16550A is an improved version of the NS16450 Universal Asynchronous Receiver/Transmitter (UART). The improved specifications ensure compatibility with the NS32532 and other state-of-the-art CPUs. Functionally identical to the NS16450 on powerup (CHARACTER mode)* the NS16550A can be put into an alternate mode (FIFO mode) to relieve the CPU of excessive software overhead.

In this mode internal FIFOs are activated allowing 16 bytes (plus 3 bits of error data per byte in the RCVR FIFO) to be stored in both receive and transmit modes. All the logic is on chip to minimize system overhead and maximize system efficiency. Two pin functions have been changed to allow signalling of DMA transfers.

The UART performs serial-to-parallel conversion on data characters received from a peripheral device or a MODEM, and parallel-to-serial conversion on data characters received from the CPU. The CPU can read the complete status of the UART at any time during the functional operation. Status information reported includes the type and condition of the transfer operations being performed by the UART, as well as any error conditions (parity, overrun, framing, or break interrupt).

The UART includes a programmable baud rate generator that is capable of dividing the timing reference clock input by divisors of 1 to ($2^{16}-1$), and producing a 16 × clock for driving the internal transmitter logic. Provisions are also included to use this 16 × clock to drive the receiver logic. The UART has complete MODEM-control capability, and a processor-interrupt system. Interrupts can be programmed to the user's requirements, minimizing the computing required to handle the communications link.

The UART is fabricated using National Semiconductor's advanced scaled N-channel silicon-gate MOS process, XMOS.

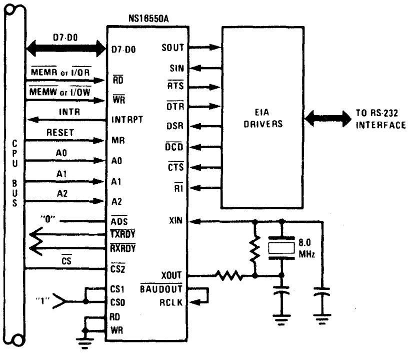
*Can also be reset to NS16450 Mode under software control.

†**Note: This part has a patent pending.**

Features

- Capable of running all existing 16450 software.
- Pin for pin compatible with the existing 16450 except for CSOUT (24) and NC (29). The former CSOUT and NC pins are TXRDY and RXRDY, respectively.
- After reset, all registers are identical to the 16450 register set.
- In the FIFO mode transmitter and receiver are each buffered with 16 byte FIFO's to reduce the number of interrrupts presented to the CPU.
- Adds or deletes standard asynchronous communication bits (start, stop, and parity) to or from the serial data.
- Holding and shift registers in the NS16450 Mode eliminate the need for precise synchronization between the CPU and serial data.
- Independently controlled transmit, receive, line status, and data set interrupts.
- Programmable baud generator divides any input clock by 1 to (2^{16} − 1) and generates the 16 × clock.
- Independent receiver clock input.
- MODEM control functions (CTS, RTS, DSR, DTR, RI, and DCD).
- Fully programmable serial-interface characteristics:
  - — 5-, 6-, 7-, or 8-bit characters
  - — Even, odd, or no-parity bit generation and detection
  - — 1-, 1½-, or 2-stop bit generation
  - — Baud generation (DC to 256k baud).
- False start bit detection.
- Complete status reporting capabilities.
- TRI-STATE® TTL drive for the data and control buses.
- Line break generation and detection.
- Internal diagnostic capabilities:
  - — Loopback controls for communications link fault isolation
  - — Break, parity, overrun, framing error simulation.
- Full prioritized interrupt system controls.

Basic Configuration



TL/C/8652-1

Table of Contents

1.0 Absolute Maximum Ratings

If Military/Aerospace specified devices are required, contact the National Semiconductor Sales Office/ Distributors for availability and specifications.

Temperature Under Bias	0°C to +70°C
Storage Temperature	−65°C to +150°C
All Input or Output Voltages with Respect to V_{SS}	−0.5V to +7.0V
Power Dissipation	1W

Note: *Maximum ratings indicate limits beyond which permanent damage may occur. Continuous operation at these limits is not intended and should be limited to those conditions specified under DC electrical characteristics.*

2.0 DC Electrical Characteristics

T_A = 0°C to +70°C, V_{CC} = +5V ±5%, V_{SS} = 0V, unless otherwise specified.

Symbol	Parameter	Conditions	Min	Max	Units
V_{ILX}	Clock Input Low Voltage		−0.5	0.8	V
V_{IHX}	Clock Input High Voltage		2.0	V_{CC}	V
V_{IL}	Input Low Voltage		−0.5	0.8	V
V_{IH}	Input High Voltage		2.0	V_{CC}	V
V_{OL}	Output Low Voltage	I_{OL} = 1.6 mA on all (Note 1)		0.4	V
V_{OH}	Output High Voltage	I_{OH} = −1.0 mA (Note 1)	2.4		V
I_{CC}(AV)	Avg. Power Supply Current (V_{CC})	V_{CC} = 5.25V No Loads on output SIN, DSR, DCD, CTS, RI = 2.0V All other inputs = 0.8V		160 (Note 2) 140 (Note 3)	mA mA
I_{IL}	Input Leakage	V_{CC} = 5.25V, V_{SS} = 0V All other pins floating. V_{IN} = 0V, 5.25V		±10	μA
I_{CL}	Clock Leakage			±10	μA
I_{OZ}	TRI-STATE Leakage	V_{CC} = 5.25V, V_{SS} = 0V V_{OUT} = 0V, 5.25V 1) Chip deselected 2) WRITE mode, chip selected		±20	μA
V_{ILMR}	MR Schmitt V_{IL}			0.8	V
V_{IHMR}	MR Schmitt V_{IH}		2.0		V

Note 1: Does not apply to XOUT

Note 2: T_A = 25°C

Note 3: T_A = 70°C

Capacitance T_A = 25°C, V_{CC} = V_{SS} = 0V

Symbol	Parameter	Conditions	Min	Typ	Max	Units
C_{XIN}	Clock Input Capacitance	f_c = 1 MHz Unmeasured pins returned to V_{SS}		15	20	pF
C_{XOUT}	Clock Output Capacitance			20	30	pF
C_{IN}	Input Capacitance			6	10	pF
C_{OUT}	Output Capacitance			10	20	pF

3.0 AC Electrical Characteristics $T_A = 0°C$ to $+70°C$, $V_{CC} = +5V \pm 5\%$

Symbol	Parameter	Conditions	Min	Max	Units
t_{ADS}	Address Strobe Width		60		ns
t_{AH}	Address Hold Time		0		ns
t_{AR}	$\overline{\text{RD}}$, RD Delay from Address	(Note 1)	30		ns
t_{AS}	Address Setup Time		60		ns
t_{AW}	$\overline{\text{WR}}$, WR Delay from Address	(Note 1)	30		ns
t_{CH}	Chip Select Hold Time		0		ns
t_{CS}	Chip Select Setup Time		60		ns
t_{CSR}	$\overline{\text{RD}}$, RD Delay from Chip Select	(Note 1)	30		ns
t_{CSW}	$\overline{\text{WR}}$, WR Delay from Select	(Note 1)	30		ns
t_{DH}	Data Hold Time		30		ns
t_{DS}	Data Setup Time		30		ns
t_{HZ}	$\overline{\text{RD}}$, RD to Floating Data Delay	@100 pF loading (Note 3)	0	100	ns
t_{MR}	Master Reset Pulse Width		5		μs
t_{RA}	Address Hold Time from $\overline{\text{RD}}$, RD	(Note 1)	20		ns
t_{RC}	Read Cycle Delay		125		ns
t_{RCS}	Chip Select Hold Time from $\overline{\text{RD}}$, RD	(Note 1)	20		ns
t_{RD}	$\overline{\text{RD}}$, RD Strobe Width		125		ns
t_{RDD}	$\overline{\text{RD}}$, RD to Driver Enable/Disable	@100 pF loading (Note 3)		60	ns
t_{RVD}	Delay from $\overline{\text{RD}}$, RD to Data	@100 pF loading		125	ns
t_{WA}	Address Hold Time from $\overline{\text{WR}}$, WR	(Note 1)	20		ns
t_{WC}	Write Cycle Delay		150		ns
t_{WCS}	Chip Select Hold Time from $\overline{\text{WR}}$, WR	(Note 1)	20		ns
t_{WR}	$\overline{\text{WR}}$, WR Strobe Width		100		ns
t_{XH}	Duration of Clock High Pulse	External Clock (8.0 MHz Max.)	55		ns
t_{XL}	Duration of Clock Low Pulse	External Clock (8.0 MHz Max.)	55		ns
RC	Read Cycle = $t_{AR} + t_{RD} + t_{RC}$	(Note 4)	280		ns
WC	Write Cycle = $t_{AW} + t_{WR} + t_{WC}$		280		ns
Baud Generator					
N	Baud Divisor		1	$2^{16}-1$	
t_{BHD}	Baud Output Positive Edge Delay	100 pF Load		175	ns
t_{BLD}	Baud Output Negative Edge Delay	100 pF Load		175	ns
t_{HW}	Baud Output Up Time	f_X = 8.0 MHz, ÷2, 100 pF Load	75		ns
t_{LW}	Baud Output Down Time	f_X = 8.0 MHz, ÷2, 100 pF Load	100		ns
Receiver					
t_{RINT}	Delay from $\overline{\text{RD}}$, RD (RD RBR/or RD LSR) to Reset Interrupt	100 pF Load		1	μs
t_{SCD}	Delay from RCLK to Sample Time			2	μs
t_{SINT}	Delay from Stop to Set Interrupt	(Note 2)		1	RCLK Cycles

Note 1: Applicable only when $\overline{\text{ADS}}$ is tied low.

Note 2: In the FIFO mode (FCR0 = 1) the trigger level interrupts, the receiver data available indication, the active RXRDY indication and the overrun error indication will be delayed 3 RCLKs. Status indicators (PE, FE, BI) will be delayed 3 RCLKs after the first byte has been received. For subsequently received bytes these indicators will be updated immediately after RDRBR goes inactive. Timeout interrupt is delayed 8 RCLKs.

Note 3: Charge and discharge time is determined by V_{OL}, V_{OH} and the external loading.

Note 4: In FIFO mode RC = 425 ns (minimum) between reads of the receiver FIFO and the status registers (interrupt identification register or line status register).

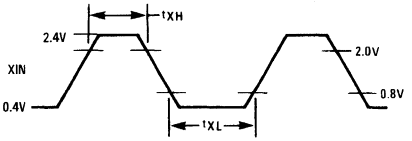
3.0 AC Electrical Characteristics (Continued)

Symbol	Parameter	Conditions	Min	Max	Units
Transmitter					
t_{HR}	Delay from $\overline{WR}$, WR (WR THR) to Reset Interrupt	100 pF Load		175	ns
t_{IR}	Delay from $\overline{RD}$, RD (RD IIR) to Reset Interrupt (THRE)	100 pF Load		250	ns
t_{IRS}	Delay from Initial INTR Reset to Transmit Start		8	24	BAUDOUT Cycles
t_{SI}	Delay from Initial Write to Interrupt	(Note 1)	16	24	BAUDOUT Cycles
t_{STI}	Delay from Stop to Interrupt (THRE)	(Note 1)	8	8	BAUDOUT Cycles
t_{SXA}	Delay from Start to TXRDY active	100 pF Load		8	BAUDOUT Cycles
t_{WXI}	Delay from Write to TXRDY inactive	100 pF Load		195	ns
Modem Control					
t_{MDO}	Delay from $\overline{WR}$, WR (WR MCR) to Output	100 pF Load		200	ns
t_{RIM}	Delay to Reset Interrupt from $\overline{RD}$, RD (RD MSR)	100 pF Load		250	ns
t_{SIM}	Delay to Set Interrupt from MODEM Input	100 pF Load		250	ns

Note 1: This delay will be lengthened by 1 character time, minus the last stop bit time if the transmitter interrupt delay circuit is active. (See FIFO Interrupt Mode Operation).

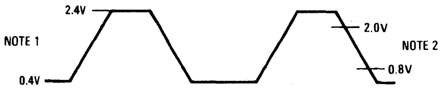
4.0 Timing Waveforms (All timings are referenced to valid 0 and valid 1)

External Clock Input (8.0 MHz Max.)



TL/C/8652-2

AC Test Points

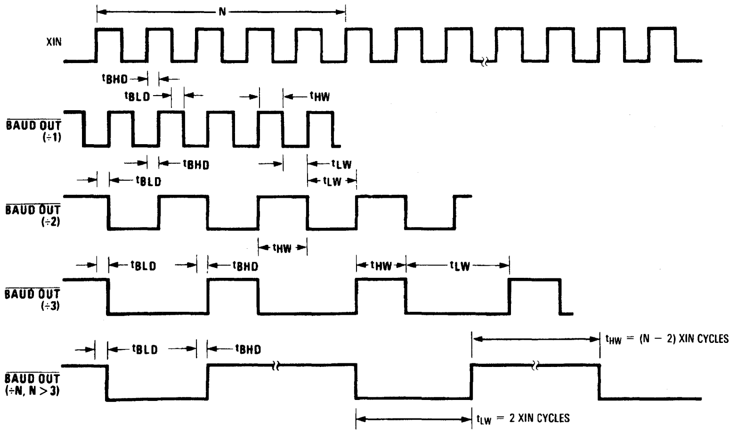


TL/C/8652-3

Note 1: The 2.4V and 0.4V levels are the voltages that the inputs are driven to during AC testing.

Note 2: The 2.0V and 0.8V levels are the voltages at which the timing tests are made.

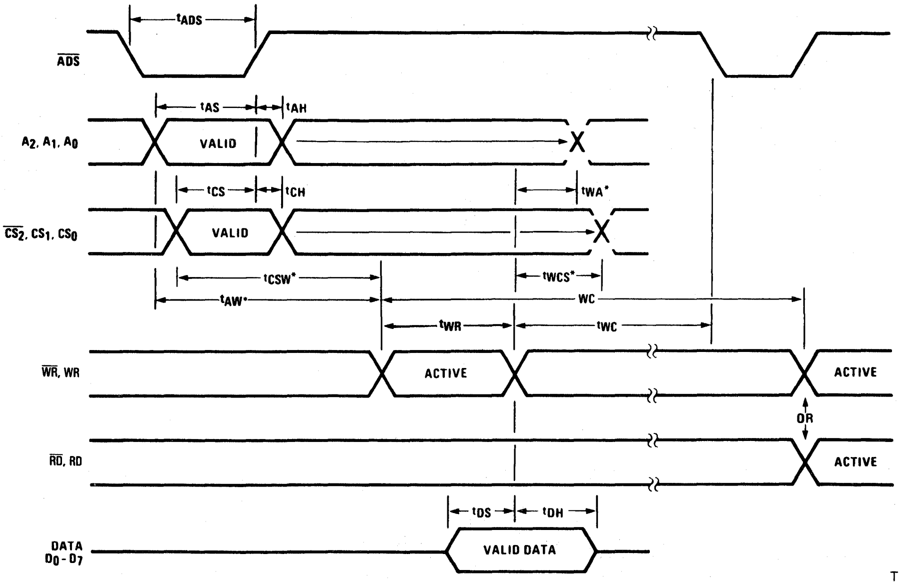
$\overline{\text{BAUDOUT}}$ Timing



TL/C/8652-4

4.0 Timing Waveforms (Continued)

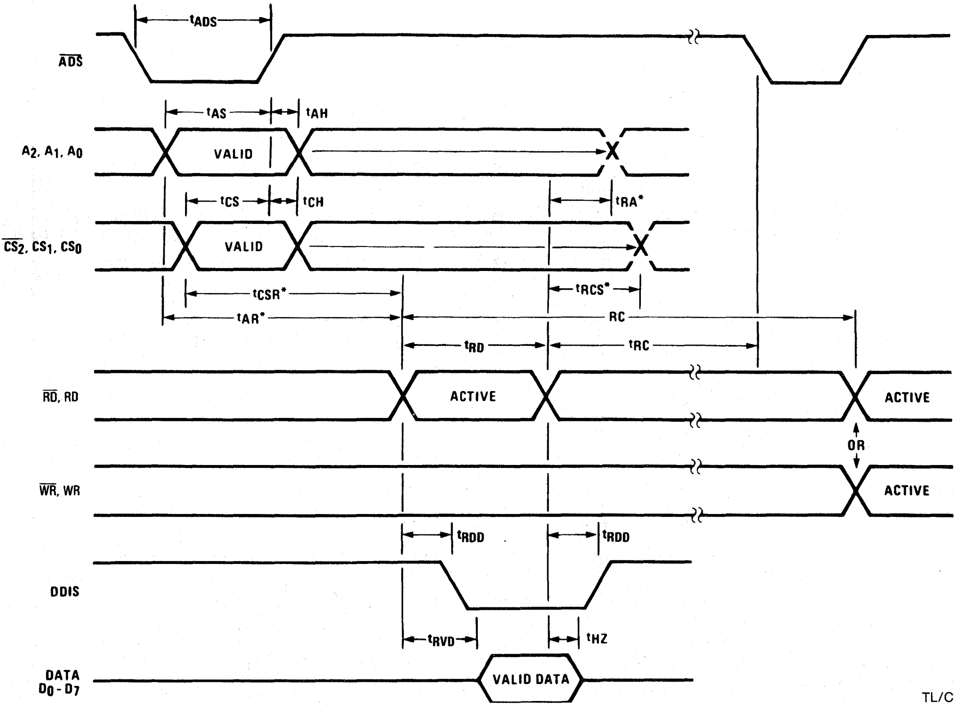
Write Cycle



TL/C/8652-5

*Applicable Only When $\overline{ADS}$ is Tied Low.

Read Cycle

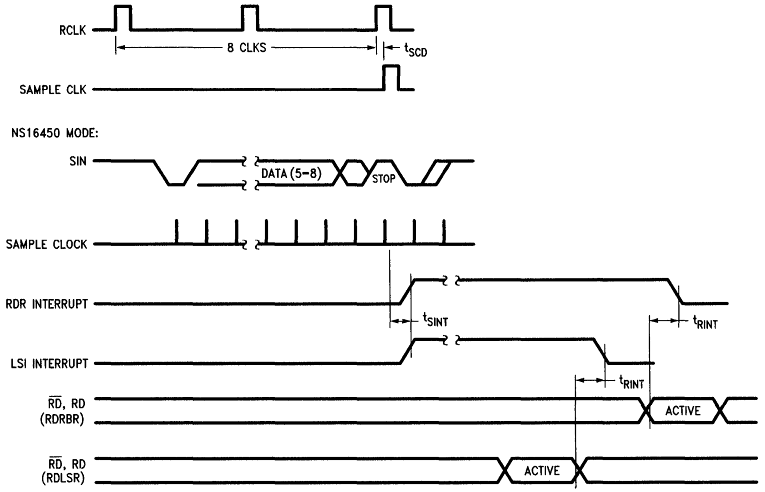


TL/C/8652-6

*Applicable Only When $\overline{ADS}$ is Tied Low.

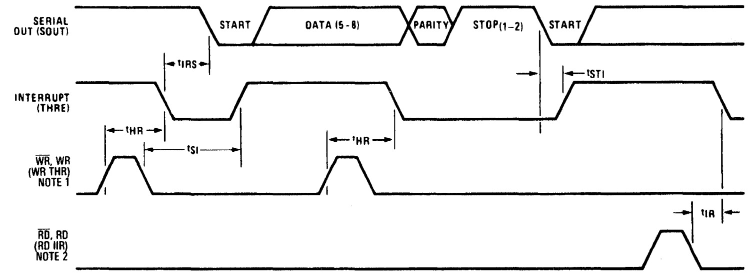
4.0 Timing Waveforms (Continued)

Receiver Timing



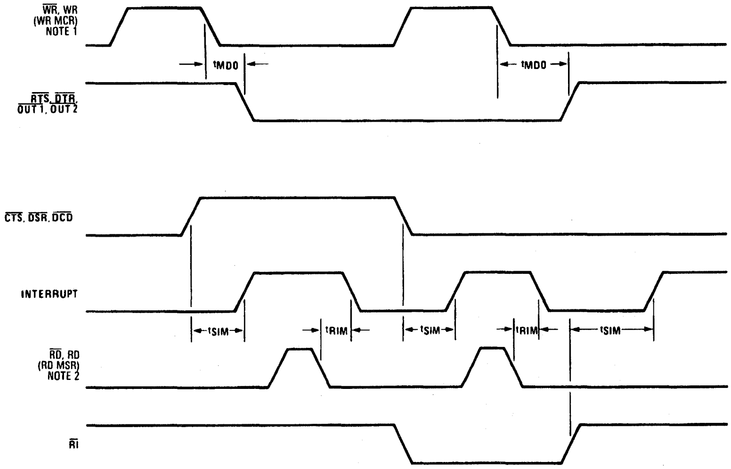
TL/C/8652-7

Transmitter Timing



TL/C/8652-8

MODEM Controls Timing


$\overline{WR}$, WR (WR MCR) NOTE 1
t_{MDO}
t_{MDO}
$\overline{RTS}$, $\overline{DTR}$, $\overline{OUT1}$, $\overline{OUT2}$
$\overline{CTS}$, $\overline{DSR}$, $\overline{DCD}$
INTERRUPT
t_{SIM}
t_{RIM}
t_{SIM}
t_{RIM}
t_{SIM}
$\overline{RD}$, RD (RD MSR) NOTE 2
$\overline{RI}$

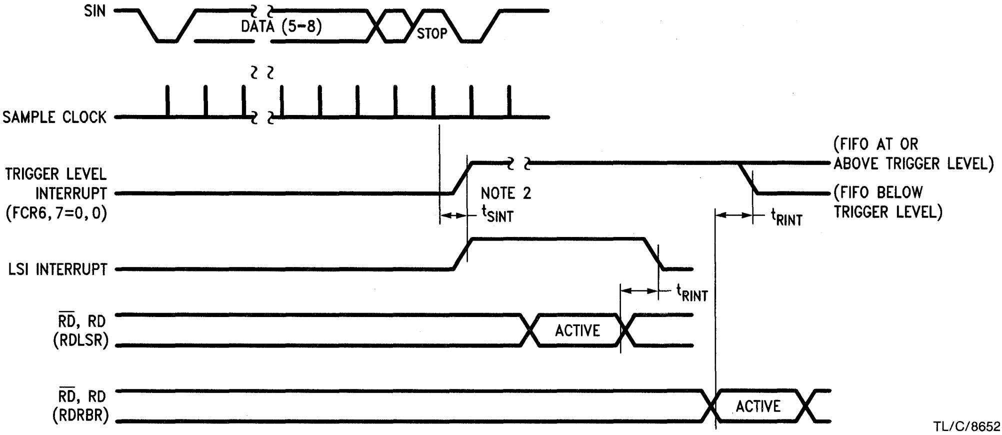
TL/C/8652-9

Note 1: See Write Cycle Timing

Note 2: See Read Cycle Timing

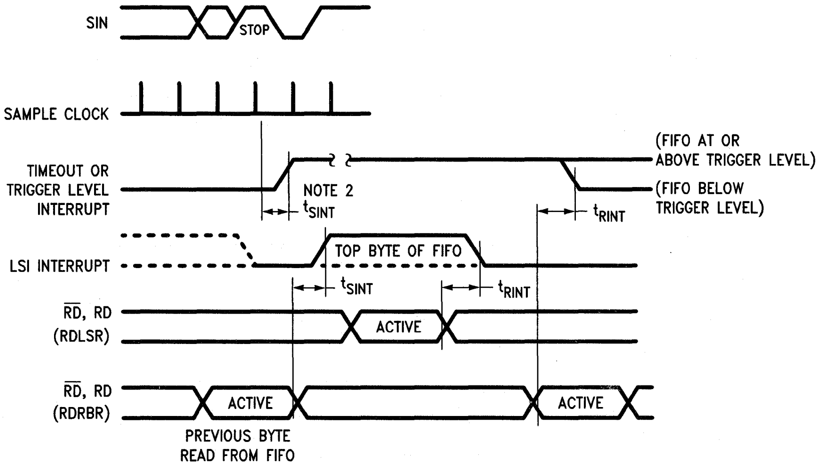
4.0 Timing Waveforms (Continued)

RCVR FIFO First Byte (This Sets RDR)



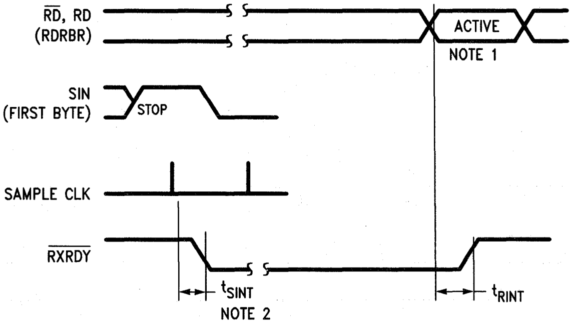
TL/C/8652-10

RCVR FIFO Bytes Other Than the First Byte (RDR Is Already Set)



TL/C/8652-11

Receiver Ready (Pin 29) FCR0 = 0 or FCR0 = 1 and FCR3 = 0 (Mode 0)



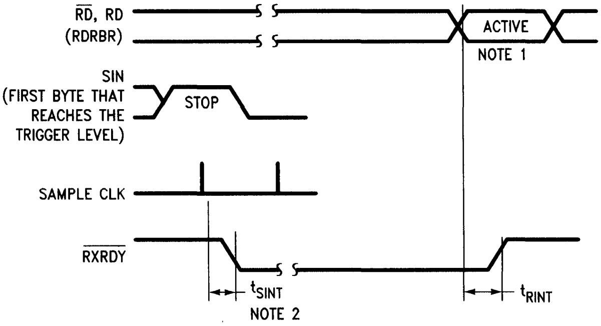
TL/C/8652-12

Note 1: This is the reading of the last byte in the FIFO.

Note 2: If FCR0 = 1, then t_{SINT} = 3 RCLKs. For a timeout interrupt, t_{SINT} = 8 RCLKs.

4.0 Timing Waveforms (Continued)

Receiver Ready (Pin 29) FCR0 = 1 and FCR3 = 1 (Mode 1)

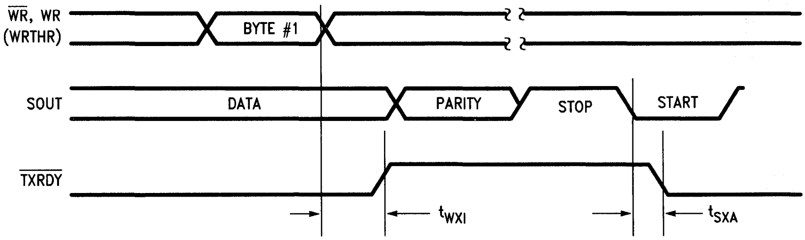


TL/C/8652-13

Note 1: This is the reading of the last byte in the FIFO.

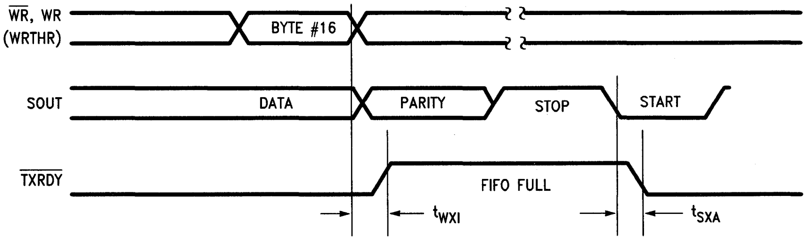
Note 2: If FCR0 = 1, t_{SINT} = 3 RCLKs.

Transmitter Ready (Pin 24) FCR0 = 0 or FCR0 = 1 and FCR3 = 0 (Mode 0)



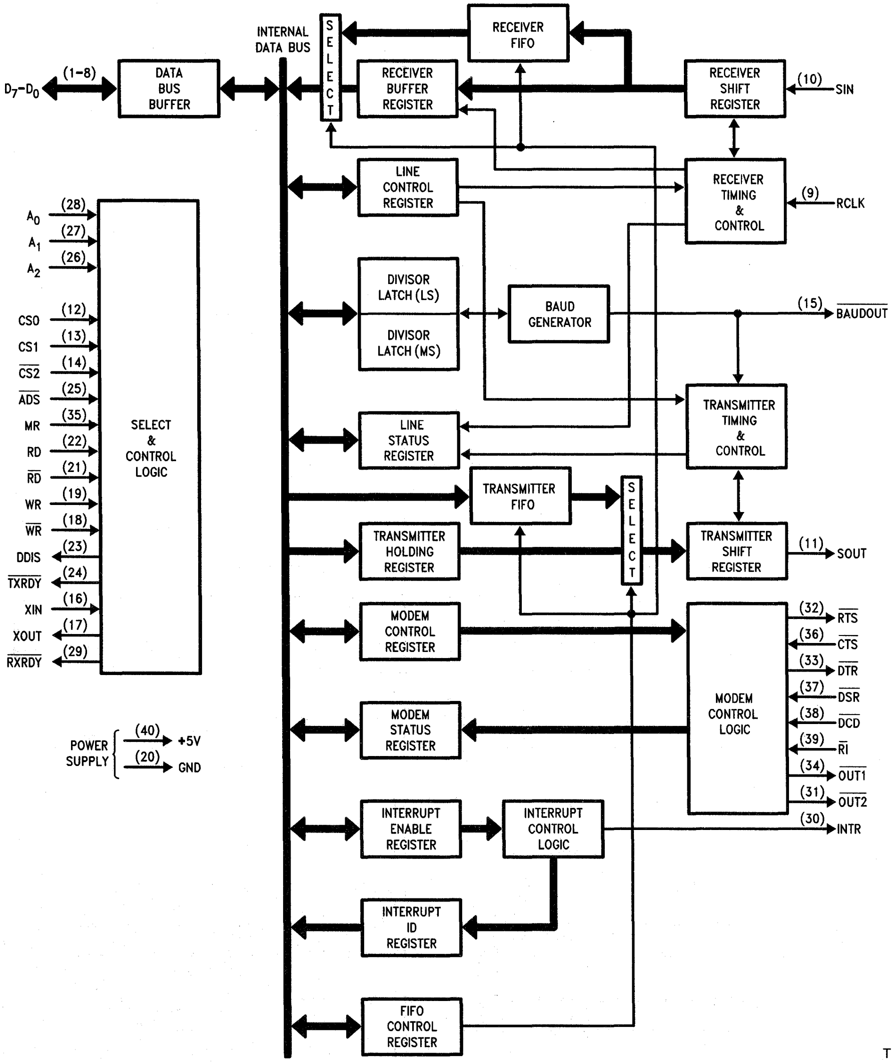
TL/C/8652-14

Transmitter Ready (Pin 24) FCR0 = 1 and FCR3 = 1 (Mode 1)



TL/C/8652-15

5.0 Block Diagram



TL/C/8652-16

Note: Applicable pinout numbers are included within parenthesis.

6.0 Pin Descriptions

The following describes the function of all UART pins. Some of these descriptions reference internal circuits.

In the following descriptions, a low represents a logic 0 (0V nominal) and a high represents a logic 1 (+2.4V nominal).

6.1 INPUT SIGNALS

Chip Select (CS0, CS1, $\overline{\text{CS2}}$), Pins 12–14: When CS0 and CS1 are high and $\overline{\text{CS2}}$ is low, the chip is selected. This enables communication between the UART and the CPU. The positive edge of an active Address Strobe signal latches the decoded chip select signals, completing chip selection. If $\overline{\text{ADS}}$ is always low, valid chip selects should stabilize according to the t_{CSW} parameter.

Read (RD, $\overline{\text{RD}}$), Pins 22 and 21: When RD is high or $\overline{\text{RD}}$ is low while the chip is selected, the CPU can read status information or data from the selected UART register.

Note: Only an active RD or $\overline{\text{RD}}$ input is required to transfer data from the UART during a read operation. Therefore, tie either the RD input permanently low or the $\overline{\text{RD}}$ input permanently high, when it is not used.

Write (WR, $\overline{\text{WR}}$), Pins 19 and 18: When WR is high or $\overline{\text{WR}}$ is low while the chip is selected, the CPU can write control words or data into the selected UART register.

Note: Only an active WR or $\overline{\text{WR}}$ input is required to transfer data to the UART during a write operation. Therefore, tie either the WR input permanently low or the $\overline{\text{WR}}$ input permanently high, when it is not used.

6.0 Pin Descriptions (Continued)

Address Strobe (ADS), Pin 25: The positive edge of an active Address Strobe (ADS) signal latches the Register Select (A0, A1, A2) and Chip Select (CS0, CS1, CS2) signals.

Note: An active ADS input is required when the Register Select (A0, A1, A2) signals are not stable for the duration of a read or write operation. If not required, tie the ADS input permanently low.

Register Select (A0, A1, A2), Pins 26–28: Address signals connected to these 3 inputs select a UART register for the CPU to read from or write to during data transfer. A table of registers and their addresses is shown below. Note that the state of the Divisor Latch Access Bit (DLAB), which is the most significant bit of the Line Control Register, affects the selection of certain UART registers. The DLAB must be set high by the system software to access the Baud Generator Divisor Latches.

Master Reset (MR), Pin 35: When this input is high, it clears all the registers (except the Receiver Buffer, Transmitter Holding, and Divisor Latches), and the control logic of the UART. The states of various output signals (SOUT, INTR, OUT 1, OUT 2, RTS, DTR) are affected by an active MR input (Refer to Table I.) This input is buffered with a TTL-compatible Schmitt Trigger with 0.5V typical hysteresis.

REGISTER ADDRESSES

DLAB	A_2	A_1	A_0	Register
0	0	0	0	Receiver Buffer (read), Transmitter Holding Register (write)
0	0	0	1	Interrupt Enable
X	0	1	0	Interrupt Identification (read)
X	0	1	0	FIFO Control (write)
X	0	1	1	Line Control
X	1	0	0	MODEM Control
X	1	0	1	Line Status
X	1	1	0	MODEM Status
X	1	1	1	Scratch
1	0	0	0	Divisor Latch (least significant byte)
1	0	0	1	Divisor Latch (most significant byte)

Receiver Clock (RCLK), Pin 9: This input is the 16 × baud rate clock for the receiver section of the chip.

Serial Input (SIN) Pin 10: Serial data input from the communications link (peripheral device, MODEM, or data set).

Clear to Send (CTS), Pin 36: When low, this indicates that the MODEM or data set is ready to exchange data. The CTS signal is a MODEM status input whose conditions can be tested by the CPU reading bit 4 (CTS) of the MODEM Status Register. Bit 4 is the complement of the CTS signal. Bit 0 (DCTS) of the MODEM Status Register indicates whether the CTS input has changed state since the previous reading of the MODEM Status Register. CTS has no effect on the Transmitter.

Note: Whenever the CTS bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.

Data Set Ready (DSR), Pin 37: When low, this indicates that the MODEM or data set is ready to establish the communications link with the UART. The DSR signal is a MODEM status input whose condition can be tested by the CPU reading bit 5 (DSR) of the MODEM Status Register. Bit 5 is the complement of the DSR signal. Bit 1 (DDSR) of the MODEM Status Register indicates whether the DSR input has changed state since the previous reading of the MODEM Status Register.

Note: Whenever the DSR bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.

Data Carrier Detect (DCD), Pin 38: When low, indicates that the data carrier has been detected by the MODEM or data set. The DCD signal is a MODEM status input whose condition can be tested by the CPU reading bit 7 (DCD) of the MODEM Status Register. Bit 7 is the complement of the DCD signal. Bit 3 (DDCD) of the MODEM Status Register indicates whether the DCD input has changed state since the previous reading of the MODEM Status Register. DCD has no effect on the receiver.

Note: Whenever the DCD bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.

Ring Indicator (RI), Pin 39: When low, this indicates that a telephone ringing signal has been received by the MODEM or data set. The RI signal is a MODEM status input whose condition can be tested by the CPU reading bit 6 (RI) of the MODEM Status Register. Bit 6 is the complement of the RI signal. Bit 2 (TERI) of the MODEM Status Register indicates whether the RI input **signal** has changed from a low to a high state since the previous reading of the MODEM Status Register.

Note: Whenever the RI **bit** of the MODEM Status Register changes from a high to a low state, an interrupt is generated if the MODEM Status Interrupt is enabled.

V_{CC}, Pin 40: +5V supply.

V_{SS}, Pin 20: Ground (0V) reference.

6.2 OUTPUT SIGNALS

Data Terminal Ready (DTR), Pin 33: When low, this informs the MODEM or data set that the UART is ready to establish a communications link. The DTR output signal can be set to an active low by programming bit 0 (DTR) of the MODEM Control Register to a high level. A Master Reset operation sets this signal to its inactive (high) state. Loop mode operation holds this signal in its inactive state.

Request to Send (RTS), Pin 32: When low, this informs the MODEM or data set that the UART is ready to exchange data. The RTS output signal can be set to an active low by programming bit 1 (RTS) of the MODEM Control Register. A Master Reset operation sets this signal to its inactive (high) state. Loop mode operation holds this signal in its inactive state.

Output 1 (OUT 1), Pin 34: This user-designated output can be set to an active low by programming bit 2 (OUT 1) of the MODEM Control Register to a high level. A Master Reset operation sets this signal to its inactive (high) state. Loop mode operation holds this signal in its inactive state. In the XMOS parts this will achieve TTL levels.

Output 2 (OUT 2), Pin 31: This user-designated output that can be set to an active low by programming bit 3 (OUT 2) of the MODEM Control Register to a high level. A Master Reset operation sets this signal to its inactive (high) state. Loop mode operation holds this signal in its inactive state. In the XMOS parts this will achieve TTL levels.

TXRDY, RXRDY, Pins 24, 29: Transmitter and Receiver DMA signalling is available through two pins (24 and 29). When operating in the FIFO mode, one of two types of DMA signalling per pin can be selected via FCR3. When operating as in the NS16450 Mode, only DMA mode 0 is allowed. Mode 0 supports single transfer DMA where a transfer is made between CPU bus cycles. Mode 1 supports multi-

6.0 Pin Descriptions (Continued)

transfer DMA where multiple transfers are made continuously until the RCVR FIFO has been emptied or the XMIT FIFO has been filled.

RXRDY Mode 0: When in the NS16450 Mode (FCR0 = 0) or in the FIFO Mode (FCR0 = 1, FCR3 = 0) and there is at least 1 character in the RCVR FIFO or RCVR holding register, the RXRDY pin (29) will be low active. Once it is activated the RXRDY pin will go inactive when there are no more characters in the FIFO or holding register.

RXRDY Mode 1: In the FIFO Mode (FCR0 = 1) when the FCR3 = 1 and the trigger level or the timeout has been reached, the RXRDY pin will go low active. Once it is activated it will go inactive when there are no more characters in the FIFO or holding register.

TXRDY Mode 0: In the NS16450 Mode (FCR0 = 0) or in the FIFO Mode (FCR0 = 1, FCR3 = 0) and there are no characters in the XMIT FIFO or XMIT holding register, the TXRDY pin (24) will be low active. Once it is activated the TXRDY pin will go inactive after the first character is loaded into the XMIT FIFO or holding register.

TXRDY Mode 1: In the FIFO Mode (FCR0 = 1) when FCR3 = 1 and there is at least one unfilled position in the XMIT FIFO, it will go low active. This pin will become inactive when the XMIT FIFO is completely full.

Driver Disable (DDIS), Pin 23: This goes low whenever the CPU is reading data from the UART. It can disable or control the direction of a data bus transceiver between the CPU and the UART.

Baud Out ($\overline{\text{BAUDOUT}}$), Pin 15: This is the 16 × clock signal from the transmitter section of the UART. The clock rate is equal to the main reference oscillator frequency divided by the specified divisor in the Baud Generator Divisor Latches. The $\overline{\text{BAUDOUT}}$ may also be used for the receiver section by tying this output to the RCLK input of the chip.

Interrupt (INTR), Pin 30: This pin goes high whenever any one of the following interrupt types has an active high condition and is enabled via the IER: Receiver Error Flag; Received Data Available: timeout (FIFO Mode only); Transmitter Holding Register Empty; and MODEM Status. The INTR signal is reset low upon the appropriate interrupt service or a Master Reset operation.

Serial Output (SOUT), Pin 11: Composite serial data output to the communications link (peripheral, MODEM or data set). The SOUT signal is set to the Marking (logic 1) state upon a Master Reset operation.

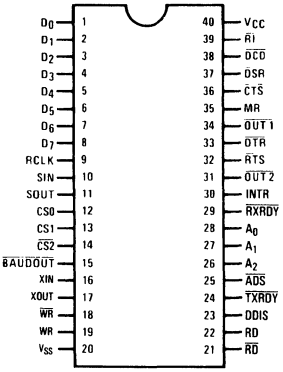
6.3 INPUT/OUTPUT SIGNALS

Data (D_7–D_0) Bus, Pins 1–8: This bus comprises eight TRI-STATE input/output lines. The bus provides bidirectional communications between the UART and the CPU. Data, control words, and status information are transferred via the D_7–D_0 Data Bus.

External Clock Input/Output (XIN, XOUT) Pins 16 and 17: These two pins connect the main timing reference (crystal or signal clock) to the UART.

7.0 Connection Diagrams

Dual-In-Line Package

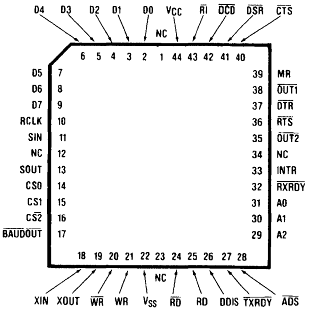


TL/C/8652–17

Top View

Order Number NS16550AN
See NS Package Number N40A

Chip Carrier Package



TL/C/8652–18

Top View

Order Number NS16550AV
See NS Package Number V44A

TABLE I. UART Reset Configuration

Register/Signal	Reset Control	Reset State
Interrupt Enable Register	Master Reset	**0000** 0000 (Note 1)
Interrupt Identification Register	Master Reset	00**00** 0001
FIFO Control	Master Reset	00**00** 0000
Line Control Register	Master Reset	0000 0000
MODEM Control Register	Master Reset	**0000** 0000
Line Status Register	Master Reset	0110 0000
MODEM Status Register	Master Reset	XXXX 0000 (Note 2)
SOUT	Master Reset	High
INTR (RCVR Errs)	Read LSR/MR	Low
INTR (RCVR Data Ready)	Read RBR/MR	Low
INTR (THRE)	Read IIR/Write THR/MR	Low
INTR (Modem Status Changes)	Read MSR/MR	Low
$\overline{\text{OUT 2}}$	Master Reset	High
$\overline{\text{RTS}}$	Master Reset	High
$\overline{\text{DTR}}$	Master Reset	High
$\overline{\text{OUT 1}}$	Master Reset	High
RCVR FIFO	MR/FCR1•FCR0/ΔFCR0	All Bits Low
XMIT FIFO	MR/FCR1•FCR0/ΔFCR0	All Bits Low

Note 1: Boldface bits are permanently low.

Note 2: Bits 7–4 are driven by the input signals.

TABLE II. Summary of Registers

Bit No.	Register Address											
	0 DLAB=0	0 DLAB=0	1 DLAB=0	2	2	3	4	5	6	7	0 DLAB=1	1 DLAB=1
	Receiver Buffer Register (Read Only)	Transmitter Holding Register (Write Only)	Interrupt Enable Register	Interrupt Ident. Register (Read Only)	FIFO Control Register (Write Only)	Line Control Register	MODEM Control Register	Line Status Register	MODEM Status Register	Scratch Reg-ister	Divisor Latch (LS)	Divisor Latch (MS)
	RBR	THR	IER	IIR	FCR	LCR	MCR	LSR	MSR	SCR	DLL	DLM
0	Data Bit 0 (Note 1)	Data Bit 0	Enable Received Data Available Interrupt (ERBFI)	"0" if Interrupt Pending	FIFO Enable	Word Length Select Bit 0 (WLS0)	Data Terminal Ready (DTR)	Data Ready (DR)	Delta Clear to Send (DCTS)	Bit 0	Bit 0	Bit 8
1	Data Bit 1	Data Bit 1	Enable Transmitter Holding Register Empty Interrupt (ETBEI)	Interrupt ID Bit (0)	RCVR FIFO Reset	Word Length Select Bit 1 (WLS1)	Request to Send (RTS)	Overrun Error (OE)	Delta Data Set Ready (DDSR)	Bit 1	Bit 1	Bit 9
2	Data Bit 2	Data Bit 2	Enable Receiver Line Status Interrupt (ELSI)	Interrupt ID Bit (1)	XMIT FIFO Reset	Number of Stop Bits (STB)	Out 1	Parity Error (PE)	Trailing Edge Ring Indicator (TERI)	Bit 2	Bit 2	Bit 10
3	Data Bit 3	Data Bit 3	Enable MODEM Status Interrupt (EDSSI)	Interrupt ID Bit (2) (Note 2)	DMA Mode Select	Parity Enable (PEN)	Out 2	Framing Error (FE)	Delta Data Carrier Detect (DDCD)	Bit 3	Bit 3	Bit 11
4	Data Bit 4	Data Bit 4	0	0	Reserved	Even Parity Select (EPS)	Loop	Break Interrupt (BI)	Clear to Send (CTS)	Bit 4	Bit 4	Bit 12
5	Data Bit 5	Data Bit 5	0	0	Reserved	Stick Parity	0	Transmitter Holding Register (THRE)	Data Set Ready (DSR)	Bit 5	Bit 5	Bit 13
6	Data Bit 6	Data Bit 6	0	FIFOs Enabled (Note 2)	RCVR Trigger (LSB)	Set Break	0	Transmitter Empty (TEMT)	Ring Indicator (RI)	Bit 6	Bit 6	Bit 14
7	Data Bit 7	Data Bit 7	0	FIFOs Enabled (Note 2)	RCVR Trigger (MSB)	Divisor Latch Access Bit (DLAB)	0	Error in RCVR FIFO (Note 2)	Data Carrier Detect (DCD)	Bit 7	Bit 7	Bit 15

Note 1: Bit 0 is the least significant bit. It is the first bit serially transmitted or received.

Note 2: These bits are always 0 in the NS16450 Mode.

8.0 Registers

The system programmer may access any of the UART registers summarized in Table II via the CPU. These registers control UART operations including transmission and reception of data. Each register bit in Table II has its name and reset state shown.

8.1 LINE CONTROL REGISTER

The system programmer specifies the format of the asynchronous data communications exchange and set the Divisor Latch Access bit via the Line Control Register (LCR). The programmer can also read the contents of the Line Control Register. The read capability simplifies system programming and eliminates the need for separate storage in system memory of the line characteristics. Table II shows the contents of the LCR. Details on each bit follow:

Bits 0 and 1: These two bits specify the number of bits in each transmitted or received serial character. The encoding of bits 0 and 1 is as follows:

Bit 1	Bit 0	Character Length
0	0	5 Bits
0	1	6 Bits
1	0	7 Bits
1	1	8 Bits

Bit 2: This bit specifies the number of Stop bits transmitted and received in each serial character. If bit 2 is a logic 0, one Stop bit is generated in the transmitted data. If bit 2 is a logic 1 when a 5-bit word length is selected via bits 0 and 1, one and a half Stop bits are generated. If bit 2 is a logic 1 when either a 6-, 7-, or 8-bit word length is selected, two Stop bits are generated. The Receiver checks the first Stop-bit only, regardless of the number of Stop bits selected.

Bit 3: This bit is the Parity Enable bit. When bit 3 is a logic 1, a Parity bit is generated (transmit data) or checked (receive data) between the last data word bit and Stop bit of the serial data. (The Parity bit is used to produce an even or odd number of 1s when the data word bits and the Parity bit are summed.)

Bit 4: This bit is the Even Parity Select bit. When bit 3 is a logic 1 and bit 4 is a logic 0, an odd number of logic 1s is transmitted or checked in the data word bits and Parity bit. When bit 3 is a logic 1 and bit 4 is a logic 1, an even number of logic 1s is transmitted or checked.

Bit 5: This bit is the Stick Parity bit. When bits 3, 4 and 5 are logic 1 the Parity bit is transmitted and checked as a logic 0. If bits 3 and 5 are 1 and bit 4 is a logic 0 then the Parity bit is transmitted and checked as a logic 1. If bit 5 is a logic 0 Stick Parity is disabled.

Bit 6: This bit is the Break Control bit. It causes a break condition to be transmitted to the receiving UART. When it is set to a logic 1, the serial output (SOUT) is forced to the Spacing (logic 0) state. The break is disabled by setting bit 6 to a logic 0. The Break Control bit acts only on SOUT and has no effect on the transmitter logic.

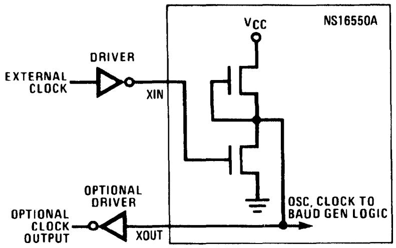
Note: This feature enables the CPU to alert a terminal in a computer communications system. If the following sequence is followed, no erroneous or extraneous characters will be transmitted because of the break.

1. Load an all 0s, pad character, in response to THRE.
2. Set break after the next THRE.
3. Wait for the transmitter to be idle, (TEMT=1), and clear break when normal transmission has to be restored.

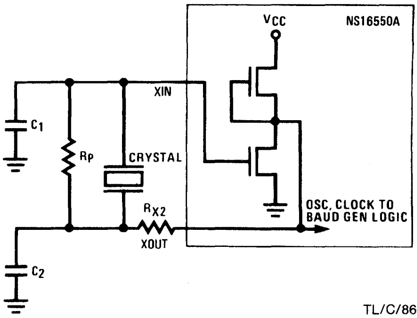
During the break, the Transmitter can be used as a character timer to accurately establish the break duration.

Bit 7: This bit is the Divisor Latch Access Bit (DLAB). It must be set high (logic 1) to access the Divisor Latches of the Baud Generator during a Read or Write operation. It must be set low (logic 0) to access the Receiver Buffer, the Transmitter Holding Register, or the Interrupt Enable Register.

8.2 TYPICAL CLOCK CIRCUITS



TL/C/8652-19



TL/C/8652-20

Typical Crystal Oscillator Network

CRYSTAL	R_P	R_{X2}	C_1	C_2
3.1 MHz	1 MΩ	1.5k	10-30 pF	40-60 pF
1.8 MHz	1 MΩ	1.5k	10-30 pF	40-60 pF

TABLE III. Baud Rates Using 1.8432 MHz Crystal

Desired Baud Rate	Decimal Divisor Used to Generate 16 x Clock	Percent Error Difference Between Desired and Actual
50	2304	—
75	1536	—
110	1047	0.026
134.5	857	0.058
150	768	—
300	384	—
600	192	—
1200	96	—
1800	64	—
2000	58	0.69
2400	48	—
3600	32	—
4800	24	—
7200	16	—
9600	12	—
19200	6	—
38400	3	—
56000	2	2.86

8.0 Registers (Continued)

8.3 PROGRAMMABLE BAUD GENERATOR

The UART contains a programmable Baud Generator that is capable of taking any clock input from DC to 8.0 MHz and dividing it by any divisor from 2 to $2^{16}-1$. 4 MHz is the highest input clock frequency recommended when the divisor = 1. The output frequency of the Baud Generator is 16 × the Baud [divisor # = (frequency input) ÷ (baud rate × 16)]. Two 8-bit latches store the divisor in a 16-bit binary format. These Divisor Latches must be loaded during initialization to ensure proper operation of the Baud Generator. Upon loading either of the Divisor Latches, a 16-bit Baud counter is immediately loaded.

Tables III, IV and V provide decimal divisors to use with crystal frequencies of 1.8432 MHz, 3.072 MHz and 8 MHz, respectively. For baud rates of 38400 and below, the error obtained is minimal. The accuracy of the desired baud rate is dependent on the crystal frequency chosen. Using a divisor of zero is **not** recommended.

8.4 LINE STATUS REGISTER

This register provides status information to the CPU concerning the data transfer. Table II shows the contents of the Line Status Register. Details on each bit follow.

Bit 0: This bit is the receiver Data Ready (DR) indicator. Bit 0 is set to a logic 1 whenever a complete incoming character has been received and transferred into the Receiver Buffer Register or the FIFO. Bit 0 is reset to a logic 0 by reading all of the data in the Receiver Buffer Register or the FIFO.

Bit 1: This bit is the Overrun Error (OE) indicator. Bit 1 indicates that data in the Receiver Buffer Register was not read by the CPU before the next character was transferred into the Receiver Buffer Register, thereby destroying the previous character. The OE indicator is set to a logic 1 upon detection of an overrun condition and reset whenever the CPU reads the contents of the Line Status Register. If the FIFO mode data continues to fill the FIFO beyond the trigger level, an overrun error will occur only after the FIFO is full and the next character has been completely received in the shift register. OE is indicated to the CPU as soon as it happens. The character in the shift register is overwritten, but it is not transferred to the FIFO.

TABLE IV. Baud Rates Using 3.072 MHz Crystal

Desired Baud Rate	Decimal Divisor Used to Generate 16 x Clock	Percent Error Difference Between Desired and Actual
50	3840	—
75	2560	—
110	1745	0.026
134.5	1428	0.034
150	1280	—
300	640	—
600	320	—
1200	160	—
1800	107	0.312
2000	96	—
2400	80	—
3600	53	0.628
4800	40	—
7200	27	1.23
9600	20	—
19200	10	—
38400	5	—

Bit 2: This bit is the Parity Error (PE) indicator. Bit 2 indicates that the received data character does not have the correct even or odd parity, as selected by the even-parity-select bit. The PE bit is set to a logic 1 upon detection of a parity error and is reset to a logic 0 whenever the CPU reads the contents of the Line Status Register. In the FIFO mode this error is associated with the particular character in the FIFO it applies to. This error is revealed to the CPU when its associated character is at the top of the FIFO.

Bit 3: This bit is the Framing Error (FE) indicator. Bit 3 indicates that the received character did not have a valid Stop bit. Bit 3 is set to a logic 1 whenever the Stop bit following the last data bit or parity bit is detected as a logic 0 bit (Spacing level). The FE indicator is reset whenever the CPU reads the contents of the Line Status Register. In the FIFO mode this error is associated with the particular character in the FIFO it applies to. This error is revealed to the CPU when its associated character is at the top of the FIFO. The UART will try to resynchronize after a framing error. To do this it assumes that the framing error was due to the next start bit, so it samples this "start" bit twice and then takes in the "data".

Bit 4: This bit is the Break Interrupt (BI) indicator. Bit 4 is set to a logic 1 whenever the received data input is held in the Spacing (logic 0) state for longer than a full word transmission time (that is, the total time of Start bit + data bits + Parity + Stop bits). The BI indicator is reset whenever the CPU reads the contents of the Line Status Register. In the FIFO mode this error is associated with the particular character in the FIFO it applies to. This error is revealed to the CPU when its associated character is at the top of the FIFO. When break occurs only one zero character is loaded into the FIFO. The next character transfer is enabled after SIN goes to the marking state and receives the next valid start bit.

Note: Bits 1 through 4 are the error conditions that produce a Receiver Line Status interrupt whenever any of the corresponding conditions are detected and the interrupt is enabled.

TABLE V. Baud Rates Using 8 MHz Crystal

Desired Baud Rate	Decimal Divisor Used to Generate 16 x Clock	Percent Error Difference Between Desired and Actual
50	10000	—
75	6667	0.005
110	4545	0.010
134.5	3717	0.013
150	3333	0.010
300	1667	0.020
600	833	0.040
1200	417	0.080
1800	277	0.080
2000	250	—
2400	208	0.160
3600	139	0.080
4800	104	0.160
7200	69	0.644
9600	52	0.160
19200	26	0.160
38400	13	0.160
56000	9	0.790
128000	4	2.344
256000	2	2.344

8.0 Registers (Continued)

TABLE VI. Interrupt Control Functions

FIFO Mode Only	Interrupt Identification Register			Interrupt Set and Reset Functions			
Bit 3	**Bit 2**	**Bit 1**	**Bit 0**	**Priority Level**	**Interrupt Type**	**Interrupt Source**	**Interrupt Reset Control**
0	0	0	1	—	None	None	—
0	1	1	0	Highest	Receiver Line Status	Overrun Error or Parity Error or Framing Error or Break Interrupt	Reading the Line Status Register
0	1	0	0	Second	Received Data Available	Receiver Data Available or Trigger Level Reached	Reading the Receiver Buffer Register or the FIFO Drops Below the Trigger Level
1	1	0	0	Second	Character Timeout Indication	No Characters Have Been Removed From or Input to the RCVR FIFO During the Last 4 Char. Times and There Is at Least 1 Char. in It During This Time	Reading the Receiver Buffer Register
0	0	1	0	Third	Transmitter Holding Register Empty	Transmitter Holding Register Empty	Reading the IIR Register (if source of interrupt) or Writing into the Transmitter Holding Register
0	0	0	0	Fourth	MODEM Status	Clear to Send or Data Set Ready or Ring Indicator or Data Carrier Detect	Reading the MODEM Status Register

Bit 5: This bit is the Transmitter Holding Register Empty (THRE) indicator. Bit 5 indicates that the UART is ready to accept a new character for transmission. In addition, this bit causes the UART to issue an interrupt to the CPU when the Transmit Holding Register Empty Interrupt enable is set high. The THRE bit is set to a logic 1 when a character is transferred from the Transmitter Holding Register into the Transmitter Shift Register. The bit is reset to logic 0 concurrently with the loading of the Transmitter Holding Register by the CPU. In the FIFO mode this bit is set when the XMIT FIFO is empty; it is cleared when at least 1 byte is written to the XMIT FIFO.

Bit 6: This bit is the Transmitter Empty (TEMT) indicator. Bit 6 is set to a logic 1 whenever the Transmitter Holding Register (THR) and the Transmitter Shift Register (TSR) are both empty. It is reset to a logic 0 whenever either the THR or TSR contains a data character. In the FIFO mode this bit is set to one whenever the transmitter FIFO and shift register are both empty.

Bit 7: In the NS16450 Mode this is a 0. In the FIFO mode LSR7 is set when there is at least one parity error, framing error or break indication in the FIFO. LSR7 is cleared when the CPU reads the LSR, if there are no subsequent errors in the FIFO.

Note: The Line Status Register is intended for read operations only. Writing to this register is not recommended as this operation is only used for factory testing.

8.5 FIFO CONTROL REGISTER

This is a write only register at the same location as the IIR (the IIR is a read only register). This register is used to enable the FIFOs, clear the FIFOs, set the RCVR FIFO trigger level, and select the type of DMA signalling.

Bit 0: Writing a 1 to FCR0 enables both the XMIT and RCVR FIFOs. Resetting FCR0 will clear all bytes in both FIFOs. When changing from FIFO Mode to NS16450 Mode and vice versa, data is automatically cleared from the FIFOs. This bit must be a 1 when other FCR bits are written to or they will not be programmed.

Bit 1: Writing a 1 to FCR1 clears all bytes in the RCVR FIFO and resets its counter logic to 0. The shift register is not cleared. The 1 that is written to this bit position is self-clearing.

Bit 2: Writing a 1 to FCR2 clears all bytes in the XMIT FIFO and resets its counter logic to 0. The shift register is not cleared. The 1 that is written to this bit position is self-clearing.

Bit 3: Setting FCR3 to a 1 will cause the RXRDY and TXRDY pins to change from mode 0 to mode 1 if FCR0 = 1 (see description of RXRDY and TXRDY pins).

Bit 4, 5: FCR4 to FCR5 are reserved for future use.

Bit 6, 7: FCR6 and FCR7 are used to set the trigger level for the RCVR FIFO interrupt.

7	6	RCVR FIFO Trigger Level (Bytes)
0	0	01
0	1	04
1	0	08
1	1	14

8.6 INTERRUPT IDENTIFICATION REGISTER

In order to provide minimum software overhead during data character transfers, the UART prioritizes interrupts into four levels and records these in the interrupt Identification Register. The four levels of interrupt conditions in order of priority are Receiver Line Status; Received Data Ready; Transmitter Holding Register Empty; and MODEM Status.

8.0 Registers (Continued)

When the CPU accesses the IIR, the UART freezes all interrupts and indicates the highest priority pending interrupt to the CPU. While this CPU access is occurring, the UART records new interrupts, but does not change its current indication until the access is complete. Table II shows the contents of the IIR. Details on each bit follow:

Bit 0: This bit can be used in a prioritized interrupt environment to indicate whether an interrupt is pending. When bit 0 is a logic 0, an interrupt is pending and the IIR contents may be used as a pointer to the appropriate interrupt service routine. When bit 0 is a logic 1, no interrupt is pending.

Bits 1 and 2: These two bits of the IIR are used to identify the highest priority interrupt pending as indicated in Table VI.

Bit 3: In the NS16450 Mode this bit is 0. In the FIFO mode this bit is set along with bit 2 when a timeout interrupt is pending.

Bits 4 and 5: These two bits of the IIR are always logic 0.

Bits 6 and 7: These two bits are set when FCR0 = 1.

8.7 INTERRUPT ENABLE REGISTER

This register enables the five types of UART interrupts. Each interrupt can individually activate the interrupt (INTR) output signal. It is possible to totally disable the interrupt system by resetting bits 0 through 3 of the Interrupt Enable Register (IER). Similarly, setting bits of the IER register to a logic 1, enables the selected interrupt(s). Disabling an interrupt prevents it from being indicated as active in the IIR and from activating the INTR output signal. All other system functions operate in their normal manner, including the setting of the Line Status and MODEM Status Registers. Table II shows the contents of the IER. Details on each bit follow.

Bit 0: This bit enables the Received Data Available Interrupt (and timeout interrupts in the FIFO mode) when set to logic 1.

Bit 1: This bit enables the Transmitter Holding Register Empty Interrupt when set to logic 1.

Bit 2: This bit enables the Receiver Line Status Interrupt when set to logic 1.

Bit 3: This bit enables the MODEM Status Interrupt when set to logic 1.

Bits 4 through 7: These four bits are always logic 0.

8.8 MODEM CONTROL REGISTER

This register controls the interface with the MODEM or data set (or a peripheral device emulating a MODEM). The contents of the MODEM Control Register are indicated in Table II and are described below.

Bit 0: This bit controls the Data Terminal Ready ($\overline{\text{DTR}}$) output. When bit 0 is set to a logic 1, the $\overline{\text{DTR}}$ output is forced to a logic 0. When bit 0 is reset to a logic 0, the $\overline{\text{DTR}}$ output is forced to a logic 1.

Note: The $\overline{\text{DTR}}$ output of the UART may be applied to an EIA inverting line driver (such as the DS1488) to obtain the proper polarity input at the succeeding MODEM or data set.

Bit 1: This bit controls the Request to Send ($\overline{\text{RTS}}$) output. Bit 1 affects the $\overline{\text{RTS}}$ output in a manner identical to that described above for bit 0.

Bit 2: This bit controls the Output 1 ($\overline{\text{OUT 1}}$) signal, which is an auxiliary user-designated output. Bit 2 affects the $\overline{\text{OUT 1}}$ output in a manner identical to that described above for bit 0.

Bit 3: This bit controls the Output 2 ($\overline{\text{OUT 2}}$) signal, which is an auxiliary user-designated output. Bit 3 affects the $\overline{\text{OUT 2}}$ output in a manner identical to that described above for bit 0.

Bit 4: This bit provides a local loopback feature for diagnostic testing of the UART. When bit 4 is set to logic 1, the following occur: the transmitter Serial Output (SOUT) is set to the Marking (logic 1) state; the receiver Serial Input (SIN) is disconnected; the output of the Transmitter Shift Register is "looped back" into the Receiver Shift Register input; the four MODEM Control inputs ($\overline{\text{CTS}}$, $\overline{\text{DSR}}$, $\overline{\text{RI}}$, and $\overline{\text{DCD}}$) are disconnected; and the four MODEM Control outputs ($\overline{\text{DTR}}$, $\overline{\text{RTS}}$, $\overline{\text{OUT 1}}$, and $\overline{\text{OUT 2}}$) are internally connected to the four MODEM Control inputs, and the MODEM Control output pins are forced to their inactive state (high). In the diagnostic mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit-and received-data paths of the UART.

In the diagnostic mode, the receiver and transmitter interrupts are fully operational. Their sources are external to the part. The MODEM Control Interrupts are also operational, but the interrupts' sources are now the lower four bits of the MODEM Control Register instead of the four MODEM Control inputs. The interrupts are still controlled by the Interrupt Enable Register.

Bits 5 through 7: These bits are permanently set to logic 0.

8.9 MODEM STATUS REGISTER

This register provides the current state of the control lines from the MODEM (or peripheral device) to the CPU. In addition to this current-state information, four bits of the MODEM Status Register provide change information. These bits are set to a logic 1 whenever a control input from the MODEM changes state. They are reset to logic 0 whenever the CPU reads the MODEM Status Register.

The contents of the MODEM Status Register are indicated in Table II and described below.

Bit 0: This bit is the Delta Clear to Send (DCTS) indicator. Bit 0 indicates that the $\overline{\text{CTS}}$ input to the chip has changed state since the last time it was read by the CPU.

Bit 1: This bit is the Delta Data Set Ready (DDSR) indicator. Bit 1 indicates that the $\overline{\text{DSR}}$ input to the chip has changed state since the last time it was read by the CPU.

Bit 2: This bit is the Trailing Edge of Ring Indicator (TERI) detector. Bit 2 indicates that the $\overline{\text{RI}}$ input to the chip has changed from a low to a high state.

Bit 3: This bit is the Delta Data Carrier Detect (DDCD) indicator. Bit 3 indicates that the $\overline{\text{DCD}}$ input to the chip has changed state.

Note: Whenever bit 0, 1, 2, or 3 is set to logic 1, a MODEM Status Interrupt is generated.

Bit 4: This bit is the complement of the Clear to Send ($\overline{\text{CTS}}$) input. If bit 4 (loop) of the MCR is set to a 1, this bit is equivalent to RTS in the MCR.

Bit 5: This bit is the complement of the Data Set Ready ($\overline{\text{DSR}}$) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to DTR in the MCR.

8.0 Registers (Continued)

Bit 6: This bit is the complement of the Ring Indicator ($\overline{RI}$) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to OUT 1 in the MCR.

Bit 7: This bit is the complement of the Data Carrier Detect ($\overline{DCD}$) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to OUT 2 in the MCR.

8.10 SCRATCHPAD REGISTER

This 8-bit Read/Write Register does not control the UART in anyway. It is intended as a scratchpad register to be used by the programmer to hold data temporarily.

8.11 FIFO INTERRUPT MODE OPERATION

When the RCVR FIFO and receiver interrupts are enabled (FCR0 = 1, IER0 = 1) RCVR interrupts will occur as follows:

A. The receive data available interrupt will be issued to the CPU when the FIFO has reached its programmed trigger level; it will be cleared as soon as the FIFO drops below its programmed trigger level.

B. The IIR receive data available indication also occurs when the FIFO trigger level is reached, and like the interrupt it is cleared when the FIFO drops below the trigger level.

C. The receiver line status interrupt (IIR = 06), as before, has higher priority than the received data available (IIR = 04) interrupt.

D. The data ready bit (LSR0) is set as soon as a character is transferred from the shift register to the RCVR FIFO. It is reset when the FIFO is empty.

When RCVR FIFO and receiver interrupts are enabled, RCVR FIFO timeout interrupts will occur as follows:

A. A FIFO timeout interrupt will occur, if the following conditions exist:

- — at least one character is in the FIFO
- — the most recent serial character received was longer than 4 continuous character times ago (if 2 stop bits are programmed the second one is included in this time delay).
- — the most recent CPU read of the FIFO was longer than 4 continuous character times ago.

This will cause a maximum character received to interrupt issued delay of 160 ms at 300 BAUD with a 12 bit character.

B. Character times are calculated by using the RCLK input for a clock signal (this makes the delay proportional to the baudrate).

C. When a timeout interrupt has occurred it is cleared and the timer reset when the CPU reads one character from the RCVR FIFO.

D. When a timeout interrupt has not occurred the timeout timer is reset after a new character is received or after the CPU reads the RCVR FIFO.

When the XMIT FIFO and transmitter interrupts are enabled (FCR0 = 1, IER1 = 1), XMIT interrupts will occur as follows:

A. The transmitter holding register interrupt (02) occurs when the XMIT FIFO is empty; it is cleared as soon as the transmitter holding register is written to (1 to 16 characters may be written to the XMIT FIFO while servicing this interrupt) or the IIR is read.

B. The transmitter FIFO empty indications will be delayed 1 character time minus the last stop bit time whenever the following occurs: THRE = 1 and there have not been at least two bytes at the same time in the transmit FIFO, since the last THRE = 1. The first transmitter interrupt after changing FCR0 will be immediate, if it is enabled.

Character timeout and RCVR FIFO trigger level interrupts have the same priority as the current received data available interrupt; XMIT FIFO empty has the same priority as the current transmitter holding register empty interrupt.

8.12 FIFO POLLED MODE OPERATION

With FCR0 = 1 resetting IER0, IER1, IER2, IER3 or all to zero puts the UART in the FIFO Polled Mode of operation. Since the RCVR and XMITTER are controlled separately either one or both can be in the polled mode of operation.

In this mode the user's program will check RCVR and XMITTER status via the LSR. As stated previously:

LSR0 will be set as long as there is one byte in the RCVR FIFO.

LSR1 to LSR4 will specify which error(s) has occurred. Character error status is handled the same way as when in the interrupt mode, the IIR is not affected since IER2 = 0.

LSR5 will indicate when the XMIT FIFO is empty.

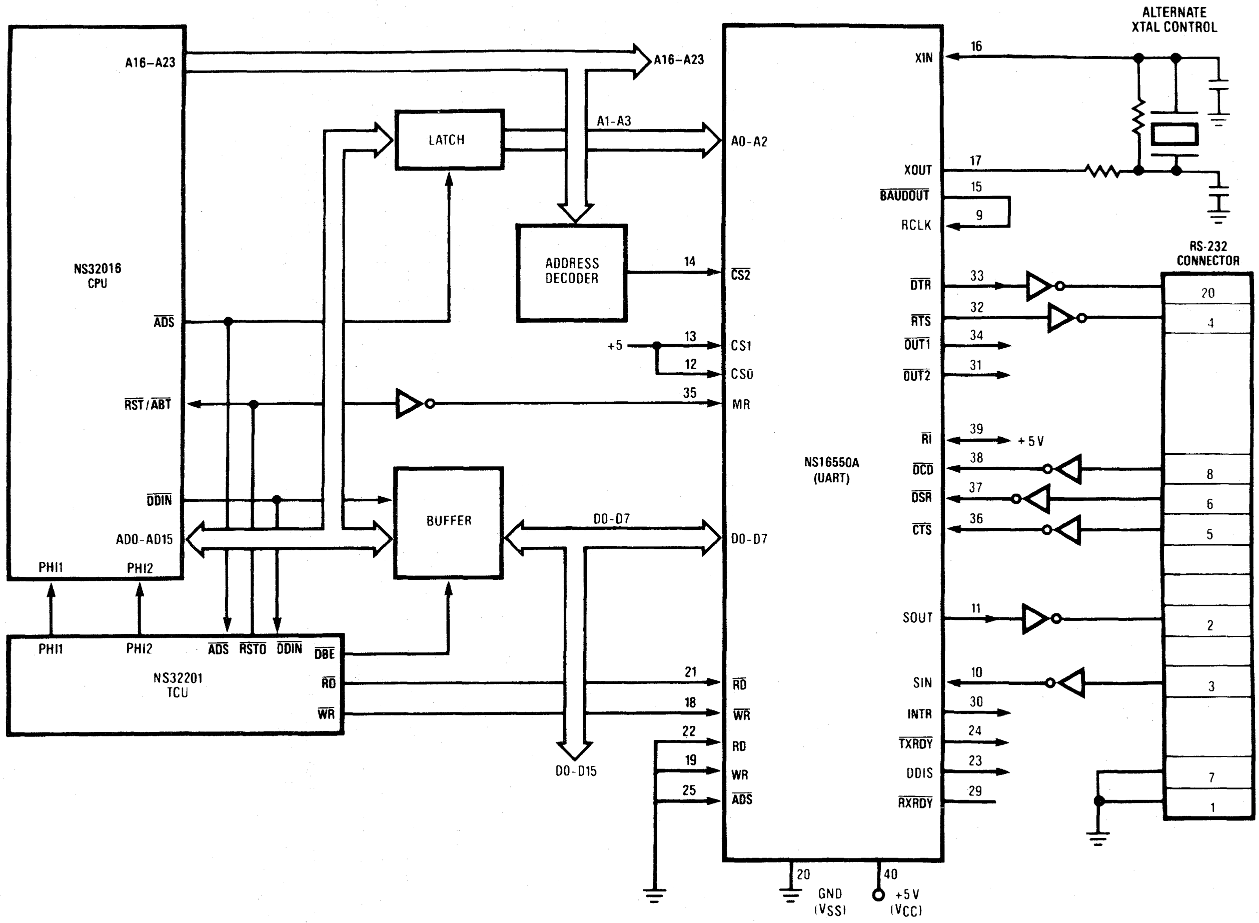
LSR6 will indicate that both the XMIT FIFO and shift register are empty.

LSR7 will indicate whether there are any errors in the RCVR FIFO.

There is no trigger level reached or timeout condition indicated in the FIFO Polled Mode, however, the RCVR and XMIT FIFOs are still fully capable of holding characters.

9.0 Typical Applications

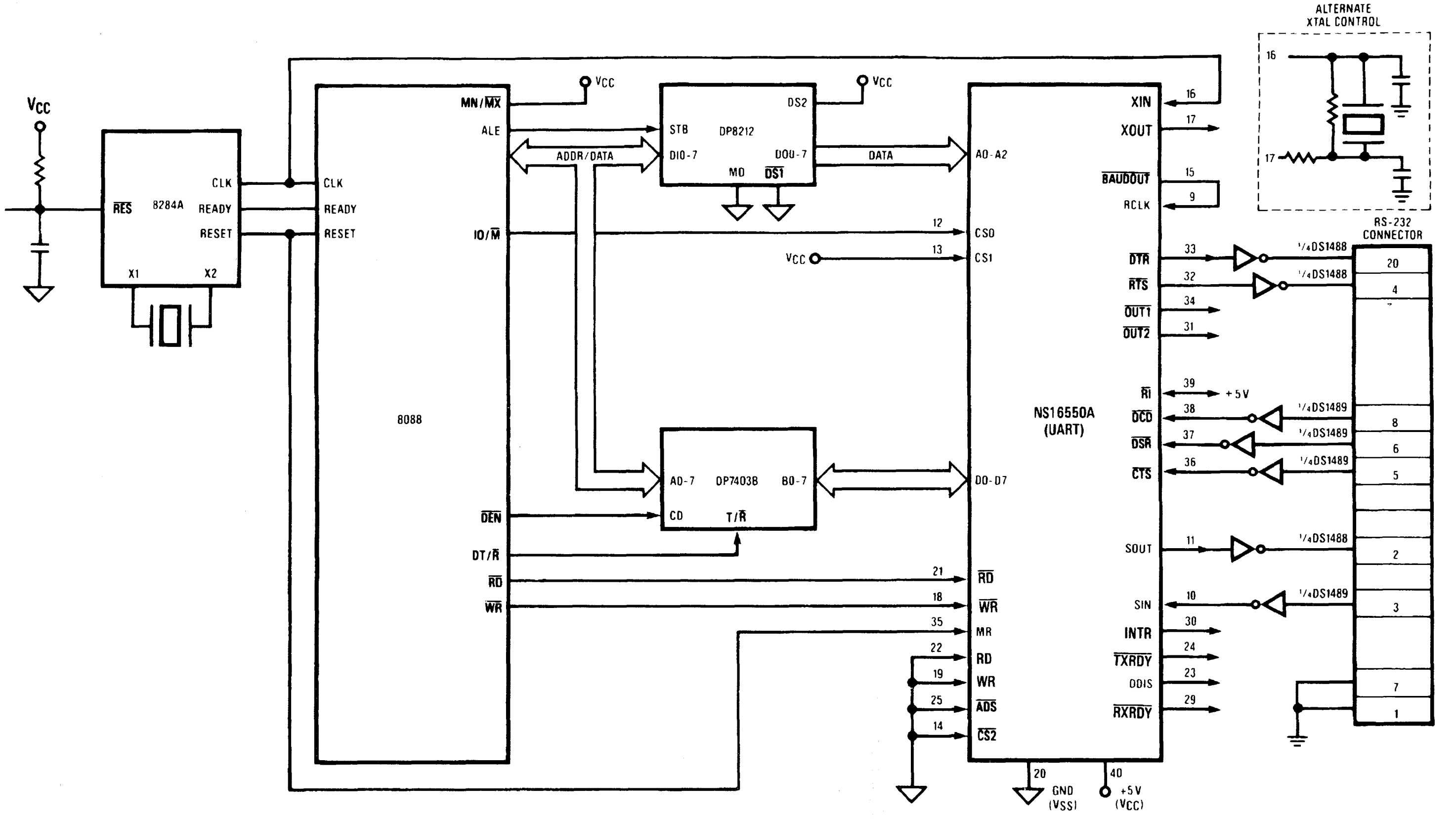
This shows the basic connections of an NS16550A to an NS32016 CPU



TL/C/8652-21

9.0 Typical Applications (Continued)

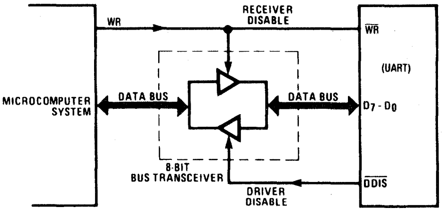
This shows the basic connections of an NS16550A to an 8088 CPU



TL/C/8652-22

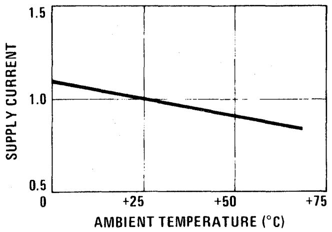
9.0 Typical Applications (Continued)

Typical Interface for a High-Capacity Data Bus



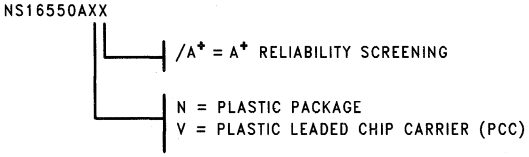
TL/C/8652–23

Typical Supply Current vs. Temperature, Normalized



TL/C/8652–24

10.0 Ordering Information


NS16550AXX
/A+ = A+ RELIABILITY SCREENING
N = PLASTIC PACKAGE
V = PLASTIC LEADED CHIP CARRIER (PCC)

TL/C/8652–25

11.0 Reliability Information

Gate Count	3,400
Transistor Count	10,300

National Semiconductor Corporation

NS16450, INS8250A, NS16C450, INS82C50A Universal Asynchronous Receiver/Transmitter

General Description

Each of these parts function as a serial data input/output interface in a microcomputer system. The system software determines the functional configuration of the UART via a TRI-STATE® 8-bit bidirectional data bus.

The UART performs serial-to-parallel conversion on data characters received from a peripheral device or a MODEM, and parallel-to-serial conversion on data characters received from the CPU. The CPU can read the complete status of the UART at any time during the functional operation. Status information reported includes the type and condition of the transfer operations being performed by the UART, as well as any error conditions (parity, overrun, framing, or break interrupt).

The UART includes a programmable baud rate generator that is capable of dividing the timing reference clock input by divisors of 1 to ($2^{16}-1$), and producing a 16 × clock for driving the internal transmitter logic. Provisions are also included to use this 16 × clock to drive the receiver logic. The UART includes a complete MODEM-control capability and a processor-interrupt system. Interrupts can be programmed to the user's requirements, minimizing the computing required to handle the communications link.

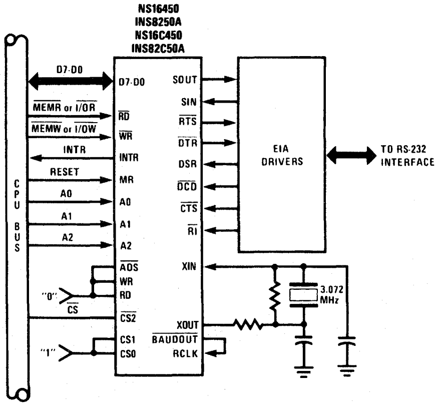
The NS16450 is an improved specification version of the INS8250A Universal Asynchronous Receiver/Transmitter (UART). The improved specifications ensure compatibility with the NS32032 and other state-of-the-art CPUs. Functionally, the NS16450 is equivalent to the INS8250A. The UART is fabricated using National Semiconductor's advanced scaled N-channel silicon-gate MOS process, XMOS.

The NS16C450 and INS82C50A are functionally equivalent to their XMOS counterparts, except that they are CMOS parts.

Features

- Easily interfaces to most popular microprocessors.
- Adds or deletes standard asynchronous communication bits (start, stop, and parity) to or from serial data stream.
- Holding and shift registers eliminate the need for precise synchronization between the CPU and the serial data.
- Independently controlled transmit, receive, line status, and data set interrupts.
- Programmable baud generator allows division of any input clock by 1 to ($2^{16}-1$) and generates the internal 16 × clock.
- Independent receiver clock input.
- MODEM control functions (CTS, RTS, DSR, DTR, RI, and DCD).
- Fully programmable serial-interface characteristics:
  - — 5-, 6-, 7-, or 8-bit characters
  - — Even, odd, or no-parity bit generation and detection
  - — 1-, 1½-, or 2-stop bit generation
  - — Baud generation (DC to 56k baud).
- False start bit detection.
- Complete status reporting capabilities.
- TRI-STATE TTL drive capabilities for bidirectional data bus and control bus.
- Line break generation and detection.
- Internal diagnostic capabilities:
  - — Loopback controls for communications link fault isolation
  - — Break, parity, overrun, framing error simulation.
- Fully prioritized interrupt system controls.

Connection Diagram



TL/C/8401-1

Table of Contents

1.0 Absolute Maximum Ratings

If Military/Aerospace specified devices are required, contact the National Semiconductor Sales Office/ Distributors for availability and specifications.

Temperature Under Bias	0°C to +70°C
Storage Temperature	−65°C to +150°C
All Input or Output Voltages with Respect to V_{SS}	−0.5V to +7.0V
Power Dissipation	700 mW

Note: *Maximum ratings indicate limits beyond which permanent damage may occur. Continuous operation at these limits is not intended and should be limited to those conditions specified under DC electrical characteristics.*

2.0 DC Electrical Characteristics

T_A = 0°C to +70°C, V_{CC} = +5V ±5%, V_{SS} = 0V, unless otherwise specified.

Symbol	Parameter	Conditions	NS16450 NS16C450 (Note 1)		INS8250A INS82C50A (Note 1)		Units
			Min	Max	Min	Max	
V_{ILX}	Clock Input Low Voltage		−0.5	0.8	−0.5	0.8	V
V_{IHX}	Clock Input High Voltage		2.0	V_{CC}	2.0	V_{CC}	V
V_{IL}	Input Low Voltage		−0.5	0.8	−0.5	0.8	V
V_{IH}	Input High Voltage		2.0	V_{CC}	2.0	V_{CC}	V
V_{OL}	Output Low Voltage	I_{OL} = 1.6 mA on all (Note 2)		0.4		0.4	V
V_{OH}	Output High Voltage	I_{OH} = −1.0 mA (Note 2)	2.4		2.4		V
I_{CC}(AV)	Avg. Power Supply Current (V_{CC}) **XMOS Parts Only**	V_{CC} = 5.25V, T_A = 25°C No Loads on output SIN, DSR, DCD, CTS, RI = 2.4V All other inputs = 0.4V		120		95	mA
I_{CC}(AV)	Avg. Power Supply Current (V_{CC}) **CMOS Parts Only**	V_{CC} = 5.25V, T_A = 25°C No Loads on output SIN, DSR, DCD, CTS, RI = 2.4V All other inputs = 0.4V Baud Rate Generator is 4 MHz Baud Rate is 50k		10		10	mA
I_{IL}	Input Leakage	V_{CC} = 5.25V, V_{SS} = 0V All other pins floating. V_{IN} = 0V, 5.25V		±10		±10	μA
I_{CL}	Clock Leakage			±10		±10	μA
I_{OZ}	TRI-STATE Leakage	V_{CC} = 5.25V, V_{SS} = 0V V_{OUT} = 0V, 5.25V 1) Chip deselected 2) WRITE mode, chip selected		±20		±20	μA
V_{ILMR}	MR Schmitt V_{IL}			0.8		0.8	V
V_{IHMR}	MR Schmitt V_{IH}		2.0		2.0		V

Capacitance T_A = 25°C, V_{CC} = V_{SS} = 0V

Symbol	Parameter	Conditions	Min	Typ	Max	Units
C_{XIN}	Clock Input Capacitance	f_c = 1 MHz Unmeasured pins returned to V_{SS}		15	20	pF
C_{XOUT}	Clock Output Capacitance			20	30	pF
C_{IN}	Input Capacitance			6	10	pF
C_{OUT}	Output Capacitance			10	20	pF

Note 1: Inputs on the CMOS parts are TTL compatible; outputs on the CMOS parts drive to GND and V_{CC}.

Note 2: Does not apply to XOUT.

3.0 AC Electrical Characteristics T_A = 0°C to +70°C, V_{CC} = +5V ±5%

Symbol	Parameter	Conditions	NS16450 NS16C450		INS8250A INS82C50A		Units
			Min	Max	Min	Max	
t_{ADS}	Address Strobe Width		60		90		ns
t_{AH}	Address Hold Time		0		0		ns
t_{AR}	RD, $\overline{\text{RD}}$ Delay from Address	(Note 1)	60		80		ns
t_{AS}	Address Setup Time		60		90		ns
t_{AW}	WR, $\overline{\text{WR}}$ Delay from Address	(Note 1)	60		80		ns
t_{CH}	Chip Select Hold Time		0		0		ns
t_{CS}	Chip Select Setup Time		60		90		ns
t_{CSC}	Chip Select Output Delay from Select	@100 pF loading (Note 1)		100		125	ns
t_{CSR}	RD, $\overline{\text{RD}}$ Delay from Chip Select	(Note 1)	50		80		ns
t_{CSW}	WR, $\overline{\text{WR}}$ Delay from Select	(Note 1)	50		80		ns
t_{DH}	Data Hold Time		40		60		ns
t_{DS}	Data Setup Time		40		90		ns
t_{HZ}	RD, $\overline{\text{RD}}$ to Floating Data Delay	@100 pF loading (Note 3)	0	100	0	100	ns
t_{MR}	Master Reset Pulse Width		5		10		µs
t_{RA}	Address Hold Time from RD, $\overline{\text{RD}}$	(Note 1)	20		20		ns
t_{RC}	Read Cycle Delay		175		500		ns
t_{RCS}	Chip Select Hold Time from RD, $\overline{\text{RD}}$	(Note 1)	20		20		ns
t_{RD}	RD, $\overline{\text{RD}}$ Strobe Width		125		175		ns
t_{RDD}	RD, $\overline{\text{RD}}$ to Driver Disable Delay	@100 pF loading (Note 3)		60		75	ns
t_{RVD}	Delay from RD, $\overline{\text{RD}}$ to Data	@100 pF loading		125		175	ns
t_{WA}	Address Hold Time from WR, $\overline{\text{WR}}$	(Note 1)	20		20		ns
t_{WC}	Write Cycle Delay		200		500		ns
t_{WCS}	Chip Select Hold Time from WR, $\overline{\text{WR}}$	(Note 1)	20		20		ns
t_{WR}	WR, $\overline{\text{WR}}$ Strobe Width		100		175		ns
t_{XH}	Duration of Clock High Pulse	External Clock (3.1 MHz Max.)	140		140		ns
t_{XL}	Duration of Clock Low Pulse	External Clock (3.1 MHz Max.)	140		140		ns
RC	Read Cycle = $t_{AR} + t_{RD} + t_{RC}$		360		755		ns
WC	Write Cycle = $t_{AW} + t_{WR} + t_{WC}$		360		755		ns
Baud Generator							
N	Baud Divisor		1	$2^{16}-1$	1	$2^{16}-1$	
t_{BHD}	Baud Output Positive Edge Delay	100 pF Load		175		250	ns
t_{BLD}	Baud Output Negative Edge Delay	100 pF Load		175		250	ns
t_{HW}	Baud Output Up Time	f_X = 3 MHz, ÷3, 100 pF Load	250		250		ns
t_{LW}	Baud Output Down Time	f_X = 2 MHz, ÷2, 100 pF Load	425		425		ns
Receiver							
t_{RINT}	Delay from RD, $\overline{\text{RD}}$ (RD RBR or RD LSR) to Reset Interrupt	100 pF Load		1		1	µs
t_{SCD}	Delay from RCLK to Sample Time			2		2	µs
t_{SINT}	Delay from Stop to Set Interrupt			1		1	RCLK Cycles (Note 2)

Note 1: Applicable only when $\overline{\text{ADS}}$ is tied low.

Note 2: RCLK is equal to t_{XH} and t_{XL}.

Note 3: Charge and discharge time is determined by V_{OL}, V_{OH} and the external loading.

3.0 AC Electrical Characteristics $T_A = 0°C$ to $+70°C$, $V_{CC} = +5V \pm 5\%$ (Continued)

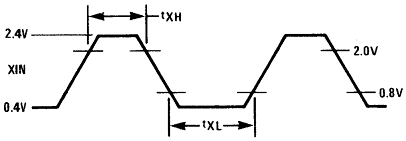
Symbol	Parameter	Conditions	NS16450 NS16C450		INS8250A INS82C50A		Units
			Min	Max	Min	Max	
Transmitter							
t_{HR}	Delay from WR, $\overline{WR}$ (WR THR) to Reset Interrupt	100 pF Load		175		1000	ns
t_{IR}	Delay from RD, $\overline{RD}$ (RD IIR) to Reset Interrupt (THRE)	100 pF Load		250		1000	ns
t_{IRS}	Delay from Initial INTR Reset to Transmit Start		24	40	24	40	BAUDOUT Cycles
t_{SI}	Delay from Initial Write to Interrupt	(Note 1)	16	24	16	24	BAUDOUT Cycles
t_{STI}	Delay from Stop to Interrupt (THRE)		8	8	8	8	BAUDOUT Cycles
Modem Control							
t_{MDO}	Delay from WR, $\overline{WR}$ (WR MCR) to Output	100 pF Load		200		1000	ns
t_{RIM}	Delay to Reset Interrupt from RD, $\overline{RD}$ (RD MSR)	100 pF Load		250		1000	ns
t_{SIM}	Delay to Set Interrupt from MODEM Input	100 pF Load		250		1000	ns

Note 1: For both the NS16C450 and INS82C50A, t_{SI} is a minimum of 16 and a maximum of 48 BAUDOUT cycles.

Note 2: For both the NS16C450 and INS82C50A, t_{IRS} is a minimum of 24 and a maximum of 40 BAUDOUT cycles.

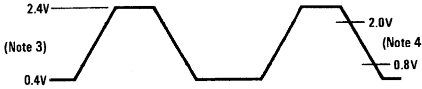
4.0 Timing Waveforms (All timings are referenced to valid 0 and valid 1)

External Clock Input (3.1 MHz Max.)



TL/C/8401-2

AC Test Points

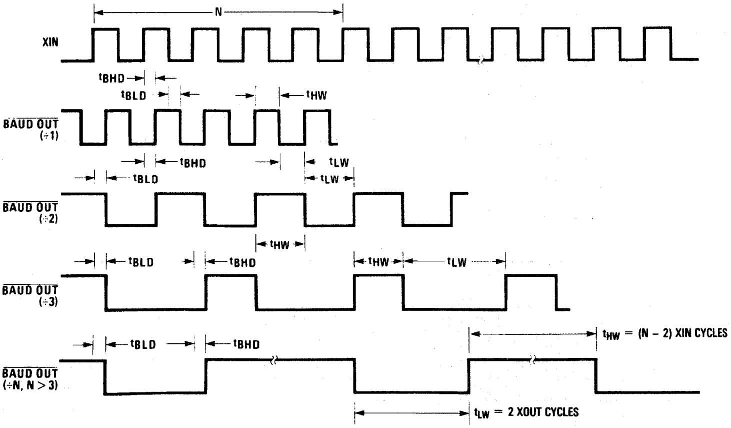


TL/C/8401-3

Note 3: The 2.4V and 0.4V levels are the voltages that the inputs are driven to during AC testing.

Note 4: The 2.0V and 0.8V levels are the voltages at which the timing tests are made.

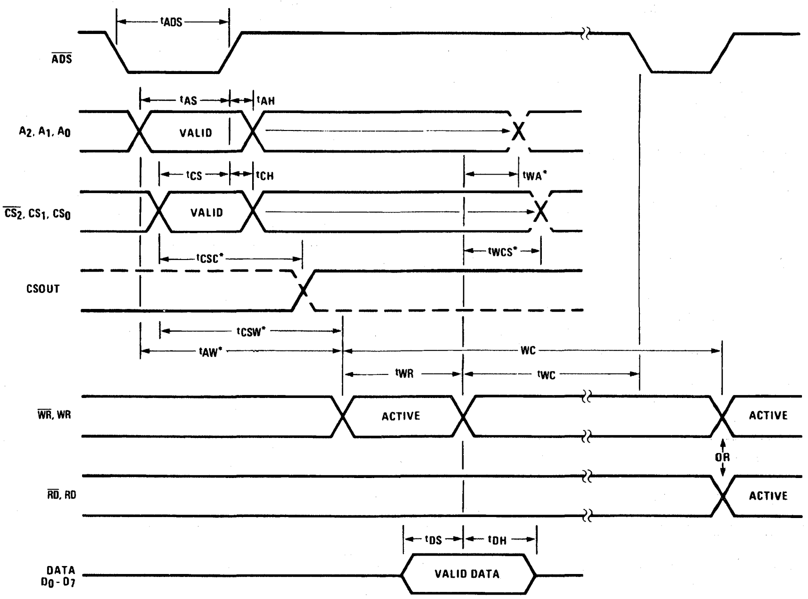
$\overline{\text{BAUDOUT}}$ Timing



TL/C/8401-4

4.0 Timing Waveforms (Continued)

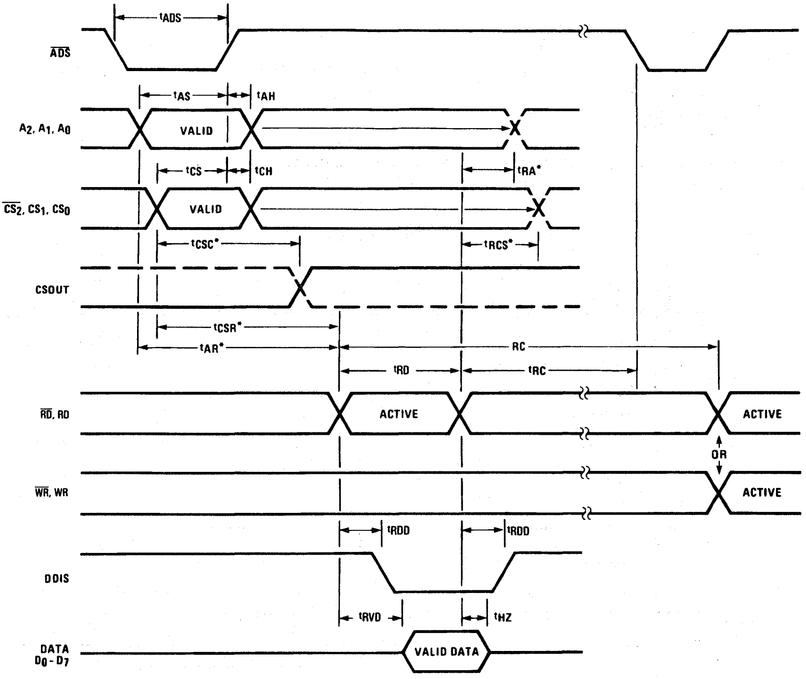
Write Cycle



*Applicable Only When $\overline{\text{ADS}}$ is Tied Low.

TL/C/8401-5

Read Cycle

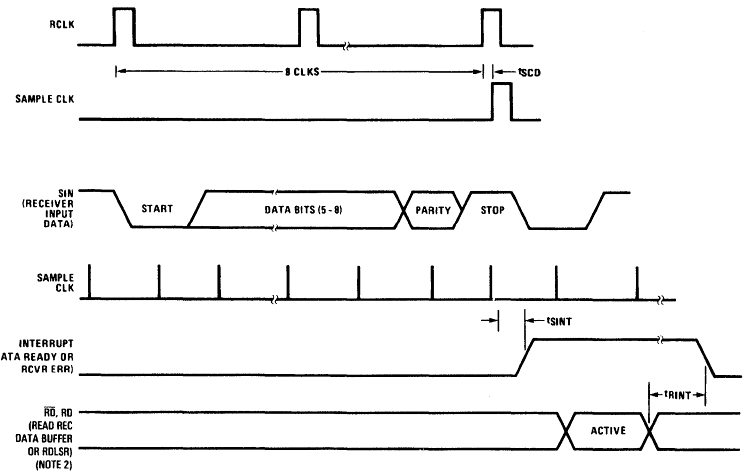


*Applicable Only When $\overline{\text{ADS}}$ is Tied Low.

TL/C/8401-6

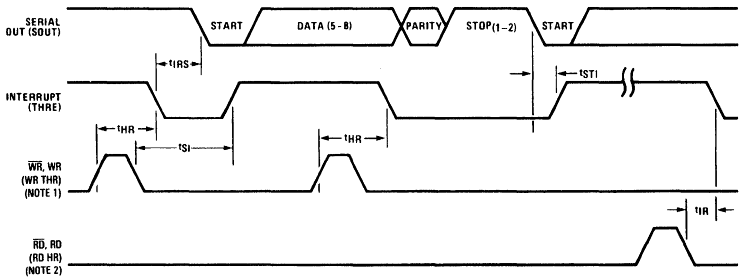
4.0 Timing Waveforms (Continued)

Receiver Timing


RCLK
8 CLKS
tSCD
SAMPLE CLK
SIN (RECEIVER INPUT DATA)
START
DATA BITS (5-8)
PARITY
STOP
SAMPLE CLK
tSINT
INTERRUPT (DATA READY OR RCVR ERR)
tRINT
RD, RD (READ REC DATA BUFFER OR RDLSR) (NOTE 2)
ACTIVE

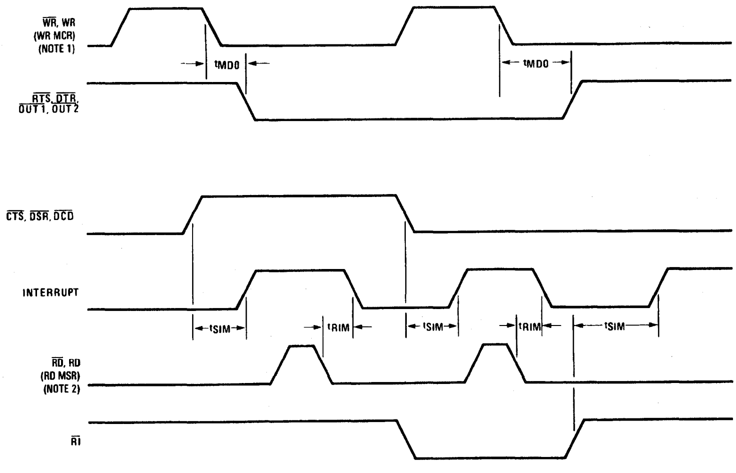
TL/C/8401-7

Transmitter Timing



TL/C/8401-8

MODEM Controls Timing

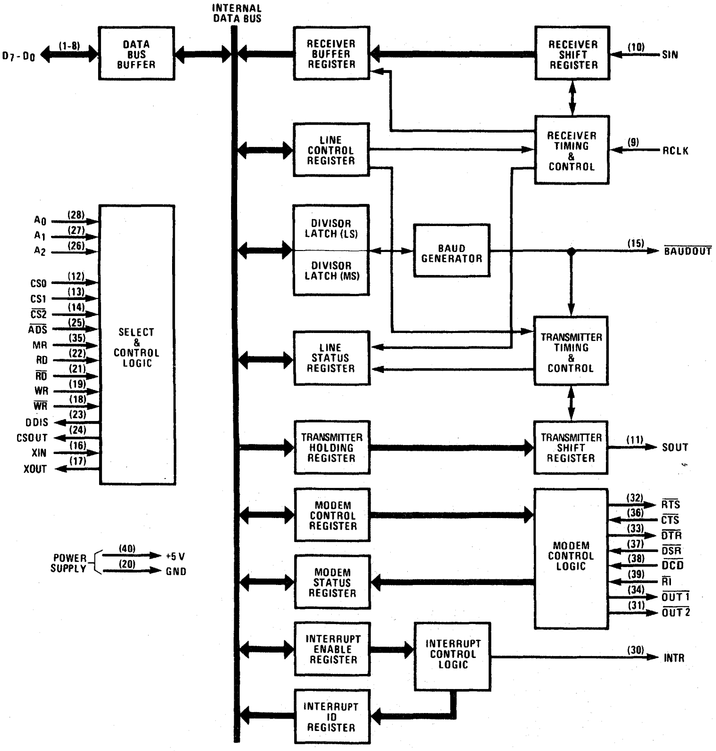


TL/C/8401-9

Note 1: See Write Cycle Timing

Note 2: See Read Cycle Timing

5.0 Block Diagram



TL/C/8401-10

Note: Applicable pinout numbers are included within parenthesis.

6.0 Pin Descriptions

The following describes the function of all UART pins. Some of these descriptions reference internal circuits.

In the following descriptions, a low represents a logic 0 (0V nominal) and a high represents a logic 1 (+2.4V nominal).

6.1 INPUT SIGNALS

Chip Select (CS0, CS1, $\overline{\text{CS2}}$), Pins 12–14: When CS0 and CS1 are high and $\overline{\text{CS2}}$ is low, the chip is selected. This enables communication between the UART and the CPU. The positive edge of an active Address Strobe signal latches the decoded chip select signals, completing chip selection. If $\overline{\text{ADS}}$ is always low, valid chip selects should stabilize according to the t_{CSW} parameter.

Read (RD, $\overline{\text{RD}}$), Pins 22 and 21: When RD is high or $\overline{\text{RD}}$ is low while the chip is selected, the CPU can read status information or data from the selected UART register.

Note: Only an active RD or $\overline{\text{RD}}$ input is required to transfer data from the UART during a read operation. Therefore, tie either the RD input permanently low or the $\overline{\text{RD}}$ input permanently high, when it is not used.

Write (WR, $\overline{\text{WR}}$), Pins 19 and 18: When WR is high or $\overline{\text{WR}}$ is low while the chip is selected, the CPU can write control words or data into the selected UART register.

Note: Only an active WR or $\overline{\text{WR}}$ input is required to transfer data to the UART during a write operation. Therefore, tie either the WR input permanently low or the $\overline{\text{WR}}$ input permanently high, when it is not used.

Address Strobe ($\overline{\text{ADS}}$), Pin 25: The positive edge of an active Address Strobe ($\overline{\text{ADS}}$) signal latches the Register Select (A0, A1, A2) and Chip Select (CS0, CS1, CS2) signals.

Note: An active $\overline{\text{ADS}}$ input is required when the Register Select (A0, A1, A2) signals are not stable for the duration of a read or write operation. If not required, tie the $\overline{\text{ADS}}$ input permanently low.

Register Select (A0, A1, A2), Pins 26–28: Address signals connected to these 3 inputs select a UART register for the CPU to read from or write to during data transfer. A table of registers and their addresses is shown below. Note that the state of the Divisor Latch Access Bit (DLAB), which is the most significant bit of the Line Control Register, affects the selection of certain UART registers. The DLAB must be set high by the system software to access the Baud Generator Divisor Latches.

6.0 Pin Descriptions (Continued)

Register Addresses

DLAB	A_2	A_1	A_0	Register
0	0	0	0	Receiver Buffer (read), Transmitter Holding Register (write)
0	0	0	1	Interrupt Enable
X	0	1	0	Interrupt Identification (read only)
X	0	1	1	Line Control
X	1	0	0	MODEM Control
X	1	0	1	Line Status
X	1	1	0	MODEM Status
X	1	1	1	Scratch
1	0	0	0	Divisor Latch (least significant byte)
1	0	0	1	Divisor Latch (most significant byte)

Master Reset (MR), Pin 35: When this input is high, it clears all the registers (except the Receiver Buffer, Transmitter Holding, and Divisor Latches), and the control logic of the UART. The states of various output signals (SOUT, INTR, $\overline{\text{OUT 1}}$, $\overline{\text{OUT 2}}$, $\overline{\text{RTS}}$, $\overline{\text{DTR}}$) are affected by an active MR input. (Refer to Table I.) This input is buffered with a TTL-compatible Schmitt Trigger with 0.5V typical hysteresis.

Receiver Clock (RCLK), Pin 9: This input is the 16 × baud rate clock for the receiver section of the chip.

Serial Input (SIN), Pin 10: Serial data input from the communications link (peripheral device, MODEM, or data set).

Clear to Send ($\overline{\text{CTS}}$), Pin 36: When low, this indicates that the MODEM or data set is ready to exchange data. The $\overline{\text{CTS}}$ signal is a MODEM status input whose conditions can be tested by the CPU reading bit 4 (CTS) of the MODEM Status Register. Bit 4 is the complement of the $\overline{\text{CTS}}$ signal. Bit 0 (DCTS) of the MODEM Status Register indicates whether the $\overline{\text{CTS}}$ input has changed state since the previous reading of the MODEM Status Register. $\overline{\text{CTS}}$ has no effect on the Transmitter.

Note: Whenever the CTS bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.

Data Set Ready ($\overline{\text{DSR}}$), Pin 37: When low, this indicates that the MODEM or data set is ready to establish the communications link with the UART. The $\overline{\text{DSR}}$ signal is a MODEM status input whose condition can be tested by the CPU reading bit 5 (DSR) of the MODEM Status Register. Bit 5 is the complement of the $\overline{\text{DSR}}$ signal. Bit 1 (DDSR) of the MODEM Status Register indicates whether the $\overline{\text{DSR}}$ input has changed state since the previous reading of the MODEM Status Register.

Note: Whenever the DSR bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.

Data Carrier Detect ($\overline{\text{DCD}}$), Pin 38: When low, indicates that the data carrier has been detected by the MODEM or data set. The $\overline{\text{DCD}}$ signal is a MODEM status input whose condition can be tested by the CPU reading bit 7 (DCD) of the MODEM Status Register. Bit 7 is the complement of the $\overline{\text{DCD}}$ signal. Bit 3 (DDCD) of the MODEM Status Register indicates whether the $\overline{\text{DCD}}$ input has changed state since the previous reading of the MODEM Status Register. $\overline{\text{DCD}}$ has no effect on the receiver.

Note: Whenever the DCD bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.

Ring Indicator ($\overline{\text{RI}}$), Pin 39: When low, this indicates that a telephone ringing signal has been received by the MODEM or data set. The $\overline{\text{RI}}$ signal is a MODEM status input whose condition can be tested by the CPU reading bit 6 (RI) of the MODEM Status Register. Bit 6 is the complement of the $\overline{\text{RI}}$ signal. Bit 2 (TERI) of the MODEM Status Register indicates whether the $\overline{\text{RI}}$ input **signal** has changed from a low to a high state since the previous reading of the MODEM Status Register.

Note: Whenever the RI **bit** of the MODEM Status Register changes from a high to a low state, an interrupt is generated if the MODEM Status interrupt is enabled.

V_{CC}, Pin 40: +5V supply.

V_{SS}, Pin 20: Ground (0V) reference.

6.2 OUTPUT SIGNALS

Data Terminal Ready ($\overline{\text{DTR}}$), Pin 33: When low, this informs the MODEM or data set that the UART is ready to establish a communications link. The $\overline{\text{DTR}}$ output signal can be set to an active low by programming bit 0 (DTR) of the MODEM Control Register to a high level. A Master Reset operation sets this signal to its inactive (high) state. Loop mode operation holds this signal in its inactive state.

Request to Send ($\overline{\text{RTS}}$), Pin 32: When low, this informs the MODEM or data set that the UART is ready to exchange data. The $\overline{\text{RTS}}$ output signal can be set to an active low by programming bit 1 (RTS) of the MODEM Control Register. A Master Reset operation sets this signal to its inactive (high) state. Loop mode operation holds this signal in its inactive state.

Output 1 ($\overline{\text{OUT 1}}$), Pin 34: This user-designated output can be set to an active low by programming bit 2 (OUT 1) of the MODEM Control Register to a high level. A Master Reset operation sets this signal to its inactive (high) state. Loop mode operation holds this signal in its inactive state. In the XMOS parts this will achieve TTL levels.

Output 2 ($\overline{\text{OUT 2}}$), Pin 31: This user-designated output can be set to an active low, by programming bit 3 (OUT 2) of the MODEM Control Register to a high level. A Master Reset operation sets this signal to its inactive (high) state. Loop mode operation holds this signal in its inactive state. In the XMOS parts this will achieve TTL levels.

Chip Select Out (CSOUT), Pin 24: When high, it indicates that the chip has been selected by active, CS0, CS1, and $\overline{\text{CS2}}$ inputs. No data transfer can be initiated until the CSOUT signal is a logic 1. CSOUT goes low when the UART is deselected.

Driver Disable (DDIS), Pin 23: This goes low whenever the CPU is reading data from the UART. It can disable or control the direction of a data bus transceiver between the CPU and the UART (see Typical Interface for a High Capacity Data Bus).

Baud Out ($\overline{\text{BAUDOUT}}$), Pin 15: This is the 16 × clock signal from the transmitter section of the UART. The clock rate is equal to the main reference oscillator frequency divided by the specified divisor in the Baud Generator Divisor Latches. The $\overline{\text{BAUDOUT}}$ may also be used for the receiver section by tying this output to the RCLK input of the chip.

6.0 Pin Descriptions (Continued)

Interrupt (INTR), Pin 30: This goes high whenever any one of the following interrupt types has an active high condition and is enabled via the IER: Receiver Line Status; Received Data Available; Transmitter Holding Register Empty; and MODEM Status. The INTR signal is reset low upon the appropriate interrupt service or a Master Reset operation.

Serial Output (SOUT), Pin 11: This is the composite serial data output to the communications link (peripheral, MODEM or data set). The SOUT signal is set to the Marking (logic 1) state upon a Master Reset operation or when the transmitter is idle.

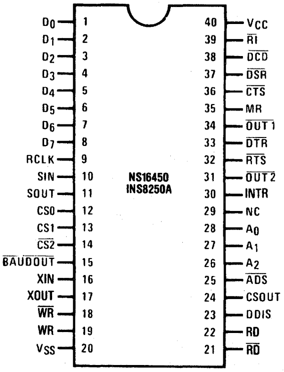
6.3 INPUT/OUTPUT SIGNALS

Data (D_7–D_0) Bus, Pins 1–8: This bus is comprised of eight TRI-STATE input/output lines. The bus provides bidirectional communications between the UART and the CPU. Data, control words, and status information are transferred via the D_7–D_0 Data Bus.

External Clock Input/Output (XIN, XOUT) Pins 16 and 17: These two pins connect the main timing reference (crystal or signal clock) to the UART. When a crystal oscillator or a clock signal is provided, it drives the UART via XIN (see typical oscillator network illustration).

7.0 Connection Diagrams

Dual-In-Line Package

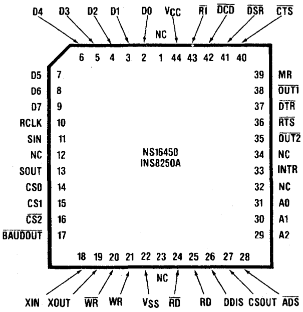

D_0 1
D_1 2
D_2 3
D_3 4
D_4 5
D_5 6
D_6 7
D_7 8
RCLK 9
SIN 10
SOUT 11
CS0 12
CS1 13
$\overline{\text{CS2}}$ 14
$\overline{\text{BAUDOUT}}$ 15
XIN 16
XOUT 17
$\overline{\text{WR}}$ 18
WR 19
V_{SS} 20
40 V_{CC}
39 $\overline{\text{RI}}$
38 $\overline{\text{DCD}}$
37 $\overline{\text{DSR}}$
36 $\overline{\text{CTS}}$
35 MR
34 $\overline{\text{OUT 1}}$
33 $\overline{\text{DTR}}$
32 $\overline{\text{RTS}}$
31 $\overline{\text{OUT 2}}$
30 INTR
29 NC
28 A_0
27 A_1
26 A_2
25 $\overline{\text{ADS}}$
24 CSOUT
23 DDIS
22 RD
21 $\overline{\text{RD}}$
NS16450
INS8250A

TL/C/8401–11

Top View

Order Number NS16450N, NS-16450N, INS8250AN, NS16C450N or INS82C50AN
See NS Package Number N40A

PCC Package



TL/C/8401–18

Top View

Order Number NS16450V, NS-16450V, INS8250AV, NS16C450V or INS82C50AV
See NS Package Number V44A

TABLE I. UART Reset Functions

Register/Signal	Reset Control	Reset State
Interrupt Enable Register	Master Reset	**0000** 0000 (Note 1)
Interrupt Identification Register	Master Reset	**0000 0**001
Line Control Register	Master Reset	0000 0000
MODEM Control Register	Master Reset	**000**0 0000
Line Status Register	Master Reset	**0**110 0000
MODEM Status Register	Master Reset	XXXX 0000 (Note 2)
SOUT	Master Reset	High
INTR (RCVR Errs)	Read LSR/MR	Low
INTR (RCVR Data Ready)	Read RBR/MR	Low
INTR (THRE)	Read IIR/Write THR/MR	Low
INTR (Modem Status Changes)	Read MSR/MR	Low
$\overline{\text{OUT 2}}$	Master Reset	High
$\overline{\text{RTS}}$	Master Reset	High
$\overline{\text{DTR}}$	Master Reset	High
$\overline{\text{OUT 1}}$	Master Reset	High

Note 1: Boldface bits are permanently low.

Note 2: Bits 7–4 are driven by the input signals.

8.0 Registers

The system programmer may access any of the UART registers summarized in Table II via the CPU. These registers control UART operations including transmission and reception of data. Each register bit in Table II has its name and reset state shown.

8.1 LINE CONTROL REGISTER

The system programmer specifies the format of the asynchronous data communications exchange and sets the Divisor Latch Access bit via the Line Control Register (LCR). The programmer can also read the contents of the Line Control Register. The read capability simplifies system programming and eliminates the need for separate storage in system memory of the line characteristics. Table II shows the contents of the LCR. Details on each bit follow:

Bits 0 and 1: These two bits specify the number of bits in each transmitted or received serial character. The encoding of bits 0 and 1 is as follows:

Bit 1	Bit 0	Character Length
0	0	5 Bits
0	1	6 Bits
1	0	7 Bits
1	1	8 Bits

Bit 2: This bit specifies the number of Stop bits transmitted and received in each serial character. If bit 2 is a logic 0, one Stop bit is generated or checked in the transmitted data. If bit 2 is a logic 1 when a 5-bit word length is selected via bits 0 and 1, one and a half Stop bits are generated. If

TABLE II. Summary of Registers

	Register Address										
	0 DLAB=0	0 DLAB=0	1 DLAB=0	2	3	4	5	6	7	0 DLAB=1	1 DLAB=1
Bit No.	**Receiver Buffer Register (Read Only)**	**Transmitter Holding Register (Write Only)**	**Interrupt Enable Register**	**Interrupt Ident. Register (Read Only)**	**Line Control Register**	**MODEM Control Register**	**Line Status Register**	**MODEM Status Register**	**Scratch Register**	**Divisor Latch (LS)**	**Divisor Latch (MS)**
	RBR	**THR**	**IER**	**IIR**	**LCR**	**MCR**	**LSR**	**MSR**	**SCR**	**DLL**	**DLM**
0	Data Bit 0 (Note 1)	Data Bit 0	Received Data Available	"0" if Interrupt Pending	Word Length Select Bit 0 (WLS0)	Data Terminal Ready (DTR)	Data Ready (DR)	Delta Clear to Send (DCTS)	Bit 0	Bit 0	Bit 8
1	Data Bit 1	Data Bit 1	Transmitter Holding Register Empty	Interrupt ID Bit (0)	Word Length Select Bit 1 (WLS1)	Request to Send (RTS)	Overrun Error (OE)	Delta Data Set Ready (DDSR)	Bit 1	Bit 1	Bit 9
2	Data Bit 2	Data Bit 2	Receiver Line Status	Interrupt ID Bit (1)	Number of Stop Bits (STB)	Out 1	Parity Error (PE)	Trailing Edge Ring Indicator (TERI)	Bit 2	Bit 2	Bit 10
3	Data Bit 3	Data Bit 3	MODEM Status	0	Parity Enable (PEN)	Out 2	Framing Error (FE)	Delta Data Carrier Detect (DDCD)	Bit 3	Bit 3	Bit 11
4	Data Bit 4	Data Bit 4	0	0	Even Parity Select (EPS)	Loop	Break Interrupt (BI)	Clear to Send (CTS)	Bit 4	Bit 4	Bit 12
5	Data Bit 5	Data Bit 5	0	0	Stick Parity	0	Transmitter Holding Register (THRE)	Data Set Ready (DSR)	Bit 5	Bit 5	Bit 13
6	Data Bit 6	Data Bit 6	0	0	Set Break	0	Transmitter Empty (TEMT)	Ring Indicator (RI)	Bit 6	Bit 6	Bit 14
7	Data Bit 7	Data Bit 7	0	0	Divisor Latch Access Bit (DLAB)	0	0	Data Carrier Detect (DCD)	Bit 7	Bit 7	Bit 15

Note 1: Bit 0 is the least significant bit. It is the first bit serially transmitted or received.

8.0 Registers (Continued)

bit 2 is a logic 1 when either a 6-, 7-, or 8-bit word length is selected, two Stop bits are generated. The Receiver checks the first Stop-bit only, regardless of the number of Stop bits selected.

Bit 3: This bit is the Parity Enable bit. When bit 3 is a logic 1, a Parity bit is generated (transmit data) or checked (receive data) between the last data word bit and Stop bit of the serial data. (The Parity bit is used to produce an even or odd number of 1s when the data word bits and the Parity bit are summed.)

Bit 4: This bit is the Even Parity Select bit. When bit 3 is a logic 1 and bit 4 is a logic 0, an odd number of logic 1s is transmitted or checked in the data word bits and Parity bit. When bit 3 is a logic 1 and bit 4 is a logic 1, an even number of logic 1s is transmitted or checked.

Bit 5: This bit is the Stick Parity bit. When bits 3, 4 and 5 are logic 1 the Parity bit is transmitted and checked as a logic 0. If bits 3 and 5 are 1 and bit 4 is a logic 0 then the Parity bit is transmitted and checked as a logic 1. If bit 5 is a logic 0 Stick Parity is disabled.

Bit 6: This bit is the Break Control bit. It causes a break condition to be transmitted by the UART. When it is set to a logic 1, the serial output (SOUT) is forced to the Spacing (logic 0) state. The break is disabled by clearing bit 6 to a logic 0. The Break Control bit acts only on SOUT and has no effect on the transmitter logic.

Note: This feature enables the CPU to alert a terminal in a computer communications system. If the following sequence is used, no erroneous or extraneous characters will be transmitted because of the break.

1. Load an all 0s, pad character, in response to THRE.
2. Set break after the next THRE.
3. Wait for the transmitter to be idle, (TEMT = 1), and clear break when normal transmission has to be restored.

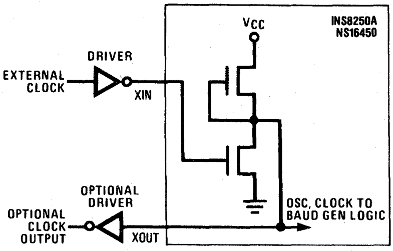
During the break, the Transmitter can be used as a character timer to accurately establish the break duration.

Bit 7: This bit is the Divisor Latch Access Bit (DLAB). It must be set high (logic 1) to access the Divisor Latches of the Baud Generator during a Read or Write operation. It must be set low (logic 0) to access the Receiver Buffer, the Transmitter Holding Register, or the Interrupt Enable Register.

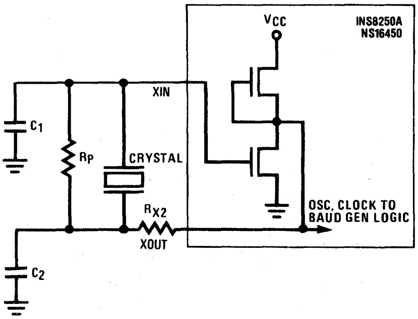
TABLE III. Baud Rates Using 1.8432 MHz Crystal

Desired Baud Rate	Decimal Divisor Used to Generate 16 x Clock	Percent Error Difference Between Desired and Actual
50	2304	—
75	1536	—
110	1047	0.026
134.5	857	0.058
150	768	—
300	384	—
600	192	—
1200	96	—
1800	64	—
2000	58	0.69
2400	48	—
3600	32	—
4800	24	—
7200	16	—
9600	12	—
19200	6	—
38400	3	—
56000	2	2.86

8.2 TYPICAL CLOCK CIRCUITS



TL/C/8401–12



TL/C/8401–13

Typical Oscillator Networks

Crystal	R_P	R_{X2}	C_1	C_2
1.8–3.1 MHz	1 MΩ	1.5k	10–30 pF	40–60 pF

TABLE IV. Baud Rates Using 3.072 MHz Crystal

Desired Baud Rate	Decimal Divisor Used to Generate 16 x Clock	Percent Error Difference Between Desired and Actual
50	3840	—
75	2560	—
110	1745	0.026
134.5	1428	0.034
150	1280	—
300	640	—
600	320	—
1200	160	—
1800	107	0.312
2000	96	—
2400	80	—
3600	53	0.628
4800	40	—
7200	27	1.23
9600	20	—
19200	10	—
38400	5	—

8.0 Registers (Continued)

8.3 PROGRAMMABLE BAUD GENERATOR

The UART contains a programmable Baud Generator that is capable of taking any clock input from DC to 3.1 MHz and dividing it by any divisor from 1 to $2^{16}-1$. The output frequency of the Baud Generator is 16 × the Baud [divisor # = (frequency input) ÷ (baud rate × 16)]. Two 8-bit latches store the divisor in a 16-bit binary format. These Divisor Latches must be loaded during initialization in order to ensure proper operation of the Baud Generator. Upon loading either of the Divisor Latches, a 16-bit Baud counter is immediately loaded.

Tables III and IV provide decimal divisors to use with crystal frequencies of 1.8432 MHz and 3.072 MHz respectively for common baud rates. For baud rates of 38400 and below, the error obtained is minimal. The accuracy of the desired baud rate is dependent on the crystal frequency chosen. Using a division of 0 is **not** recommended.

Note: The maximum operating frequency of the Baud Generator is 3.1 MHz. However, when using divisors of 3 and below, the maximum frequency is equal to the divisor in MHz. For example, if the divisor is 1, then the maximum frequency is 1 MHz. In no case should the data rate be greater than 56k Baud.

8.4 LINE STATUS REGISTER

This 8-bit register provides status information to the CPU concerning the data transfer. Table II shows the contents of the Line Status Register. Details on each bit follow:

Bit 0: This bit is the receiver Data Ready (DR) indicator. Bit 0 is set to a logic 1 whenever a complete incoming character has been received and transferred into the Receiver Buffer Register. Bit 0 is reset to a logic 0 by reading the data in the Receiver Buffer Register.

Bit 1: This bit is the Overrun Error (OE) indicator. Bit 1 indicates that data in the Receiver Buffer Register was not read by the CPU before the next character was transferred into the Receiver Buffer Register, thereby destroying the previous character. The OE indicator is set to a logic 1 upon detection of an overrun condition and reset whenever the CPU reads the contents of the Line Status Register.

Bit 2: This bit is the Parity Error (PE) indicator. Bit 2 indicates that the received data character does not have the correct even or odd parity, as selected by the even-parity-select bit. The PE bit is set to a logic 1 upon detection of a parity error and is reset to a logic 0 whenever the CPU reads the contents of the Line Status Register.

Bit 3: This bit is the Framing Error (FE) indicator. Bit 3 indicates that the received character did not have a valid Stop bit. Bit 3 is set to a logic 1 whenever the Stop bit following the last data bit or parity bit is a logic 0 (Spacing level). The FE indicator is reset whenever the CPU reads the contents of the Line Status Register. The UART will try to resynchronize after a framing error. To do this it assumes that the framing error was due to the next start bit, so it samples this "start" bit twice and then takes in the "data".

Bit 4: This bit is the Break Interrupt (BI) indicator. Bit 4 is set to a logic 1 whenever the received data input is held in the Spacing (logic 0) state for longer than a full word transmission time (that is, the total time of Start bit + data bits + Parity + Stop bits). The BI indicator is reset whenever the CPU reads the contents of the Line Status Register. Restarting after a break is received, requires the SIN pin to be logical 1 for at least ½ bit time.

Note: Bits 1 through 4 are the error conditions that produce a Receiver Line Status interrupt whenever any of the corresponding conditions are detected and the interrupt is enabled.

Bit 5: This bit is the Transmitter Holding Register Empty (THRE) indicator. Bit 5 indicates that the UART is ready to accept a new character for transmission. In addition, this bit causes the UART to issue an interrupt to the CPU when the Transmit Holding Register Empty Interrupt enable is set high. The THRE bit is set to a logic 1 when a character is transferred from the Transmitter Holding Register into the Transmitter Shift Register. The bit is reset to logic 0 whenever the CPU loads the Transmitter Holding Register.

Bit 6: This bit is the Transmitter Empty (TEMT) indicator. Bit 6 is set to a logic 1 whenever the Transmitter Holding Register (THR) and the Transmitter Shift Register (TSR) are both empty. It is reset to a logic 0 whenever either the THR or TSR contains a data character.

Bit 7: This bit is permanently set to logic 0.

Note: The Line Status Register is intended for read operations only. Writing to this register is not recommended as this operation is only used for factory testing.

TABLE V. Interrupt Control Functions

Interrupt Identification Register				Interrupt Set and Reset Functions		
Bit 2	Bit 1	Bit 0	Priority Level	Interrupt Type	Interrupt Source	Interrupt Reset Control
0	0	1	—	None	None	—
1	1	0	Highest	Receiver Line Status	Overrun Error or Parity Error or Framing Error or Break Interrupt	Reading the Line Status Register
1	0	0	Second	Received Data Available	Receiver Data Available	Reading the Receiver Buffer Register
0	1	0	Third	Transmitter Holding Register Empty	Transmitter Holding Register Empty	Reading the IIR Register (if source of interrupt) or Writing into the Transmitter Holding Register
0	0	0	Fourth	MODEM Status	Clear to Send or Data Set Ready or Ring Indicator or Data Carrier Detect	Reading the MODEM Status Register

8.0 Registers (Continued)

8.5 INTERRUPT IDENTIFICATION REGISTER

In order to provide minimum software overhead during data character transfers, the UART prioritizes interrupts into four levels and records these in the Interrupt Identification Register. The four levels of interrupt conditions in order of priority are Receiver Line Status; Received Data Ready; Transmitter Holding Register Empty; and MODEM Status.

When the CPU accesses the IIR, the UART freezes all interrupts and indicates the highest priority pending interrupt to the CPU. While this CPU access is occurring, the UART records new interrupts, but does not change its current indication until the access is complete. Table II shows the contents of the IIR. Details on each bit follow:

Bit 0: This bit can be used in an interrupt environment to indicate whether an interrupt condition is pending. When bit 0 is a logic 0, an interrupt is pending and the IIR contents may be used as a pointer to the appropriate interrupt service routine. When bit 0 is a logic 1, no interrupt is pending.

Bits 1 and 2: These two bits of the IIR are used to identify the highest priority interrupt pending as indicated in Table V.

Bits 3 through 7: These five bits of the IIR are always logic 0.

8.6 INTERRUPT ENABLE REGISTER

This register enables the four types of UART interrupts. Each interrupt can individually activate the interrupt (INTR) output signal. It is possible to totally disable the interrupt system by resetting bits 0 through 3 of the Interrupt Enable Register (IER). Similarly, setting bits of this register to a logic 1, enables the selected interrupt(s). Disabling an interrupt prevents it from being indicated as active in the IIR and from activating the INTR output signal. All other system functions operate in their normal manner, including the setting of the Line Status and MODEM Status Registers. Table II shows the contents of the IER. Details on each bit follow.

Bit 0: This bit enables the Received Data Available Interrupt when set to logic 1.

Bit 1: This bit enables the Transmitter Holding Register Empty Interrupt when set to logic 1.

Bit 2: This bit enables the Receiver Line Status Interrupt when set to logic 1.

Bit 3: This bit enables the MODEM Status Interrupt when set to logic 1.

Bits 4 through 7: These four bits are always logic 0.

8.7 MODEM CONTROL REGISTER

This register controls the interface with the MODEM or data set (or a peripheral device emulating a MODEM). The contents of the MODEM Control Register (MCR) are indicated in Table II and are described below. Table II shows the contents of the MCR. Details on each bit follow.

Bit 0: This bit controls the Data Terminal Ready ($\overline{\text{DTR}}$) output. When bit 0 is set to a logic 1, the $\overline{\text{DTR}}$ output is forced to a logic 0. When bit 0 is reset to a logic 0, the $\overline{\text{DTR}}$ output is forced to a logic 1.

Note: The $\overline{\text{DTR}}$ output of the UART may be applied to an EIA inverting line driver (such as the DS1488) to obtain the proper polarity input at the succeeding MODEM or data set.

Bit 1: This bit controls the Request to Send ($\overline{\text{RTS}}$) output. Bit 1 affects the $\overline{\text{RTS}}$ output in a manner identical to that described above for bit 0.

Bit 2: This bit controls the Output 1 ($\overline{\text{OUT 1}}$) signal, which is an auxiliary user-designated output. Bit 2 affects the $\overline{\text{OUT 1}}$ output in a manner identical to that described above for bit 0.

Bit 3: This bit controls the Output 2 ($\overline{\text{OUT 2}}$) signal, which is an auxiliary user-designated output. Bit 3 affects the $\overline{\text{OUT 2}}$ output in a manner identical to that described above for bit 0.

Bit 4: This bit provides a local loopback feature for diagnostic testing of the UART. When bit 4 is set to logic 1, the following occur: the transmitter Serial Output (SOUT) is set to the Marking (logic 1) state; the receiver Serial Input (SIN) is disconnected; the output of the Transmitter Shift Register is "looped back" into the Receiver Shift Register input; the four MODEM Control inputs ($\overline{\text{CTS}}$, $\overline{\text{DSR}}$, $\overline{\text{RI}}$, and $\overline{\text{DCD}}$) are disconnected; and the four MODEM Control outputs ($\overline{\text{DTR}}$, $\overline{\text{RTS}}$, $\overline{\text{OUT 1}}$, and $\overline{\text{OUT 2}}$) are internally connected to the four MODEM Control inputs. The MODEM Control output pins are forced to their inactive state (high). In the diagnostic mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit-and received-data paths of the UART.

In the diagnostic mode, the receiver and transmitter interrupts are fully operational. The MODEM Control Interrupts are also operational, but the interrupts' sources are now the lower four bits of the MODEM Control Register instead of the four MODEM Control inputs. The interrupts are still controlled by the Interrupt Enable Register.

Bits 5 through 7: These bits are permanently set to logic 0.

8.8 MODEM STATUS REGISTER

This register provides the current state of the control lines from the MODEM (or peripheral device) to the CPU. In addition to this current-state information, four bits of the MODEM Status Register provide change information. These bits are set to a logic 1 whenever a control input from the MODEM changes state. They are reset to logic 0 whenever the CPU reads the MODEM Status Register.

8.0 Registers (Continued)

Table II shows the contents of the MSR. Details on each bit follow.

Bit 0: This bit is the Delta Clear to Send (DCTS) indicator. Bit 0 indicates that the $\overline{\text{CTS}}$ input to the chip has changed state since the last time it was read by the CPU.

Bit 1: This bit is the Delta Data Set Ready (DDSR) indicator. Bit 1 indicates that the $\overline{\text{DSR}}$ input to the chip has changed state since the last time it was read by the CPU.

Bit 2: This bit is the Trailing Edge of Ring Indicator (TERI) detector. Bit 2 indicates that the $\overline{\text{RI}}$ input to the chip has changed from a low to a high state.

Bit 3: This bit is the Delta Data Carrier Detect (DDCD) indicator. Bit 3 indicates that the $\overline{\text{DCD}}$ input to the chip has changed state.

Note: Whenever bit 0, 1, 2, or 3 is set to logic 1, a MODEM Status Interrupt is generated.

Bit 4: This bit is the complement of the Clear to Send ($\overline{\text{CTS}}$) input. If bit 4 (loop) of the MCR is set to a 1, this bit is equivalent to RTS in the MCR.

Bit 5: This bit is the complement of the Data Set Ready ($\overline{\text{DSR}}$) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to DTR in the MCR.

Bit 6: This bit is the complement of the Ring Indicator ($\overline{\text{RI}}$) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to OUT 1 in the MCR.

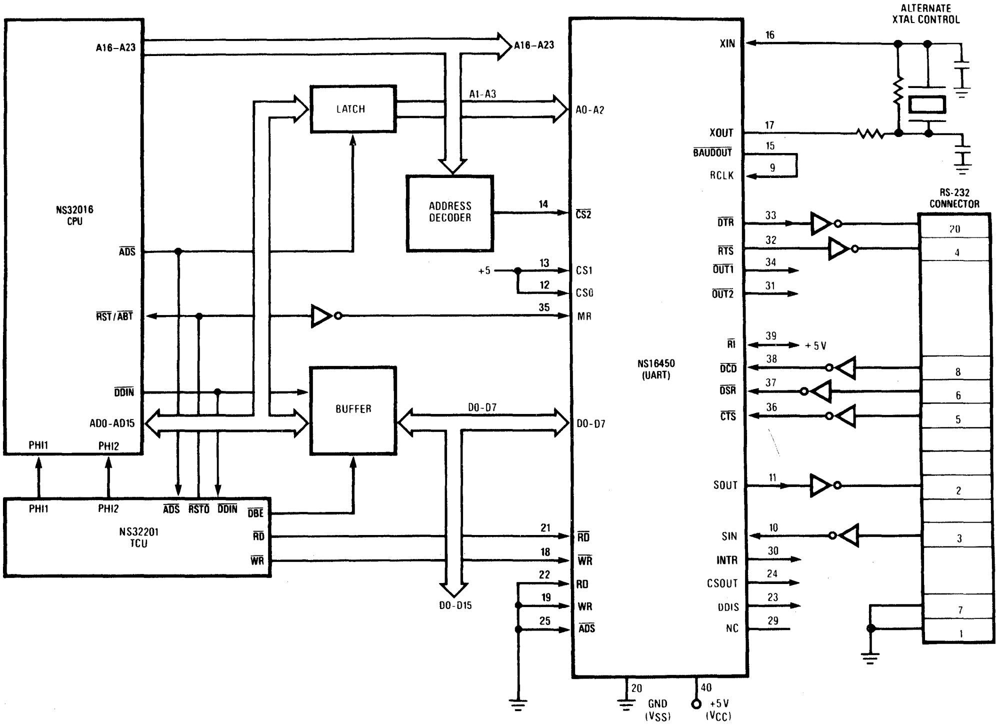
Bit 7: This bit is the complement of the Data Carrier Detect ($\overline{\text{DCD}}$) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to OUT 2 in the MCR.

8.9 SCRATCHPAD REGISTER

This 8-bit Read/Write Register does not control the UART in any way. It is intended as a scratchpad register to be used by the programmer to hold data temporarily.

9.0 Typical Applications

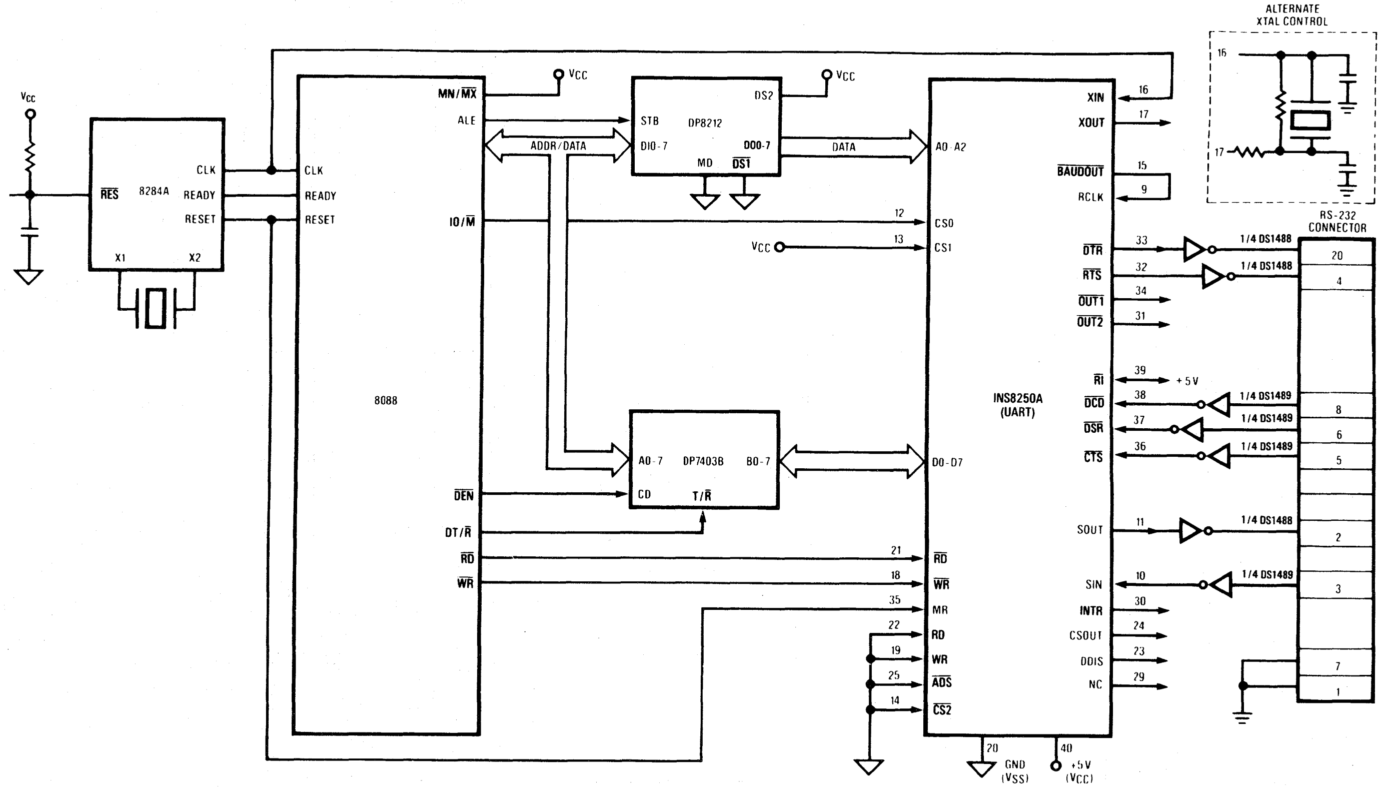
This shows the basic connections of an NS16450 to an NS32016 CPU



TL/C/8401–14

9.0 Typical Applications (Continued)

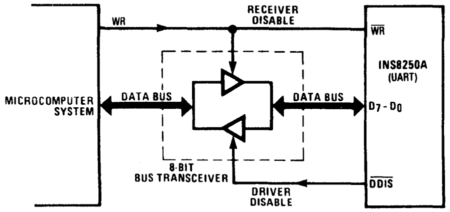
Typical shows the basic connections of an INS8250A to an 8088 CPU



TL/C/8401-15

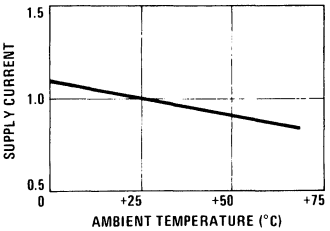
9.0 Typical Applications (Continued)

Typical Interface for a High-Capacity Data Bus



TL/C/8401-16

Typical Supply Current vs Temperature, Normalized



TL/C/8401-17

10.0 Ordering Information

Order Number	Description
Plastic Dip Package	
NS16450N or NS-16450N	high speed part
INS8250AN	V_{CC} = 5V ±5%
NS16C450N	CMOS high speed part
INS82C50AN	CMOS V_{CC} = 5V ±5%
Plastic Chip Carrier Package	
NS16450V or NS-16450V	high speed part
INS8250A	V_{CC} = 5V ± 5%
NS16C450V	CMOS high speed part
INS82C50AV	CMOS V_{CC} = 5V ±5%

11.0 Reliability Information

Gate Count	
XMOS	2,000
CMOS	1,600
Transistor Count	
XMOS	4,500
CMOS	6,300

National Semiconductor Corporation

INS8250, INS8250-B Universal Asynchronous Receiver/Transmitter

General Description

Each of these parts function as a serial data input/output interface in a microcomputer system. The system software determines the functional configuration of the UART via a TRI-STATE® 8-bit bidirectional data bus.

The UART performs serial-to-parallel conversion on data characters received from a peripheral device or a MODEM, and parallel-to-serial conversion on data characters received from the CPU. The CPU can read the complete status of the UART. Status information reported includes the type and condition of the transfer operations being performed by the UART, as well as any error conditions (parity, overrun, framing, or break interrupt).

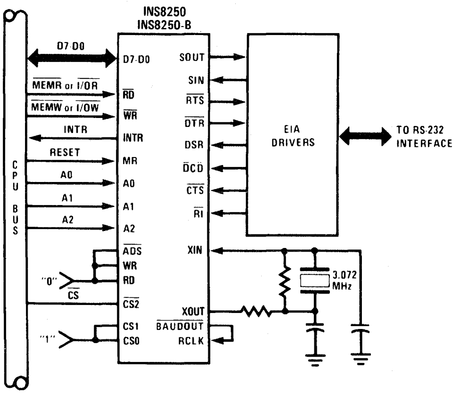
The UART includes a programmable baud rate generator that is capable of dividing the timing reference clock input by divisors of 1 to ($2^{16}-1$), and producing a 16 × clock for driving the internal transmitter logic. Provisions are also included to use this 16 × clock to drive the receiver logic. The UART includes a complete MODEM-control capability and a processor-interrupt system. Interrupts can be programmed to the user's requirements minimizing the computing required to handle the communications link.

National's INS8250 universal asynchronous receiver transmitter (UART) is the unanimous choice of almost every PC and add-on manufacturer in the world. The INS8250 is a programmable communications chip available in a standard 40-pin dual-in-line and a 44-pin PCC package. The chip is fabricated using N-channel silicon gate technology.

Features

- Easily interfaces to most popular microprocessors.
- Adds or deletes standard asynchronous communication bits (start, stop, and parity) to or from serial data stream.
- Holding and shift registers eliminate the need for precise synchronization between the CPU and the serial data.
- Independently controlled transmit, receive, line status, and data set interrupts.
- Programmable baud generator allows division of any input clock by 1 to (2^{16} − 1) and generates the internal 16 × clock.
- Independent receiver clock input.
- MODEM control functions (CTS, RTS, DSR, DTR, RI, and DCD).
- Fully programmable serial-interface characteristics:
  - — 5-, 6-, 7-, or 8-bit characters
  - — Even, odd, or no-parity bit generation and detection
  - — 1-, 1½-, or 2-stop bit generation
  - — Baud generation (DC to 56k baud).
- False start bit detection.
- Complete status reporting capabilities.
- TRI-STATE TTL drive capabilities for bidirectional data bus and control bus.
- Line break generation and detection.
- Internal diagnostic capabilities:
  - — Loopback controls for communications link fault isolation
  - — Break, parity, overrun, framing error simulation.
- Fully prioritized interrupt system controls.

Connection Diagram



TL/C/9329-1

Table of Contents

1.0 Absolute Maximum Ratings

If Military/Aerospace specified devices are required, contact the National Semiconductor Sales Office/ Distributors for availability and specifications.

Temperature Under Bias	0°C to +70°C
Storage Temperature	−65°C to +150°C
All Input or Output Voltages with Respect to V_{SS}	−0.5V to +7.0V
Power Dissipation	400 mW

Note: *Maximum ratings indicate limits beyond which permanent damage may occur. Continuous operation at these limits is not intended and should be limited to those conditions specified under DC electrical characteristics.*

2.0 DC Electrical Characteristics

T_A = 0°C to +70°C, V_{CC} = +5V ±5%, V_{SS} = 0V, unless otherwise specified.

Symbol	Parameter	Conditions	INS8250		INS8250-B		Units
			Min	Max	Min	Max	
V_{ILX}	Clock Input Low Voltage		−0.5	0.8	−0.5	0.8	V
V_{IHX}	Clock Input High Voltage		2.0	V_{CC}	2.0	V_{CC}	V
V_{IL}	Input Low Voltage		−0.5	0.8	−0.5	0.8	V
V_{IH}	Input High Voltage		2.0	V_{CC}	2.0	V_{CC}	V
V_{OL}	Output Low Voltage	I_{OL} = 1.6 mA on all (Note 1)		0.4		0.4	V
V_{OH}	Output High Voltage	I_{OH} = −1.0 mA (Note 1)	2.4		2.4		V
I_{CC}(AV)	Avg. Power Supply Current (V_{CC})	V_{CC} = 5.25V, T_A = 25°C No Loads on output SIN, DSR, DCD, CTS, RI = 2.4V All other inputs = 0.4V		80		80	mA
I_{IL}	Input Leakage	V_{CC} = 5.25V, V_{SS} = 0V All other pins floating. V_{IN} = 0V, 5.25V		±10		±10	μA
I_{CL}	Clock Leakage			±10		±10	μA
I_{OZ}	TRI-STATE Leakage	V_{CC} = 5.25V, V_{SS} = 0V V_{OUT} = 0V, 5.25V 1) Chip deselected 2) WRITE mode, chip selected		±20		±20	μA

Capacitance T_A = 25°C, V_{CC} = V_{SS} = 0V

Symbol	Parameter	Conditions	Min	Typ	Max	Units
C_{XIN}	Clock Input Capacitance	f_c = 1 MHz Unmeasured pins returned to V_{SS}		15	20	pF
C_{XOUT}	Clock Output Capacitance			20	30	pF
C_{IN}	Input Capacitance			6	10	pF
C_{OUT}	Output Capacitance			10	20	pF

Note 1: Does not apply to XOUT.

3.0 AC Electrical Characteristics T_A = 0°C to +70°C, V_{CC} = +5V ±5%

Symbol	Parameter	Conditions	INS8250		INS8250-B		Units
			Min	Max	Min	Max	
t_{ADS}	Address Strobe Width		90		120		ns
t_{AH}	Address Hold Time		0		60		ns
t_{AR}	$\overline{\text{RD}}$/RD Delay from Address	(Note 1)	110		110		ns
t_{AS}	Address Setup Time		110		110		ns
t_{CH}	Chip Select Hold Time		0		60		ns
t_{CS}	Chip Select Setup Time		110		110		ns
t_{CSC}	Chip Select Output Delay from Select	@100 pF loading (Note 1)		200		200	ns
t_{CSR}	$\overline{\text{RD}}$/RD Delay from Chip Select	(Note 1)	110		110		ns
t_{CSS}	Chip Select Output Delay from Strobe		0	150	0	150	ns
t_{CSW}	$\overline{\text{WR}}$/WR Delay from Select	(Note 1)	160		160		ns
t_{DH}	Data Hold Time		60		100		ns
t_{DS}	Data Setup Time		175		350		ns
t_{HZ}	$\overline{\text{RD}}$/RD to Floating Data Delay	@100 pF loading (Note 3)	0	150	0	150	ns
t_{MR}	Master Reset Pulse Width		10		10		µs
t_{RA}	Address Hold Time from $\overline{\text{RD}}$/RD	(Note 1)	50		50		ns
t_{RC}	Read Cycle Delay		1735		1735		ns
t_{RCS}	Chip Select Hold Time from $\overline{\text{RD}}$/RD	(Note 1)	50		50		ns
t_{RD}	$\overline{\text{RD}}$/RD Strobe Width		175		350		ns
t_{RDA}	Read Strobe Delay		0		0		ns
t_{RDD}	$\overline{\text{RD}}$/RD to Driver Disable Delay	@100 pF loading (Note 3)		150		250	ns
t_{RVD}	Delay from $\overline{\text{RD}}$/RD to Data	@100 pF loading		250		300	ns
t_{WA}	Address Hold Time from $\overline{\text{WR}}$/WR	(Note 1)	50		50		ns
t_{WC}	Write Cycle Delay		1785		1785		ns
t_{WCS}	Chip Select Hold Time from $\overline{\text{WR}}$/WR	(Note 1)	50		50		ns
t_{WDA}	Write Strobe Delay		50		50		ns
t_{WR}	$\overline{\text{WR}}$/WR Strobe Width		175		350		ns
t_{XH}	Duration of Clock High Pulse	External Clock (3.1 MHz Max.)	140		140		ns
t_{XL}	Duration of Clock Low Pulse	External Clock (3.1 MHz Max.)	140		140		ns
RC	Read Cycle = $t_{AR} + t_{DIW} + t_{RC}$		2000		2205		ns
WC	Write Cycle = $t_{DDA} + t_{DOW} + t_{WC}$		2100		2305		ns
Baud Generator							
N	Baud Divisor		1	$2^{16}-1$	1	$2^{16}-1$	
t_{BHD}	Baud Output Positive Edge Delay	100 pF Load		250		250	ns
t_{BLD}	Baud Output Negative Edge Delay	100 pF Load		250		250	ns
t_{HW}	Baud Output Up Time	f_X = 3 MHz, ÷3, 100 pF Load	330		330		ns
t_{LW}	Baud Output Down Time	f_X = 2 MHz, ÷2, 100 pF Load	425		425		ns
Receiver							
t_{RINT}	Delay from $\overline{\text{RD}}$/RD (RD RBR or RD LSR) to Reset Interrupt	100 pF Load		1000		1000	ns
t_{SCD}	Delay from RCLK to Sample Time			2000		2000	ns
t_{SINT}	Delay from Stop to Set Interrupt			2000		2000	ns

Note 1: Applicable only when $\overline{\text{ADS}}$ is tied low.

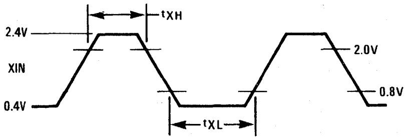
Note 2: Charge and discharge time is determined by V_{OL}, V_{OH} and the external loading.

3.0 AC Electrical Characteristics $T_A = 0°C$ to $+70°C$, $V_{CC} = +5V \pm 5\%$ (Continued)

Symbol	Parameter	Conditions	INS8250		INS82C50-B		Units
			Min	Max	Min	Max	
Transmitter							
t_{HR}	Delay from $\overline{WR}$/WR (WR THR) to Reset Interrupt	100 pF Load		1000		1000	ns
t_{IR}	Delay from $\overline{RD}$/RD (RD IIR) to Reset Interrupt (THRE)	100 pF Load		1000		1000	ns
t_{IRS}	Delay from Initial INTR Reset to Transmit Start			16		16	BAUDOUT Cycles
t_{SI}	Delay from Initial Write to Interrupt			50		50	BAUDOUT Cycles
t_{SS}	Delay from Stop to Next Start			1000		1000	ns
t_{STI}	Delay from Stop to Interrupt (THRE)			8		8	BAUDOUT Cycles
Modem Control							
t_{MDO}	Delay from $\overline{WR}$/WR (WR MCR) to Output	100 pF Load		1000		1000	ns
t_{RIM}	Delay to Reset Interrupt from $\overline{RD}$/RD (RD MSR)	100 pF Load		1000		1000	ns
t_{SIM}	Delay to Set Interrupt from MODEM Input	100 pF Load		1000		1000	ns

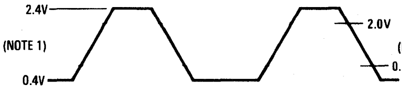
4.0 Timing Waveforms (All timings are referenced to valid 0 and valid 1)

External Clock Input (3.1 MHz Max.)



TL/C/9329-2

AC Test Points

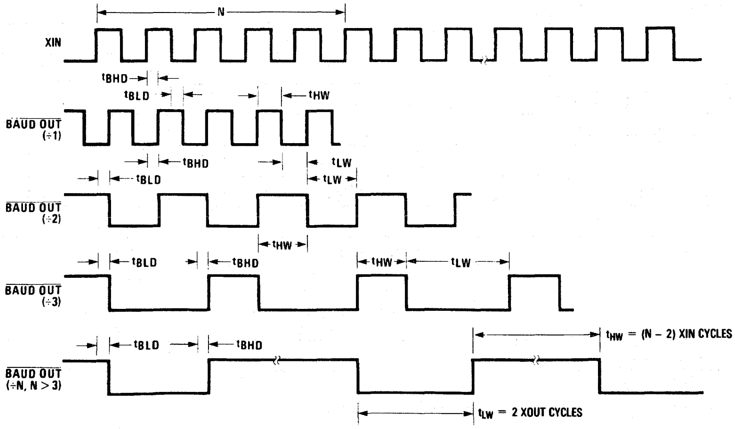


TL/C/9329-3

Note 1: The 2.4V and 0.4V levels are the voltages that the inputs are driven to during AC testing.

Note 2: The 2.0V and 0.8V levels are the voltages at which the timing tests are made.

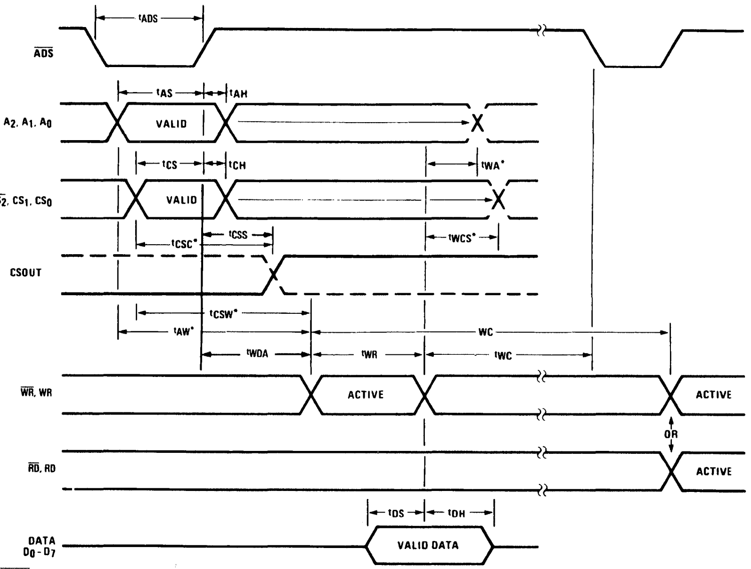
$\overline{\text{BAUDOUT}}$ Timing



TL/C/9329-4

4.0 Timing Waveforms (Continued)

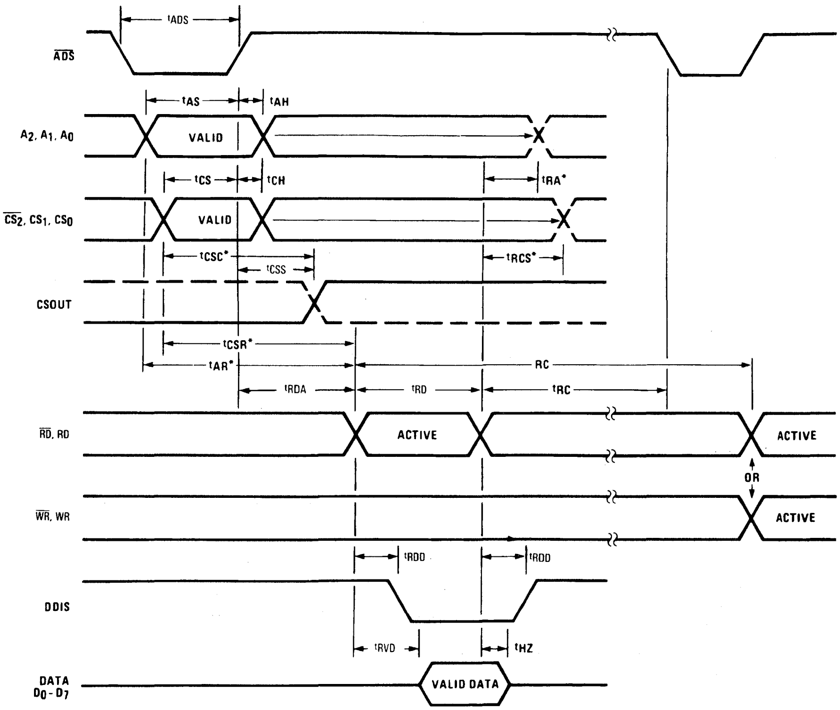
Write Cycle



*Applicable Only When $\overline{\text{ADS}}$ is Tied Low.

TL/C/9329–5

Read Cycle

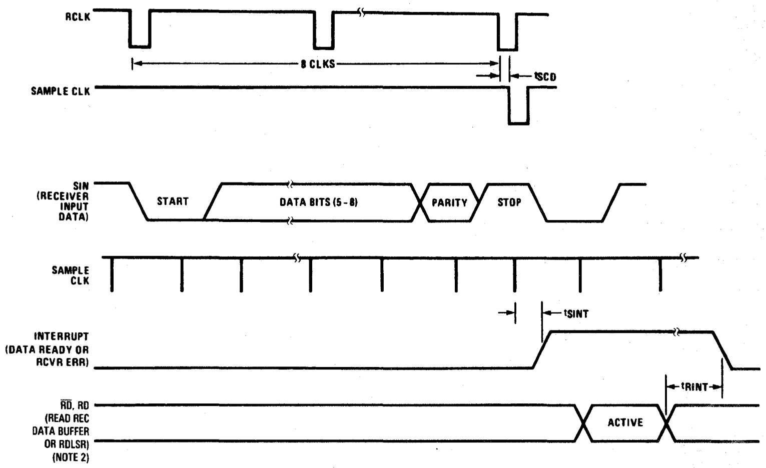


*Applicable Only When $\overline{\text{ADS}}$ is Tied Low.

TL/C/9329–6

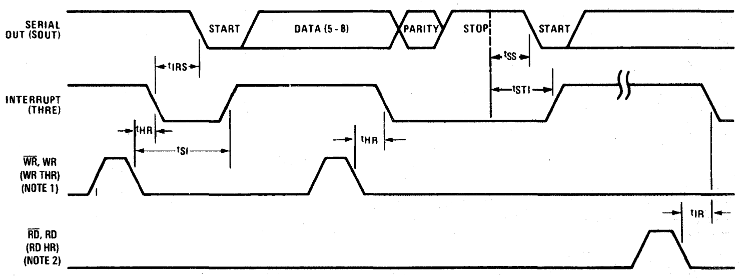
4.0 Timing Waveforms (Continued)

Receiver Timing


RCLK
8 CLKS
tSCD
SAMPLE CLK
SIN (RECEIVER INPUT DATA)
START
DATA BITS (5-8)
PARITY
STOP
SAMPLE CLK
tSINT
INTERRUPT (DATA READY OR RCVR ERR)
tRINT
RD, RD (READ REC DATA BUFFER OR RDLSR) (NOTE 2)
ACTIVE

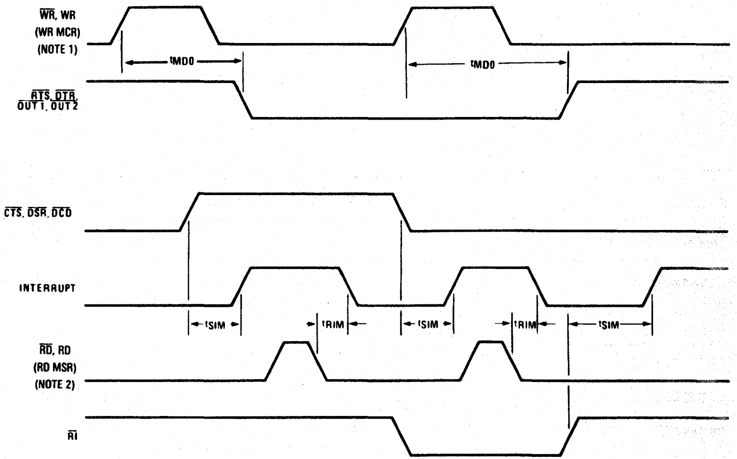
TL/C/9329-7

Transmitter Timing



TL/C/9329-8

MODEM Controls Timing

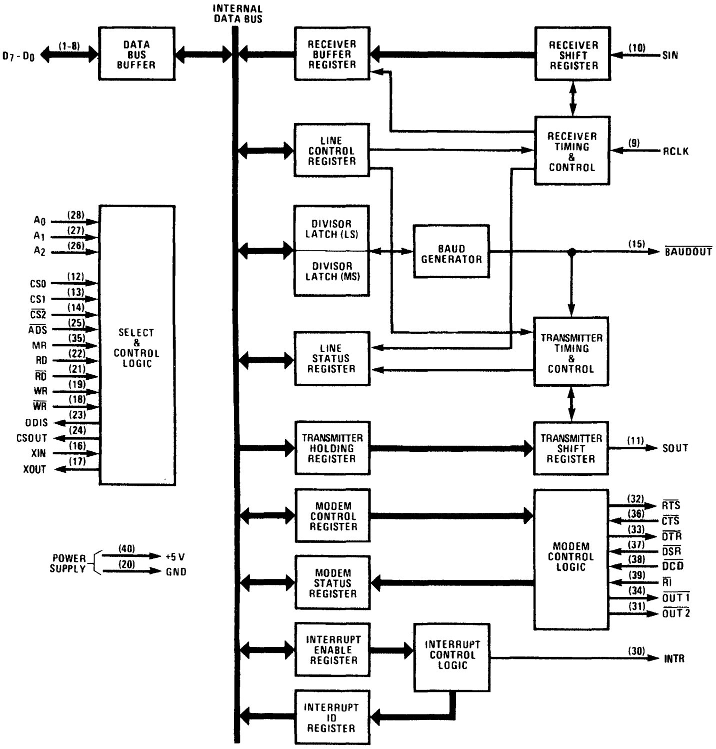


TL/C/9329-9

Note 1: See Write Cycle Timing

Note 2: See Read Cycle Timing

5.0 Block Diagram



TL/C/9329-10

Note: Applicable pinout numbers are included within parenthesis.

6.0 Pin Descriptions

The following describes the function of all UART, pins. Some of these descriptions reference internal circuits.

In the following descriptions, a low represents a logic 0 (0V nominal) and a high represents a logic 1 (+2.4V nominal).

6.1 INPUT SIGNALS

Chip Select (CS0, CS1, $\overline{\text{CS2}}$), Pins 12–14: When CS0 and CS1 are high and $\overline{\text{CS2}}$ is low, the chip is selected. This enables communication between the UART and the CPU. The positive edge of an active Address Strobe signal latches the decoded chip select signals, completing chip selection. If $\overline{\text{ADS}}$ is always low valid chip selects should stabilize according to the t_{CSW} parameter.

Read (RD, $\overline{\text{RD}}$), Pins 22 and 21: When RD is high or $\overline{\text{RD}}$ is low while the chip is selected, the CPU can read status information or data from the selected UART register.

Note: Only an active RD or $\overline{\text{RD}}$ input is required to transfer data from the UART during a read operation. Therefore, tie either the RD input permanently low or the $\overline{\text{RD}}$ input permanently high, when it is not used.

Write (WR, $\overline{\text{WR}}$), Pins 19 and 18: When WR is high or $\overline{\text{WR}}$ is low while the chip is selected, the CPU can write control words or data into the selected UART register.

Note: Only an active WR or $\overline{\text{WR}}$ input is required to transfer data to the UART during a write operation. Therefore, tie either the WR input permanently low or the $\overline{\text{WR}}$ input permanently high, when it is not used.

Address Strobe ($\overline{\text{ADS}}$), Pin 25: The positive edge of an active Address Strobe ($\overline{\text{ADS}}$) signal latches the Register Select (A0, A1, A2) and Chip Select (CS0, CS1, CS2) signals.

Note: An active $\overline{\text{ADS}}$ input is required when the Register Select (A0, A1, A2) signals are not stable for the duration of a read or write operation. If not required, tie the $\overline{\text{ADS}}$ input permanently low.

Register Select (A0, A1, A2), Pins 26–28: Address signals connected to these 3 inputs select a UART register for the CPU to read from or write to during data transfer. A table of registers and their addresses is shown below. Note that the state of the Divisor Latch Access Bit (DLAB), which is the most significant bit of the Line Control Register, affects the selection of certain UART registers. The DLAB must be set high by the system software to access the Baud Generator Divisor Latches.

6.0 Pin Descriptions (Continued)

DLAB	A_2	A_1	A_0	Register
0	0	0	0	Receiver Buffer (read), Transmitter Holding Register (write)
0	0	0	1	Interrupt Enable
X	0	1	0	Interrupt Identification (read only)
X	0	1	1	Line Control
X	1	0	0	MODEM Control
X	1	0	1	Line Status
X	1	1	0	MODEM Status
1	0	0	0	Divisor Latch (least significant byte)
1	0	0	1	Divisor Latch (most significant byte)

Register Addresses

Master Reset (MR), Pin 35: When this input is high, it clears all the registers (except the Receiver Buffer, Transmitter Holding, and Divisor Latches), and the control logic of the UART. The states of various output signals (SOUT, INTR, $\overline{\text{OUT 1}}$, $\overline{\text{OUT 2}}$, $\overline{\text{RTS}}$, $\overline{\text{DTR}}$) are affected by an active MR input. (Refer to Table I.).

Receiver Clock (RCLK), Pin 9: This input is the 16 × baud rate clock for the receiver section of the chip.

Serial Input (SIN), Pin 10: Serial data input from the communications link (peripheral device, MODEM, or data set).

Clear to Send ($\overline{\text{CTS}}$), Pin 36: When low, this indicates that the MODEM or data set is ready to exchange data. The $\overline{\text{CTS}}$ signal is a MODEM status input whose conditions can be tested by the CPU reading bit 4 (CTS) of the MODEM Status Register. Bit 4 is the complement of the $\overline{\text{CTS}}$ signal. Bit 0 (DCTS) of the MODEM Status Register indicates whether the $\overline{\text{CTS}}$ input has changed state since the previous reading of the MODEM Status Register. $\overline{\text{CTS}}$ has no effect on the Transmitter.

Note: Whenever the CTS bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.

Data Set Ready ($\overline{\text{DSR}}$), Pin 37: When low, this indicates that the MODEM or data set is ready to establish the communications link with the UART. The $\overline{\text{DSR}}$ signal is a MODEM status input whose condition can be tested by the CPU reading bit 5 (DSR) of the MODEM Status Register. Bit 5 is the complement of the $\overline{\text{DSR}}$ signal. Bit 1 (DDSR) of the MODEM Status Register indicates whether the $\overline{\text{DSR}}$ input has changed state since the previous reading of the MODEM Status Register.

Note: Whenever the DSR bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.

Data Carrier Detect ($\overline{\text{DCD}}$), Pin 38: When low, indicates that the data carrier has been detected by the MODEM or data set. The $\overline{\text{DCD}}$ signal is a MODEM status input whose condition can be tested by the CPU reading bit 7 (DCD) of the MODEM Status Register. Bit 7 is the complement of the $\overline{\text{DCD}}$ signal. Bit 3 (DDCD) of the MODEM Status Register indicates whether the $\overline{\text{DCD}}$ input has changed state since the previous reading of the MODEM Status Register. $\overline{\text{DCD}}$ has no effect on the receiver.

Note: Whenever the DCD bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.

Ring Indicator ($\overline{\text{RI}}$), Pin 39: When low, this indicates that a telephone ringing signal has been received by the MODEM or data set. The $\overline{\text{RI}}$ signal is a MODEM status input whose condition can be tested by the CPU reading bit 6 (RI) of the MODEM Status Register. Bit 6 is the complement of the $\overline{\text{RI}}$ signal. Bit 2 (TERI) of the MODEM Status Register indicates whether the $\overline{\text{RI}}$ input **signal** has changed from a low to a high state since the previous reading of the MODEM Status Register.

Note: Whenever the RI **bit** of the MODEM Status Register changes from a high to a low state, an interrupt is generated if the MODEM Status Interrupt is enabled.

V_{CC}, Pin 40: +5V supply.

V_{SS}, Pin 20: Ground (0V) reference.

6.2 OUTPUT SIGNALS

Data Terminal Ready ($\overline{\text{DTR}}$), Pin 33: When low, this informs the MODEM or data set that the UART is ready to establish a communications link. The $\overline{\text{DTR}}$ output signal can be set to an active low by programming bit 0 (DTR) of the MODEM Control Register to a high level. A Master Reset operation sets this signal to its inactive (high) state.

Request to Send ($\overline{\text{RTS}}$), Pin 32: When low, this informs the MODEM or data set that the UART is ready to exchange data. The $\overline{\text{RTS}}$ output signal can be set to an active low by programming bit 1 (RTS) of the MODEM Control Register. A Master Reset operation sets this signal to its inactive (high) state.

Output 1 ($\overline{\text{OUT 1}}$), Pin 34: This user-designated output can be set to an active low by programming bit 2 (OUT 1) of the MODEM Control Register to a high level. A Master Reset operation sets this signal to its inactive (high) state. In the XMOS parts this will achieve TTL levels.

Output 2 ($\overline{\text{OUT 2}}$), Pin 31: This user-designated output can be set to an active low by programming bit 3 (OUT 2) of the MODEM Control Register to a high level. A Master Reset operation sets this signal to its inactive (high) state. In the XMOS parts this will achieve TTL levels.

Chip Select Out (CSOUT), Pin 24: When high, it indicates that the chip has been selected by active, CS0, CS1, and $\overline{\text{CS2}}$ inputs. No data transfer can be initiated until the CSOUT signal is a logic 1. CSOUT goes low when the UART is deselected.

Driver Disable (DDIS), Pin 23: This goes low whenever the CPU is reading data from the UART. It can disable or control the direction of a data bus transceiver between the CPU and the UART (see Typical Interface for a High Capacity Data Bus).

Baud Out ($\overline{\text{BAUDOUT}}$), Pin 15: This is the 16 × clock signal from the transmitter section of the UART. The clock rate is equal to the main reference oscillator frequency divided by the specified divisor in the Baud Generator Divisor Latches. The $\overline{\text{BAUDOUT}}$ may also be used for the receiver section by tying this output to the RCLK input of the chip.

6.0 Pin Descriptions (Continued)

Interrupt (INTR), Pin 30: This goes high whenever any one of the following interrupt types has an active high condition and is enabled via the IER: Receiver Line Status; Received Data Available; Transmitter Holding Register Empty; and MODEM Status. The INTR signal is reset low upon the appropriate interrupt service or a Master Reset operation.

Serial Output (SOUT), Pin 11: This is the composite serial data output to the communications link (peripheral, MODEM or data set). The SOUT signal is set to the Marking (logic 1) state upon a Master Reset operation or when the transmitter is idle.

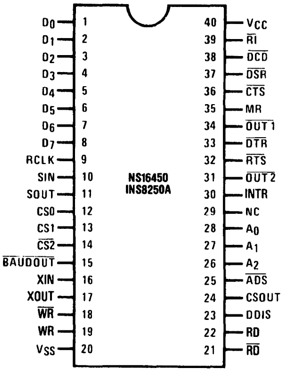
6.3 INPUT/OUTPUT SIGNALS

Data (D_7–D_0) Bus, Pins 1–8: This bus is comprised of eight TRI-STATE input/output lines. The bus provides bidirectional communications between the UART and the CPU. Data, control words, and status information are transferred via the D_7–D_0 Data Bus.

External Clock Input/Output (X_{IN}, X_{OUT}) Pins 16 and 17: These two pins connect the main timing reference (crystal or signal clock) to the UART. When a crystal oscillator or a clock signal is provided, it drives the UART via XIN (see typical oscillator network illustration).

7.0 Connection Diagrams

Dual-In-Line Package

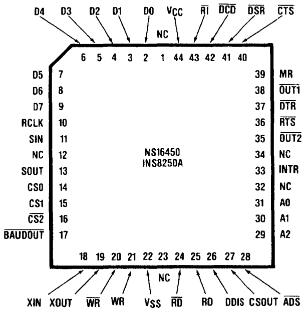


TL/C/9329-11

Top View

Order Number INS8250N, INS8250N-B or INS8250N/A+
See NS Package Number N40A

PCC Package



TL/C/9329-18

Top View

Order Number INS8250V-B
See NS Package Number V44A

TABLE I. UART Reset Functions

Register/Signal	Reset Control	Reset State
Interrupt Enable Register	Master Reset	**0000** 0000 (Note 1)
Interrupt Identification Register	Master Reset	**00000** 001
Line Control Register	Master Reset	0000 0000
MODEM Control Register	Master Reset	**0000** 0000
Line Status Register	Master Reset	**0**110 0000
MODEM Status Register	Master Reset	XXXX 0000 (Note 2)
SOUT	Master Reset	High
INTR (RCVR Errs)	Read LSR/MR	Low
INTR (RCVR Data Ready)	Read RBR/MR	Low
INTR (THRE)	Read IIR/Write THR/MR	Low
INTR (Modem Status Changes)	Read MSR/MR	Low
$\overline{\text{OUT 2}}$	Master Reset	High
$\overline{\text{RTS}}$	Master Reset	High
$\overline{\text{DTR}}$	Master Reset	High
$\overline{\text{OUT 1}}$	Master Reset	High

Note 1: Underlined bits are permanently low.

Note 2: Bits 7–4 are driven by the input signals.

8.0 Registers

The system programmer may access any of the UART registers summarized in Table II via the CPU. These registers control UART operations including transmission and reception of data. Each register bit in Table II has its name and reset state shown.

8.1 LINE CONTROL REGISTER

The system programmer specifies the format of the asynchronous data communications exchange and sets the Divisor Latch Access bit via the Line Control Register (LCR). The programmer can also read the contents of the Line Control Register. The read capability simplifies system programming and eliminates the need for separate storage in system memory of the line characteristics. Table II shows the contents of the LCR. Details on each bit follow:

Bits 0 and 1: These two bits specify the number of bits in each transmitted or received serial character. The encoding of bits 0 and 1 is as follows:

Bit 1	Bit 0	Character Length
0	0	5 Bits
0	1	6 Bits
1	0	7 Bits
1	1	8 Bits

Bit 2: This bit specifies the number of Stop bits transmitted and recevied in each serial character. If bit 2 is a logic 0, one Stop bit is generated or checked in the serial data. If bit 2 is a logic 1 when a 5-bit word length is selected via bits 0

TABLE II. Summary of Registers

	Register Address									
	0 DLAB=0	0 DLAB=0	1 DLAB=0	2	3	4	5	6	0 DLAB=1	1 DLAB=1
Bit No.	**Receiver Buffer Register (Read Only)**	**Transmitter Holding Register (Write Only)**	**Interrupt Enable Register**	**Interrupt Ident. Register (Read Only)**	**Line Control Register**	**MODEM Control Register**	**Line Status Register**	**MODEM Status Register**	**Divisor Latch (LS)**	**Division Latch (MS)**
	RBR	**THR**	**IER**	**IIR**	**LCR**	**MCR**	**LSR**	**MSR**	**DLL**	**DLM**
0	Data Bit 0 (Note 1)	Data Bit 0	Received Data Available	"0" if Interrupt Pending	Word Length Select Bit 0 (WLS0)	Data Terminal Ready (DTR)	Data Ready (DR)	Delta 0 Clear to Send (DCTS)	Bit 0	Bit 8
1	Data Bit 1	Data Bit 1	Transmitter Holding Register Empty	Interrupt ID Bit (0)	Word Length Select Bit 1 (WLS1)	Request to Send (RTS)	Overrun Error (OE)	Delta Data Set Ready (DDSR)	Bit 1	Bit 9
2	Data Bit 2	Data Bit 2	Receiver Line Status	Interrupt ID Bit (1)	Number of Stop Bits (STB)	Out 1	Parity Error (PE)	Trailing Edge Ring Indicator (TERI)	Bit 2	Bit 10
3	Data Bit 3	Data Bit 3	MODEM Status	0	Parity Enable (PEN)	Out 2	Framing Error (FE)	Delta Data Carrier Detect (DDCD)	Bit 3	Bit 11
4	Data Bit 4	Data Bit 4	0	0	Even Parity Select (EPS)	Loop	Break Interrupt (BI)	Clear to Send (CTS)	Bit 4	Bit 12
5	Data Bit 5	Data Bit 5	0	0	Stick Parity	0	Transmitter Holding Register (THRE)	Data Set Ready (DSR)	Bit 5	Bit 13
6	Data Bit 6	Data Bit 6	0	0	Set Break	0	Transmitter Shift Register Empty (TSRE)	Ring Indicator (RI)	Bit 6	Bit 14
7	Data Bit 7	Data Bit 7	0	0	Divisor Latch Access Bit (DLAB)	0	0	Data Carrier Detect (DCD)	Bit 7	Bit 15

Note 1: Bit 0 is the least significant bit. It is the first bit serially transmitted or received.

8.0 Registers (Continued)

and 1, one and a half Stop bits are generated. If bit 2 is a logic 1 when either a 6-, 7-, or 8-bit word length is selected, two Stop bits are generated. The Receiver checks the first Stop bit only, regardless of the number of Stop bits selected.

Bit 3: This bit is the Parity Enable bit. When bit 3 is a logic 1, a Parity bit is generated (transmit data) or checked (receive data) between the last data word bit and Stop bit of the serial data. (The Parity bit is used to produce an even or odd number of 1s when the data word bits and the Parity bit are summed.)

Bit 4: This bit is the Even Parity Select bit. When bit 3 is a logic 1 and bit 4 is a logic 0, an odd number of logic 1s is transmitted or checked in the data word bits and Parity bit. When bit 3 is a logic 1 and bit 4 is a logic 1, an even number of logic 1s is transmitted or checked.

Bit 5: This bit is the Stick Parity bit. When bits 3, 4 and 5 are logic 1 the Parity bit is transmitted and checked as a logic 0. If bits 3 and 5 are 1 and bit 4 is a logic 0 then the Parity bit is transmitted and checked as a logic 1. If bit 5 is a logic 0 Stick Parity is disabled.

Bit 6: This bit is the Break Control bit. It causes a break condition to be transmitted by the UART. When it is set to a logic 1, the serial output (SOUT) is forced to the Spacing (logic 0) state. The break is disabled by clearing bit 6 to a logic 0. The Break Control bit acts only on SOUT and has no effect on the transmitter logic.

Note: This feature enables the CPU to alert a terminal in a computer communications system. If the following sequence is used no erroneous or extraneous characters will be transmitted because of the break.

1. Load an all 0s, pad character, in response to THRE.

2. Set break after the next THRE.

3. Wait for the transmitter to be idle, (TSRE = 1), and clear break when normal transmission has to be restored.

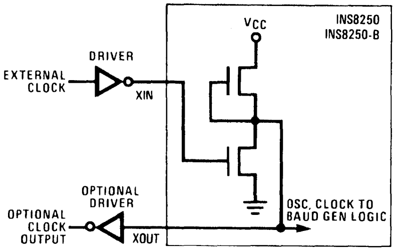
During the break, the Transmitter can be used as a character timer to accurately establish the break duration.

Bit 7: This bit is the Divisor Latch Access Bit (DLAB). It must be set high (logic 1) to access the Divisor Latches of the Baud Generator during a Read or Write operation. It must be set low (logic 0) to access the Receiver Buffer, the Transmitter Holding Register, or the Interrupt Enable Register.

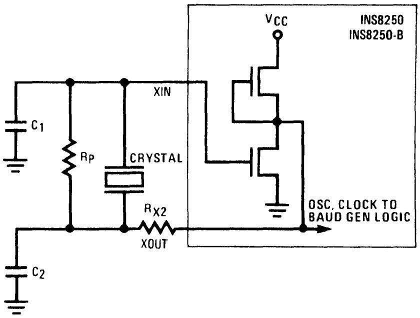
TABLE III. Baud Rates Using 1.8432 MHz Crystal

Desired Baud Rate	Decimal Divisor Used to Generate 16 x Clock	Percent Error Difference Between Desired and Actual
50	2304	—
75	1536	—
110	1047	0.026
134.5	857	0.058
150	768	—
300	384	—
600	192	—
1200	96	—
1800	64	—
2000	58	0.69
2400	48	—
3600	32	—
4800	24	—
7200	16	—
9600	12	—
19200	6	—
38400	3	—
56000	2	2.86

8.2 Typical Clock Circuits



TL/C/9329–12



TL/C/9329–13

Typical Oscillator Networks

Crystal	R_P	R_{X2}	C_1	C_2
1.8–3.1 MHz	1 MΩ	1.5k	10–30 pF	40–60 pF

TABLE IV. Baud Rates Using 3.072 MHz Crystal

Desired Baud Rate	Decimal Divisor Used to Generate 16 x Clock	Percent Error Difference Between Desired and Actual
50	3840	—
75	2560	—
110	1745	0.026
134.5	1428	0.034
150	1280	—
300	640	—
600	320	—
1200	160	—
1800	107	0.312
2000	96	—
2400	80	—
3600	53	0.628
4800	40	—
7200	27	1.23
9600	20	—
19200	10	—
38400	5	—

8.0 Registers (Continued)

8.3 PROGRAMMABLE BAUD GENERATOR

The UART contains a programmable Baud Generator that is capable of taking any clock input from DC to 3.1 MHz and dividing it by any divisor from 1 to $2^{16}-1$. The output frequency of the Baud Generator is 16 × the Baud [divisor # = (frequency input) ÷ (baud rate × 16)]. Two 8-bit latches store the divisor in a 16-bit binary format. These Divisor Latches must be loaded during initialization in order to ensure proper operation of the Baud Generator. Upon loading either of the Divisor Latches, a 16-bit Baud counter is immediately loaded.

Tables III and IV provide decimal divisors to use with crystal frequencies of 1.8432 MHz and 3.072 MHz, respectively, for common baud rates. For baud rates of 38400 and below, the error obtained is minimal. The accuracy of the desired baud rate is dependent on the crystal frequency chosen. Using a division of 0 is **not** recommended.

Note: The maximum operating frequency of the Baud Generator is 3.1 MHz. However, when using divisors of 3 and below, the maximum frequency is equal to the divisor in MHz. For example, if the divisor is 1, then the maximum frequency is 1 MHz. In no case should the data rate be greater than 56k Baud.

8.4 LINE STATUS REGISTER

This 8-bit register provides status information to the CPU concerning the data transfer. Table II shows the contents of the Line Status Register. Details on each bit follow:

Bit 0: This bit is the receiver Data Ready (DR) indicator. Bit 0 is set to a logic 1 whenever a complete incoming character has been received and transferred into the Receiver Buffer Register. Bit 0 is reset to a logic 0 by reading the data in the Receiver Buffer Register.

Bit 1: This bit is the Overrun Error (OE) indicator. Bit 1 indicates that data in the Receiver Buffer Register was not read by the CPU before the next character was transferred into the Receiver Buffer Register, thereby destroying the previous character. The OE indicator is set to a logic 1 upon detection of an overrun condition and reset whenever the CPU reads the contents of the Line Status Register.

Bit 2: This bit is the Parity Error (PE) indicator. Bit 2 indicates that the received data character does not have the correct even or odd parity, as selected by the even-parity-select bit. The PE bit is set to a logic 1 upon detection of a parity error and is reset to a logic 0 whenever the CPU reads the contents of the Line Status Register.

Bit 3: This bit is the Framing Error (FE) indicator. Bit 3 indicates that the received character did not have a valid Stop bit. Bit 3 is set to a logic 1 whenever the Stop bit following the last data bit or parity bit is a logic 0 (Spacing level). The FE indicator is reset whenever the CPU reads the contents of the Line Status Register. The UART will try to resynchronize after a framing error. To do this it assumes that the framing error was due to the next start bit, so it samples this "start" bit twice and then takes in the "data".

Bit 4: This bit is the Break Interrupt (BI) indicator. Bit 4 is set to a logic 1 whenever the received data input is held in the Spacing (logic 0) state for longer than a full word transmission time (that is, the total time of Start bit + data bits + Parity + Stop bits). The BI indicator is reset whenever the CPU reads the contents of the Line Status Register. Restarting after a break is received, requires the SIN pin to be logical 1 for at least ½ bit time.

Note: Bits 1 through 4 are the error conditions that produce a Receiver Line Status interrupt whenever any of the corresponding conditions are detected and the interrupt is enabled.

Bit 5: This bit is the Transmitter Holding Register Empty (THRE) indicator. Bit 5 indicates that the UART is ready to accept a new character for transmission. In addition, this bit causes the UART to issue an interrupt to the CPU when the Transmit Holding Register Empty Interrupt enable is set high. The THRE bit is set to a logic 1 when a character is transferred from the Transmitter Holding Register into the Transmitter Shift Register. The bit is reset to logic 0 whenever the CPU loads the Transmitter Holding Register.

Bit 6: This bit is the Transmitter Shift Register Empty (TSRE) indicator. Bit 6 is set to a logic 1 whenever the Transmitter Shift Register (TSR) is empty. It is reset to a logic 0 whenever a data character is transferred to the TSR.

Bit 7: This bit is permanently set to logic 0.

Note: The Line Status Register is intended for read operations only. Writing to this register is not recommended as this operation is only used for factory testing.

TABLE V. Interrupt Control Functions

Interrupt Identification Register				Interrupt Set and Reset Functions		
Bit 2	**Bit 1**	**Bit 0**	**Priority Level**	**Interrupt Type**	**Interrupt Source**	**Interrupt Reset Control**
0	0	1	—	None	None	—
1	1	0	Highest	Receiver Line Status	Overrun Error or Parity Error or Framing Error or Break Interrupt	Reading the Line Status Register
1	0	0	Second	Received Data Available	Receiver Data Available	Reading the Receiver Buffer Register
0	1	0	Third	Transmitter Holding Register Empty	Transmitter Holding Register Empty	Reading the IIR Register (if source of interrupt) or Writing into the Transmitter Holding Register
0	0	0	Fourth	MODEM Status	Clear to Send or Data Set Ready or Ring Indicator or Data Carrier Detect	Reading the MODEM Status Register

8.0 Registers (Continued)

8.5 INTERRUPT IDENTIFICATION REGISTER

In order to provide minimum software overhead during data character transfers, the UART prioritizes interrupts into four levels and records these in the Interrupt Identification Register. The four levels of interrupt conditions in order of priority are Receiver Line Status; Received Data Ready; Transmitter Holding Register Empty; and MODEM Status.

When the CPU accesses the IIR, the UART freezes all interrupts and indicates the highest priority pending interrupt to the CPU. While this CPU access is occurring, the UART records new interrupts, but does not change its current indication until the access is complete. Table II shows the contents of the IIR. Details on each bit follow:

Bit 0: This bit can be used in an interrupt environment to indicate whether an interrupt condition is pending. When bit 0 is a logic 0, an interrupt is pending and the IIR contents may be used as a pointer to the appropriate interrupt service routine. When bit 0 is a logic 1, no interrupt is pending.

Bits 1 and 2: These two bits of the IIR are used to identify the highest priority interrupt pending as indicated in Table V.

Bits 3 through 7: These five bits of the IIR are always logic 0.

8.6 INTERRUPT ENABLE REGISTER

This register enables the four types of UART interrupts. Each interrupt can individually activate the interrupt (INTR) output signal. It is possible to totally disable the interrupt system by resetting bits 0 through 3 of the Interrupt Enable Register (IER). Similarly, setting bits of this register to a logic 1, enables the selected interrupt(s). Disabling an interrupt prevents it from being indicated as active in the IIR and from activating the INTR output signal. All other system functions operate in their normal manner, including the setting of the Line Status and MODEM Status Registers. Table II shows the contents of the IER. Details on each bit follow.

Bit 0: This bit enables the Received Data Available Interrupt when set to logic 1.

Bit 1: This bit enables the Transmitter Holding Register Empty Interrupt when set to logic 1.

Bit 2: This bit enables the Receiver Line Status Interrupt when set to logic 1.

Bit 3: This bit enables the MODEM Status Interrupt when set to logic 1.

Bits 4 through 7: These four bits are always logic 0.

8.7 MODEM CONTROL REGISTER

This register controls the interface with the MODEM or data set (or a peripheral device emulating a MODEM). The contents of the MODEM Control Register (MCR) are indicated in Table II and are described below. Table II shows the contents of the MCR. Details on each bit follow.

Bit 0: This bit controls the Data Terminal Ready ($\overline{\text{DTR}}$) output. When bit 0 is set to a logic 1, the $\overline{\text{DTR}}$ output is forced to a logic 0. When bit 0 is reset to a logic 0, the $\overline{\text{DTR}}$ output is forced to a logic 1.

Note: The $\overline{\text{DTR}}$ output of the UART may be applied to an EIA inverting line driver (such as the DS1488) to obtain the proper polarity input at the succeeding MODEM or data set.

Bit 1: This bit controls the Request to Send ($\overline{\text{RTS}}$) output. Bit 1 affects the $\overline{\text{RTS}}$ output in a manner identical to that described above for bit 0.

Bit 2: This bit controls the Output 1 ($\overline{\text{OUT 1}}$) signal, which is an auxiliary user-designated output. Bit 2 affects the $\overline{\text{OUT 1}}$ output in a manner identical to that described above for bit 0.

Bit 3: This bit controls the Output 2 ($\overline{\text{OUT 2}}$) signal, which is an auxiliary user-designated output. Bit 3 affects the $\overline{\text{OUT 2}}$ output in a manner identical to that described above for bit 0.

Bit 4: This bit provides a local loopback feature for diagnostic testing of the UART. When bit 4 is set to logic 1, the following occur: the transmitter Serial Output (SOUT) is set to the Marking (logic 1) state; the receiver Serial Input (SIN) is disconnected; the output of the Transmitter Shift Register is "looped back" into the Receiver Shift Register input; the four MODEM Control inputs ($\overline{\text{CTS}}$, $\overline{\text{DSR}}$, $\overline{\text{RI}}$, and $\overline{\text{DCD}}$) are disconnected; and the four MODEM Control outputs ($\overline{\text{DTR}}$, $\overline{\text{RTS}}$, $\overline{\text{OUT 1}}$, and $\overline{\text{OUT 2}}$) are internally connected to the four MODEM Control inputs. In the diagnostic mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit-and received-data paths of the UART.

In the diagnostic mode, the receiver and transmitter interrupts are fully operational. The MODEM Control Interrupts are also operational, but the interrupts' sources are now the lower four bits of the MODEM Control Register instead of the four MODEM Control inputs. The interrupts are still controlled by the Interrupt Enable Register.

Bits 5 through 7: These bits are permanently set to logic 0.

8.8 MODEM STATUS REGISTER

This register provides the current state of the control lines from the MODEM (or peripheral device) to the CPU. In addition to this current-state information, four bits of the MODEM Status Register provide change information. These bits are set to a logic 1 whenever a control input from the MODEM changes state. They are reset to logic 0 whenever the CPU reads the MODEM Status Register.

Table II shows the contents of the MSR. Details on each bit follow:

Bit 0: This bit is the Delta Clear to Send (DCTS) indicator. Bit 0 indicates that the $\overline{\text{CTS}}$ input to the chip has changed state since the last time it was read by the CPU.

Bit 1: This bit is the Delta Data Set Ready (DDSR) indicator. Bit 1 indicates that the $\overline{\text{DSR}}$ input to the chip has changed state since the last time it was read by the CPU.

Bit 2: This bit is the Trailing Edge of Ring Indicator (TERI) detector. Bit 2 indicates that the $\overline{\text{RI}}$ input to the chip has changed from a low to a high state.

Bit 3: This bit is the Delta Data Carrier Detect (DDCD) indicator. Bit 3 indicates that the $\overline{\text{DCD}}$ input to the chip has changed state.

Note: Whenever bit 0, 1, 2, or 3 is set to logic 1, a MODEM Status Interrupt is generated.

Bit 4: This bit is the complement of the Clear to Send ($\overline{\text{CTS}}$) input. If bit 4 (loop) of the MCR is set to a 1, this bit is equivalent to RTS in the MCR.

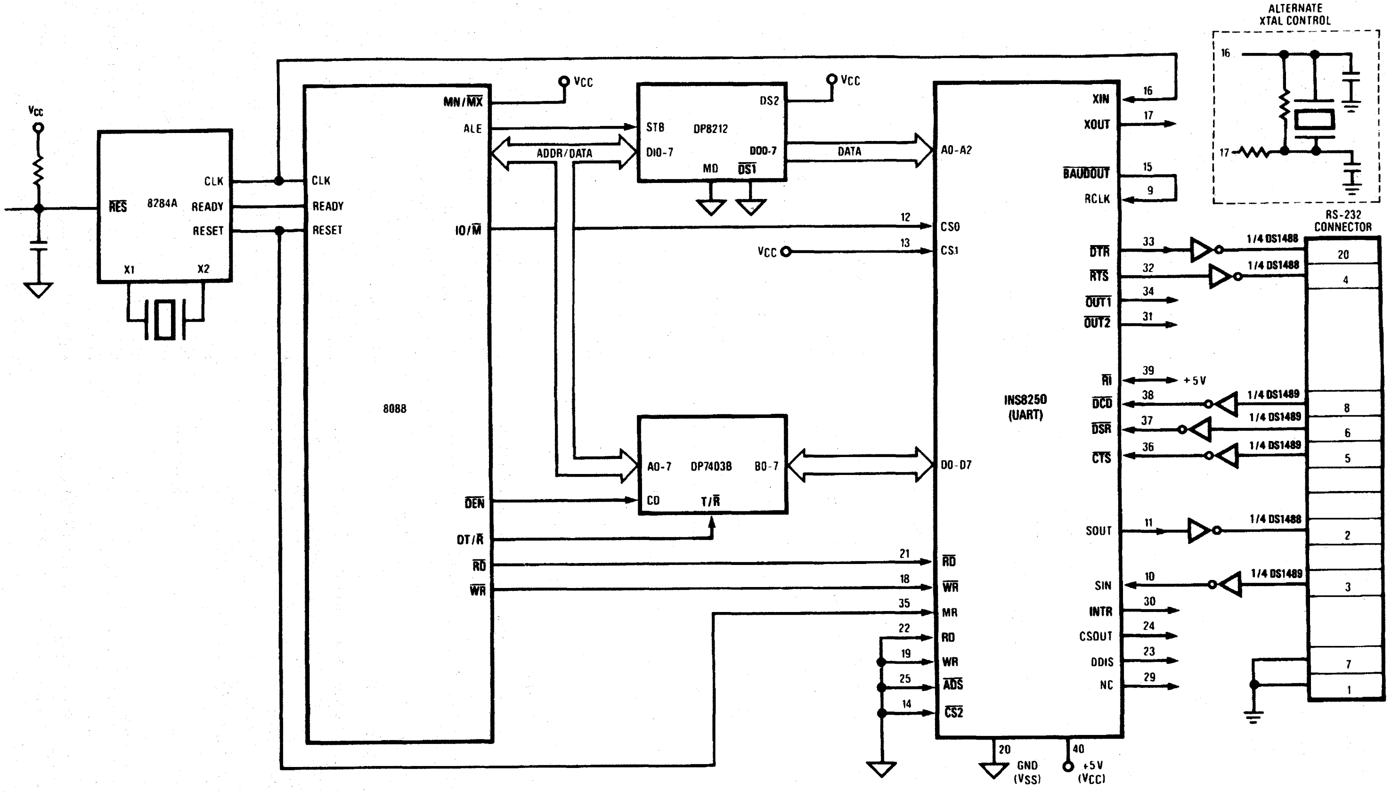
Bit 5: This bit is the complement of the Data Set Ready ($\overline{\text{DSR}}$) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to DTR in the MCR.

Bit 6: This bit is the complement of the Ring Indicator ($\overline{\text{RI}}$) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to OUT 1 in the MCR.

Bit 7: This bit is the complement of the Data Carrier Detect ($\overline{\text{DCD}}$) input. If bit 4 of the MCR is set to a 1, this bit is equivalent to OUT 2 in the MCR.

9.0 Typical Applications

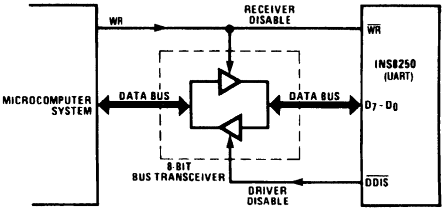
Basic Connections of an INS8250 to an 8088 CPU



TL/C/9329-15

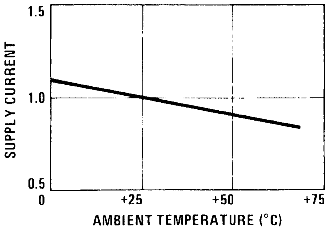
9.0 Typical Applications (Continued)

Typical Interface for a High-Capacity Data Bus



TL/C/9329-16

Typical Supply Current vs Temperature, Normalized



TL/C/9329-17

10.0 Ordering Information

```
INS8250XX
        ||
        |└──┤ /A+ = A+ RELIABILITY SCREENING
        |
        └───┤ N = PLASTIC PACKAGE
```

TL/C/9329-19

```
INS8250XX
        ||
        |└──┤ -B
        |
        └───┤ N = PLASTIC PACKAGE
            │ V = PLASTIC LEADED CHIP CARRIER PACKAGE
```

TL/C/9329-20

11.0 Reliability Information

Gate Count 2,000

Transistor Count 4,500

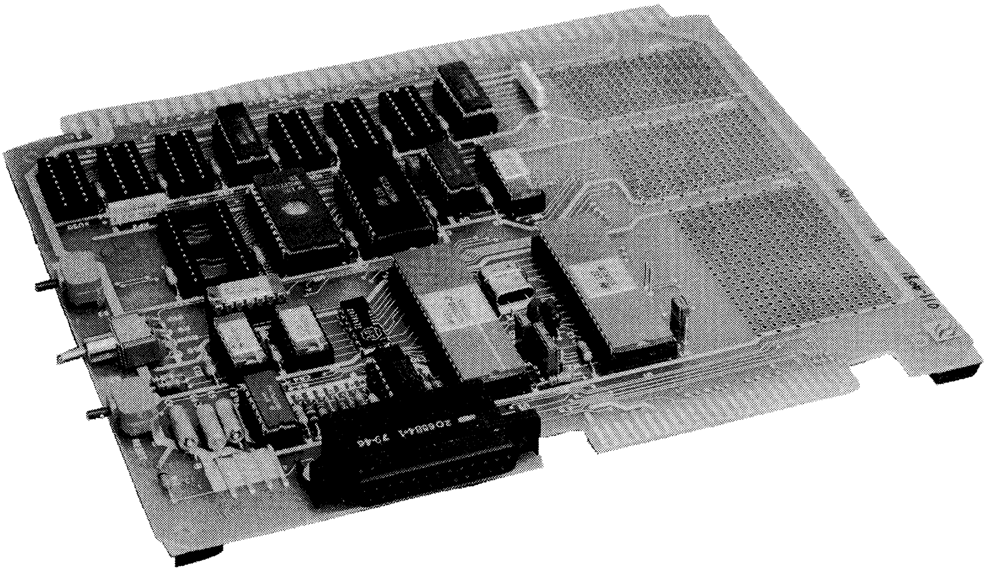
Section 3
Evaluation Board

Section 3 Contents

National Semiconductor Corp.

microCMOS

NSC888
NSC800™ Evaluation Board



TL/C/8533-1

- NSC800 8-Bit microCMOS CPU
- Executes Z80® Instruction Set
- 20 programmable parallel I/O lines
- Two 16-Bit programmable counters/timers
- Powerful 2k x 8 monitor program
- Five levels of vectored prioritized interrupts
- RS232 Interface
- 1k x 8 microCMOS RAM with sockets for up to 4k x 8 RAM
- Socket for additional 2k x 8, 2716 compatible memory component
- Wire wrap area
- Edge connectors for system expansion
- Single-step operation mode
- Fully assembled and tested

Product Overview

The NSC888 is a self-contained microprocessor board which enables the user to quickly evaluate the performance and features of the NSC800 product family. This fully assembled, tested board requires only the addition of a ±5V supply and an RS232 interface cable to the user's terminal to begin NSC800 evaluation.

A powerful system monitor is provided on the board which controls serial communications via the RS232 port. The monitor also includes command functions to load, execute and debug NSC800 programs.

The board includes an NSC800 CPU plus RAM, EPROM, I/O, Timers and interface components yet draws only 30 mA from the +5V supply and 3 mA from the −5V supply.

Although designed primarily as an assessment vehicle, the NSC888 can be readily programmed and adapted to a variety of uses. Wire wrap area is provided on-board for the user to build up additional circuitry or interfaces, thus tailoring this high-performance, low-power microprocessor board to meet individual needs.

Functional Description

Figure 1 and *Figure 2* provide information on the organization of the NSC888 board. Please refer to these figures for the following discussion.

Central Processor

The powerful NSC800 is the central processor for the NSC888. It provides bus control, clock generation and extensive interrupt capability. Featuring a multiplicity of programmable registers and sophisticated addressing modes, the NSC800 executes the Z80 instruction set.

Memory

- 128 bytes of RAM are provided by the NC810A RAM-I/O-Timer and are used by the monitor program for the system stack.
- 1024 bytes of RAM are provided by two 1k x 4 NMC6514's. Sockets are provided for six additional NMC6514's, for a total of 4k bytes of RAM.
- A 2k byte EPROM system monitor is provided on-board which includes facilities to load, execute and debug a users program.
- An additional EPROM socket is also on-board which accepts a 2k byte 2716 compatible memory component.

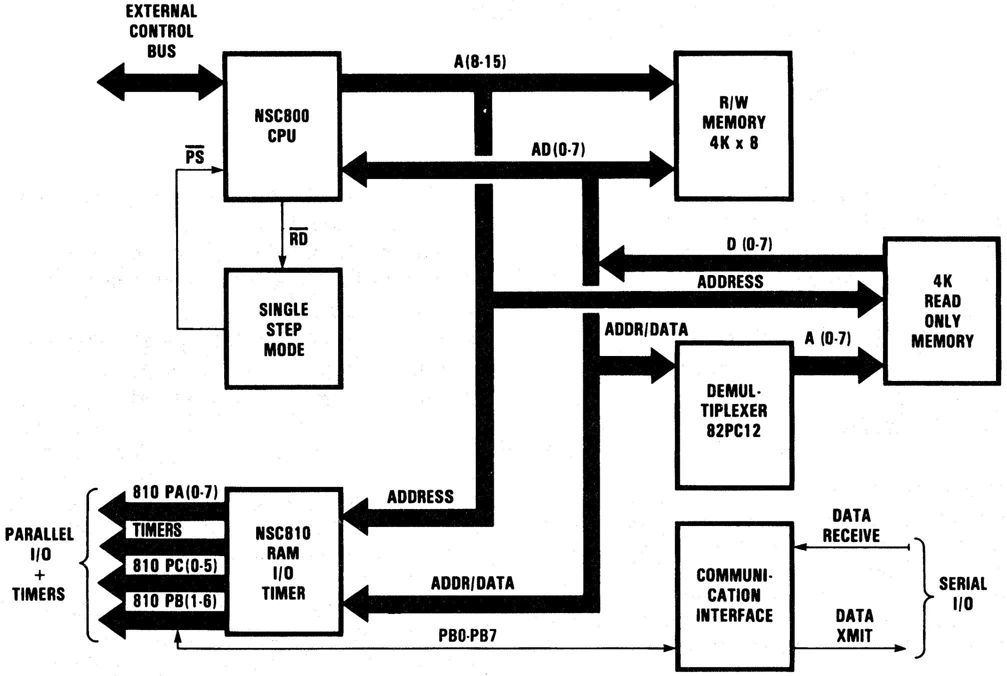
Input/Output

- **Parallel I/O**

  The NSC888 provides 20 programmable parallel I/O lines implemented using the I/O ports of the NSC810A RAM-I/O-Timer. The port bits may be individually defined as input or output, and can also be written to or read from in bytes. The I/O lines are conveniently brought to a 50 contact edge connector for user interface.
- **Serial I/O**

  An RS232 connector and accompanying support circuitry are provided on-board. Two I/O lines from the NSC810A RAM-I/O-Timer are used for the serial communications function, which is controlled exclusively by software. The baud rate is determined upon system initialization by the character bit rate from the users terminal. The maximum baud rate is 2400 baud.

Block Diagram



TL/C/8533-2

FIGURE 1

Functional Description (Continued)

Timers

The NSC888 provides two fully programmable binary 16-bit counters/timers utilizing the NSC810A RAM-I/O-Timer. These signals are also brought to the parallel I/O connector. Each timer may operate in any of six different modes:

- Event Counter
- Accumulative Timer
- Restartable Timer
- One Shot
- Square Wave
- Pulse Generator

Connectors

- **Parallel I/O**

  The parallel I/O lines and timer lines from the NSC810A RAM-I/O-Timer, plus interrupt lines from the CPU are brought to this 50 contact edge connector.

- **System Bus**

  All NSC800 CPU lines except XIN are brought to this 86 contact edge connector. In addition, the −5V line is also brought to the system bus connector.

- **RS232**

  This connector is provided for system interface to the users terminal.

Interrupts

The NSC888 utilizes the powerful interrupt processing capability of the NSC800 CPU. Interrupts are routed via a jumper matrix to the five interrupt inputs of the NSC800. Each input, which may be from the NSC810A I/O ports, NSC810A timers or off board via the system bus connector, generates a unique memory address (see Table I). All interrupts with the exception of NMI can be masked via software. Interrupt lines are also brought to the parallel I/O connector.

TABLE I.

Interrupt Input	Memory Address	Type	Priority
NMI	0066H	Non-maskable	Highest
RSTA	003CH	Maskable	
RSTB	0034H	Maskable	
RSTC	002CH	Maskable	
INTR	0038H*	Maskable	Lowest

*mode 1

NSC888 Firmware

The NSC888 system monitor is provided by a preprogrammed EPROM. This comprehensive monitor includes facilities to load, execute and debug programs. The monitor allows the user to examine and modify any RAM memory location or CPU register. It permits the insertion of break points to facilitate debugging. Programs can be executed starting at any location. The commands supported by the NSC888 system monitor are as follows:

- B - Select a new baud rate
- D - Display memory
- F - Fill memory between ranges
- G - Execute program with break points
- H - Hexadecimal math routine
- J - Non-destructive memory test
- K - Store 16-bit value in memory
- M - Move a block of data
- P - Put ASCII characters in memory
- Q - Query I/O ports
- S - Substitute and/or examine memory
- T - Type memory contents in ASCII
- V - Verify two blocks of data
- X - Examine or modify CPU registers
- Y - Memory search for string

These commands are fully explained in the NSC888 Hardware/Software Users Manual.

Single Step/Power Save

The NSC888 provides a unique single-step mode, utilizing the Power Save input of the NSC800 CPU. This input, when activated, reduces CPU power consumption from 50 mW to only 25 mW. It also allows the user to single-step through a program, checking and modifying code. This function is controlled via a switch on the board.

Specifications

Microprocessor

CPU—	NSC800
Data Word—	8 bits
Instruction Word—	8, 16, 24, 32 bits
Cycle Time—	2.00 μs (minimum instruction time)
System Clock—	2.00 MHz
Registers—	14 general purpose (8-bit) 2 index registers (16-bit) 1 stack pointer (16-bit) 1 program counter (16-bit)
Number of Instructions—	158
Address Capability—	64k bytes

Memory

RAM—	1152 bytes on-board plus sockets for an additional 3k bytes
ROM/EPROM—	Sockets for 4k bytes on-board
Access Time—	625 ns for opcode fetch 875 ns for memory read

Specifications (Continued)

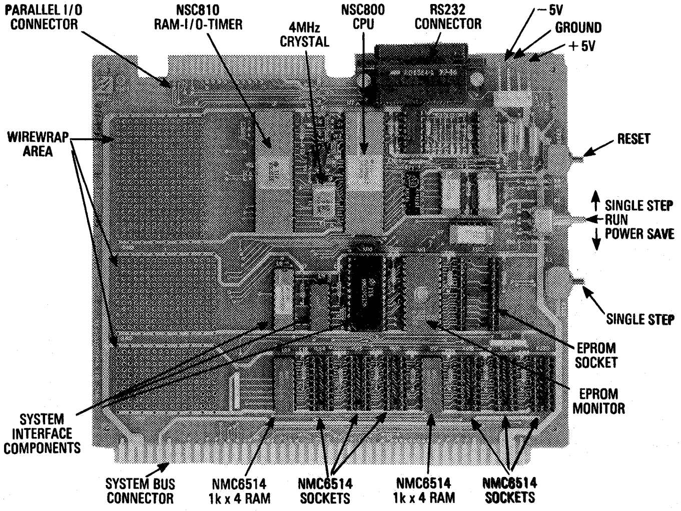
Connectors	
System Bus	86-pin double-sided card cage edge connector on 0.156 inch centers
Parallel I/O	50-pin double-sided edge connector on 0.1 inch centers Recommended mating connector: 3M 3415-0001 AMP 2-86792-3
Serial I/O	Standard RS232 connector
Power	+5V 30 mA (27C16 EPROM monitor) or 90 mA (2716 EPROM monitor) −5V 3 mA
Physical	
Height	6.75 (17.15 cm)
Width	7.85 (19.94 cm)

Order Information

NSC888	Includes CPU, 1152 bytes of RAM, sockets for additional 3k bytes of RAM, 2k byte monitor with additional socket for 2k byte ROM/EPROM, 20 I/O lines, RS232 interface, wire wrap area.

Documentation

The NSC888 Hardware/ Software Users Manual and NSC800 Microprocessor Family Handbook are shipped with the NSC888 Evaluation Board



TL/C/8533-3

FIGURE 2. NSC888 Evaluation Board

Section 4

Logic Devices

Section 4 Contents

National
Semiconductor
Corporation

microCMOS

MM82PC08 8-Bit Bidirectional Transceiver

General Description

The MM82PC08 is an 8-bit TRI-STATE® high-performance, low-power microCMOS transceiver. It provides bidirectional drive for bus-oriented microprocessor and digital communications systems. Straight through bidirectional transceivers are featured.

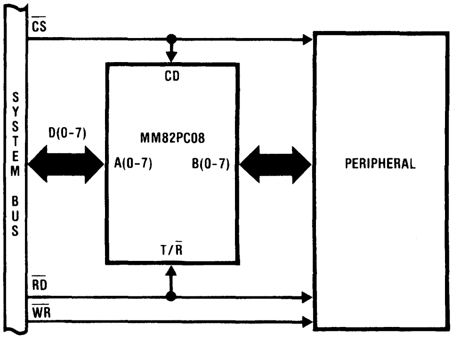
One input, Transmit/Receive, determines the direction of logic signals through the bidirectional transceiver; Transmit specifies data flow from Port A to Port B; Receive specifies data flow from Port B to Port A. The Chip Disable input disables both ports by placing them in the TRI-STATE mode.

The MM82PC08 may be utilized in completing NSC800™ high-performance, low-power designs. For military applications, the MM82PC08 is available with class B screening in accordance with Method 5004 of MIL-STD-883.

Features

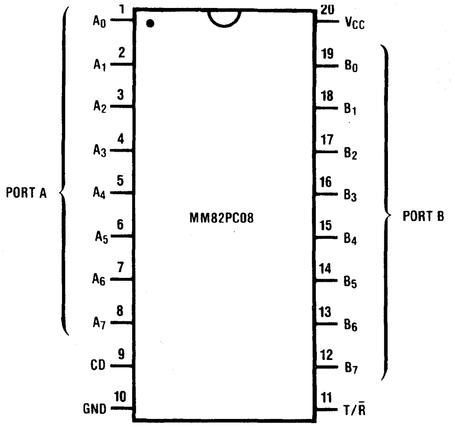
- microCMOS technology
- 8-bit bidirectional data flow reduces system package count
- Bidirectional TRI-STATE inputs/outputs interface with bus-oriented systems
- Full interface to CMOS logic levels
- Pinouts simplify system interconnections
- Transmit/receive and chip disable simplify control logic
- Compact 20-pin dual-in-line package
- Compact 28-pin leaded chip carrier
- Low power
- Both ports have 150 pF load drive capability
- TTL drive capability
  When $V_{CC} = 5V$

System Configuration and Connection Diagrams



TL/C/5595-1

Dual-In-Line Package



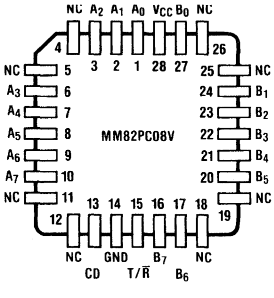
TL/C/5595-7

Top View

Order Number MM82PC08J or N
See NS Package J20A or N20A

Order Number MM82PC08V
See NS Package V28A

V Package



NC = No Connect

TL/C/5595-6

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, contact the National Semiconductor Sales Office/ Distributors for availability and specifications.

Storage Temperature Range	−65°C to +150°C
Voltage at Any Pin with Respect to Ground	−0.3V to V_{CC} +0.3V
Lead Temp. (Soldering, 10 seconds)	300°C
Power Dissipation	500 mW
Maximum V_{CC}	7V

Note: *Absolute maximum ratings are those values beyond which the safety of the device cannot be guaranteed. Continuous operation at these limits is not intended; operation should be limited to those conditions specified under DC Electrical Characteristics.*

Operating Conditions V_{CC} = 5V ±10%

Ambient Temperature

Military	−55°C to +125°C
Industrial	−40°C to +85°C
Commercial	0°C to +70°C

DC Electrical Characteristics

V_{CC} +5V ±10%, GND = 0V, unless otherwise specified

Symbol	Parameter	Test Conditions	Min	Typ	Max	Units
V_{IH}	Input High Voltage		0.7 V_{CC}		V_{CC}	V
V_{IL}	Input Low Voltage		0		0.2 V_{CC}	V
V_{OH}	Output High Voltage	V_{CC} = 4.5V, V_{IH} = 4.5V, I_{OH} = −2 mA	2.4			V
V_{OL}	Output Low Voltage	V_{CC} = 5.5V, V_{IL} = 0V V_{IH} = 5.5V, I_{OL} = 2 mA			0.4	V
I_{IH}	Input High Current	V_{CC} = 5.5V, V_{IN} = 5.5V			10	μA
I_{IL}	Input Low Current	V_{CC} = 5.5V, V_{IN} = 0V			−10	μA
I_{OH}	Output High Current	V_{CC} = 4.5V, V_{OUT} = 2.4V, V_{IH} = 4.5V	−2.0			mA
I_{OL}	Output Low Current	V_{CC} = 5.5V, V_{OUT} = 0.4V, V_{IL} = 0V	2.0			mA
I_{CC}	Power Supply Current	V_{CC} = 5.5V, V_{IH} = 5.5,V V_{IL} = 0V			400	μA
I_{OZL}	TRI-STATE Low Leakage Current	V_{CC} = 5.5V, V_{OUT} = 0V			−10	μA
V_{OZH}	TRI-STATE High Leakage Current	V_{CC} = 4.5V, V_{OUT} = 4.5V			+10	μA

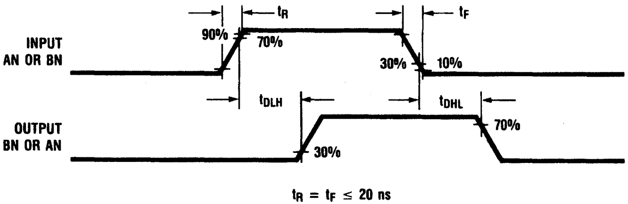
AC Electrical Characteristics V_{CC} = 5V ±10%, GND = 0V, C_L = 150 pF

Symbol	Parameter	Test Conditions	Min	Typ 100 pF	Max 100 pF	Units
t_{DLH}	Propagation Delay to Logical "1" from Port A, B to Port B, A	See *Figure 1*		50	70	ns
t_{DHL}	Propagation Delay to Logical "0" from Port A, B to Port B, A	See *Figure 1*		50	70	ns
t_{ZHTR}	Propagation Delay from High Impedance to Logical "1"from T/R to Port	See *Figure 2*		55	100	ns
t_{ZLTR}	Propagation Delay from High Impedance to Logical "0"from T/R to Port	See *Figure 2*		65	100	ns
t_{HZTR}	Propagation Delay from Logical "1" to High Impedance from T/R to Port	See *Figure 2*		50	100	ns

AC Electrical Characteristics V_{CC} = 5V ±10%, GND = 0V, C_L = 150 pF (Continued)

Symbol	Parameter	Test Conditions	Min	Typ 100 pF	Max 100 pF	Units
t_{LZTR}	Propagation Delay from Logical "0" to High Impedance from T/R to Port	See *Figure 2*		55	100	ns
t_{ZH}	Propagation Delay from High Impedance to Logical "1" from CD to Port	See *Figure 3*		50	100	ns
t_{ZL}	Propagation Delay from High Impedance to Logical "0" from CD to Port	See *Figure 3*		65	100	ns
t_{HZ}	Propagation Delay from Logical "1" to High Impedance from CD to Port	See *Figure 3*		50	100	ns
t_{LZ}	Propagation Delay from Logical "0" to High Impedance from CD to Port	See *Figure 3*		55	100	ns

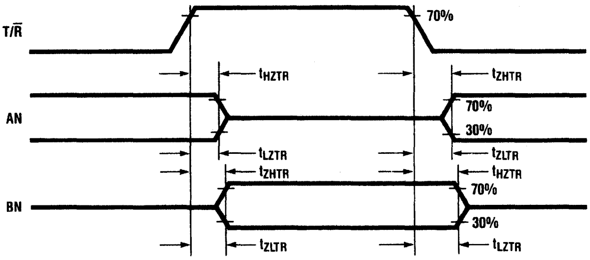
Timing Waveforms



$t_R = t_F \leq 20$ ns
10% TO 90%

TL/C/5595-2

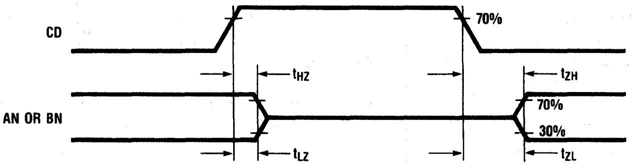
FIGURE 1. Propagation Delay from Input Port to Output Port



TL/C/5595-3

FIGURE 2. Propagation Delay from T/R to Ports

Timing Waveforms (Continued)



TL/C/5595-4

FIGURE 3. Propagation Delay from CD to Ports

Pin Descriptions

INPUT SIGNALS

Chip Disable (CD): When CD is high, Port A and Port B are disabled. A low on CD allows data to be transmitted in the direction specified by T/$\overline{\text{R}}$.

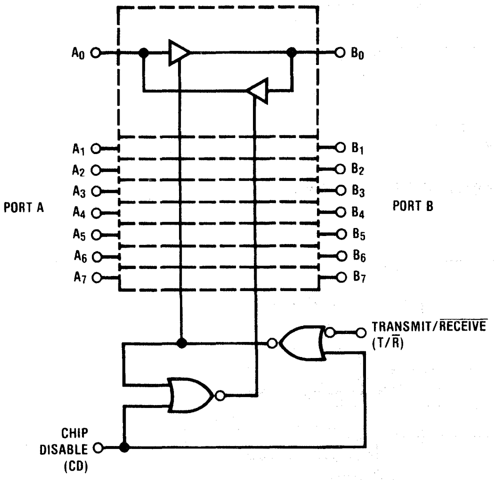
Transmit/Receive (T/$\overline{\text{R}}$): When T/$\overline{\text{R}}$ is high, Port A is designated as "IN" and Port B is designated as "OUT." When T/$\overline{\text{R}}$ is low, the flow is reversed so that the Port B is "IN" and Port A is "OUT".

INPUT/OUTPUT SIGNALS

Port A (A_0–A_7): Port A is an 8-bit bidirectional port with TRI-STATE outputs for bus-oriented microprocessor and digital communications systems.

Port B (B_0–B_7): Port B is identical to Port A including drive capability.

Logic Diagram


A0
B0
A1
A2
A3
A4
A5
A6
A7
B1
B2
B3
B4
B5
B6
B7
PORT A
PORT B
TRANSMIT/$\overline{\text{RECEIVE}}$
(T/$\overline{\text{R}}$)
CHIP
DISABLE
(CD)

TL/C/5595-5

FIGURE 4

Truth Table

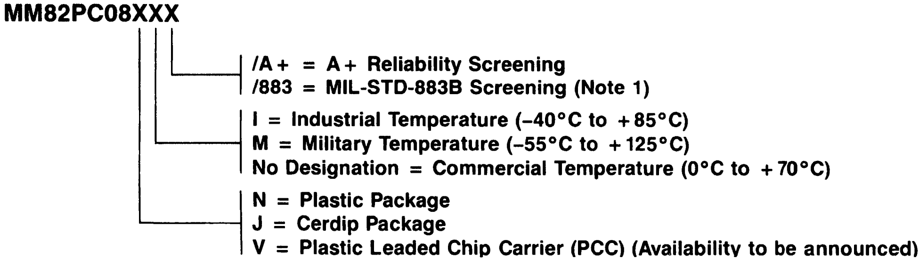
Inputs		Resulting Conditions	
Chip Disable	Transmit/$\overline{\text{Receive}}$	Port A	Port B
0	0	OUT	IN
0	1	IN	OUT
1	X	High Z	High Z

X = don't care

Reliability Information

Gate Count 70
Transistor Count 174

Ordering Information



MM82PC08XXX

/A+ = A+ Reliability Screening
/883 = MIL-STD-883B Screening (Note 1)

I = Industrial Temperature (-40°C to +85°C)
M = Military Temperature (-55°C to +125°C)
No Designation = Commercial Temperature (0°C to +70°C)

N = Plastic Package
J = Cerdip Package
V = Plastic Leaded Chip Carrier (PCC) (Availability to be announced)

TL/C/5595–8

Note 1: Do not specify a temperature option; all parts are screened to military temperature.

National Semiconductor Corporation

microCMOS

MM82PC12 8-Bit Input/Output Port

General Description

The MM82PC12 is a microCMOS 8-bit input/output port contained in a standard 24-pin dual-in-line package. The MM82PC12 can be used to implement latches, gated buffers, or multiplexers. Thus, all of the major peripheral and input/output functions of a microcomputer system can be implemented with this device.

The MM82PC12 includes an 8-bit latch with TRI-STATE® output buffers, and device selection and control logic. Also included is a service request flip-flop for the generation and control of interrupts to the microprocessor.

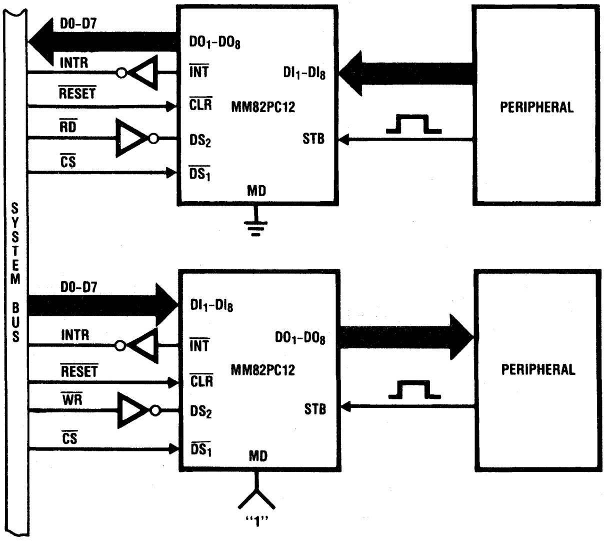
The MM82PC12 is pinout and function compatible with standard INS8212 and DP8212 devices.

For military applications, the MM82PC12 is available with class B screening in accordance with method 5004 of MIL-STD-883.

Features

- Drive capability—150 pF load
- High noise immunity
- Low power dissipation
- Full interface to CMOS logic levels
- microCMOS technology
- TTL drive capability when V_{CC} = 5V
- 8-bit data latch and buffer
- Service request flip-flop for generation and control of interrupts
- 1 μA input load current
- Reduces system package count by replacing buffers, latches, and multiplexers in microcomputer systems

System Configuration



TL/C/5596–1

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Storage Temperature Range	−65°C to +150°C
Voltage at Any Pin With Respect to Ground	−0.3V to V_{CC} + 0.3V
Lead Temperature (Soldering, 10 seconds)	300°C
Power Dissipation	500 mW
Maximum V_{CC}	7V

Note: *Absolute maximum ratings are those values beyond which the safety of the device cannot be guaranteed. Continuous operation at these limits is not intended; operation should be limited to those conditions specified under DC Electrical Characteristics.*

Operating Range V_{CC} = 5V ±10%

Ambient Temperature

Military	−55°C to +125°C
Industrial	−40°C to +85°C
Commercial	0°C to +70°C

DC Electrical Characteristics

V_{CC} = 5V ±10%, GND = 0V, unless otherwise specified

Symbol	Parameter	Test Conditions	Min	Typ	Max	Units
V_{IH}	Input High Voltage		0.7 V_{CC}		V_{CC}	V
V_{IL}	Input Low Voltage		0		0.2 V_{CC}	V
V_{OH}	Output High Voltage	V_{CC}= 4.5V, V_{IH} = 4.5V, I_{OH} = −2 mA	2.4			V
V_{OL}	Output Low Voltage	V_{CC}= 5.5V, V_{IL} = 0V, V_{IH} = 5.5V, I_{OL} = 2 mA			0.4	V
I_{IH}	Input High Current	V_{CC}= 5.5V, V_{IN} = 5.5V			10	μA
I_{IL}	Input Low Current	V_{CC}= 5.5V, V_{IN} = 0V			−10	μA
I_{OH}	Output High Current	V_{CC}= 4.5V, V_{OUT} = 2.4V, V_{IH} = 4.5V	−2.0			mA
I_{OL}	Output Low Current	V_{CC}= 5.5V, V_{OUT} = 0.4V, V_{IL} = 0V	2.0			mA
I_{CC}	Power Supply Current	V_{CC}= 5.5V, V_{IH} = 5.5V, V_{IL} = 0V			400	μA
I_{OZL}	TRI-STATE Low Leakage Current	V_{CC}= 5.5V, V_{OUT} = 0V			−10	μA
I_{OZH}	TRI-STATE High Leakage Current	V_{CC}= 4.5V, V_{OUT} = 4.5V			10	μA

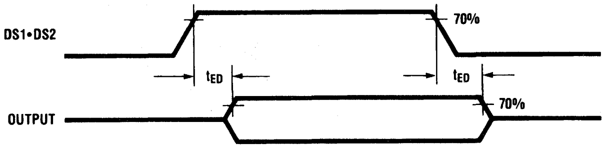
AC Electrical Characteristics

T_A = −55°C to +125°C, V_{CC} = 5V ±10%, GND = 0V, unless otherwise specified

Symbol	Parameter	Test Conditions	Min	Typ	Max	Units
t_{PW}	Pulse Width (STB, DS1 • DS2, CLR)			25	40	ns
t_{PD}	Data In to Data Out			45	60	ns
t_{WE}	Write Enable to Data Out			55	75	ns
t_{SET}	Data Setup Time		15			ns
t_H	Data Hold Time		20			ns
t_R	Reset to Data Out			50	65	ns
t_S	Select to Interrupt			50	65	ns
t_C	Clear to Data Out			45	60	ns
t_{ED}	Output Enable/Disable Time			50	65	ns

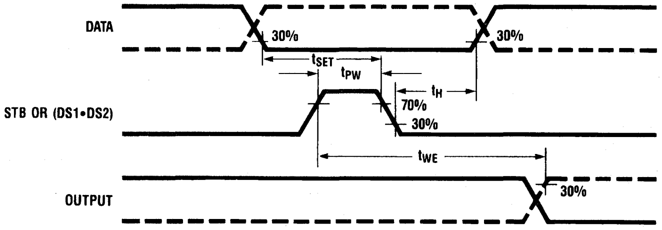
Timing Waveforms

Read Timing



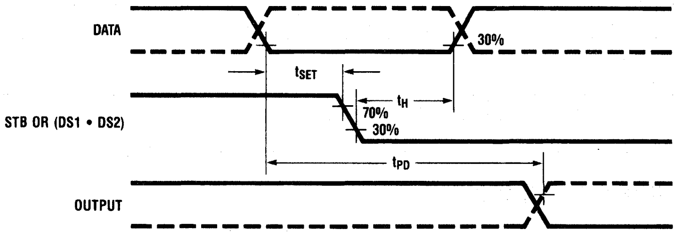
TL/C/5596-2

Write Timing



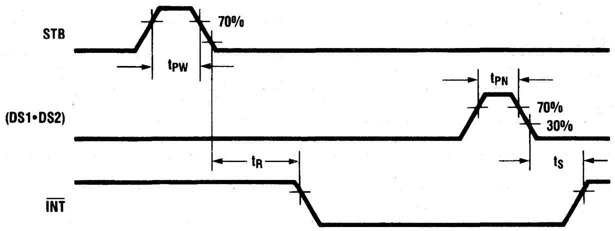
TL/C/5596-3

Data Setup, Hold Delay Timing



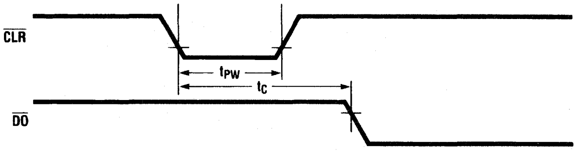
TL/C/5596-4

Interrupt Timing



TL/C/5596-5

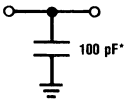
Clear Timing



TL/C/5596-6

Propagation Delays

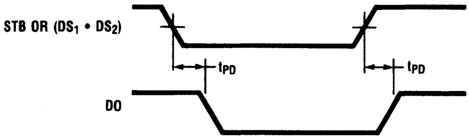
Figure 1 illustrates the calculations of a more useful propagation delay. The figure uses a 5V supply with a tolerance of ±10%, ambient temperature of +25°C, and a load capacitance of 100 pF. The AC Characteristics table depicts t_{PD}, at 5V, 25°C, equalling 25 ns. Use the graph in *Figure 1* to get the degradation multiple for 150 pF. The number shown is 1.09. The adjusted propagation delay is, therefore 25 × 1.09 or 27 ns.



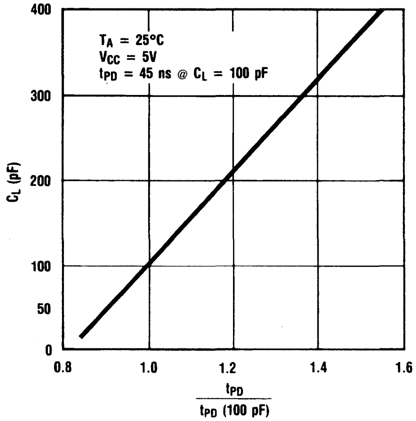
TL/C/5596-7

*Including jig and probe capacitance.

Output Test Circuit for Propagation Delays



TL/C/5596-8



TL/C/5596-9

FIGURE 1. Normalized Typical Propagation Delay vs. Load Capacitance

Pin Descriptions

The following describes the function of all the MM82PC12 input/output pins. Some of these descriptions reference internal circuits.

INPUT SIGNALS

Device Select ($\overline{DS_1}$, DS_2: When $\overline{DS_1}$ is low and DS_2 is high, the device is selected. The output buffers are enabled and the service request flip-flop is asynchronously reset (cleared) when the device is selected.

Mode (MD): When MD is high (output mode), the output buffers are enabled and the source of the data latch clock input is the device selection logic ($DS_1 \bullet DS_2$). When MD is low (input mode), the state of the output buffers is determined by the device selection logic ($DS_1 \bullet DS_2$) and the source of the data latch clock input is the strobe (STB) input.

Strobe (STB): STB is used as the data latch clock input when the mode (MD) input is low (input mode). STB is also used to synchronously set the service request flip-flop, which is negative edge triggered.

Data In (DI_1–DI_8): Data In is the 8-bit data input to the data latch, which consists of eight D-type flip-flops incorporating a level sensitive clock. While the data latch clock input is high, the Q output of each flip-flop follows the data input. When the clock input returns low, the data latch stores the data input. Clear ($\overline{CLR}$) is only effective when the clock is low (latch in the latched state).

Clear ($\overline{CLR}$): When $\overline{CLR}$ is low, the data latch is reset (cleared) if the clock is also low. The clock input high overrides the clear ($\overline{CLR}$) input data latch reset. $\overline{CLR}$ being low also resets the service request flip-flop. The service request flip-flop is in the non-interrupting state when reset.

OUTPUT SIGNALS

Interrupt ($\overline{INT}$): The interrupt pin goes low (interrupting state) when either the service request flip-flop is synchronously set by the strobe (STB) input or the device is selected.

Data Out (DO_1–DO_8): Data Out is the 8-bit data output of data buffers, which are TRI-STATE, non-inverting stages. These buffers have a common control line that either enables the buffers to transmit the data from the data latch outputs or disables the buffers by placing them in the high-impedance state.

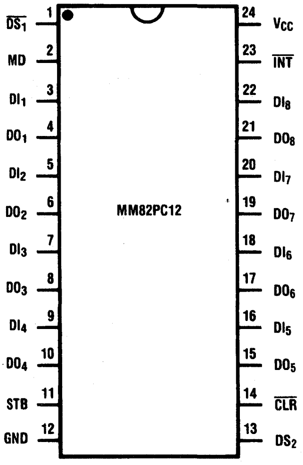
Reliability Information

Gate Count 108

Transistor Count 248

Connection Diagrams

Dual-In-Line Package

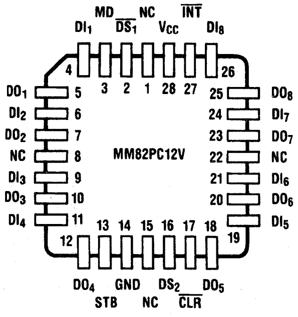


TL/C/5596–10

Top View

Order Number MM82PC12J or N
See NS Package Number J24A or N24A

Plastic Chip Carrier

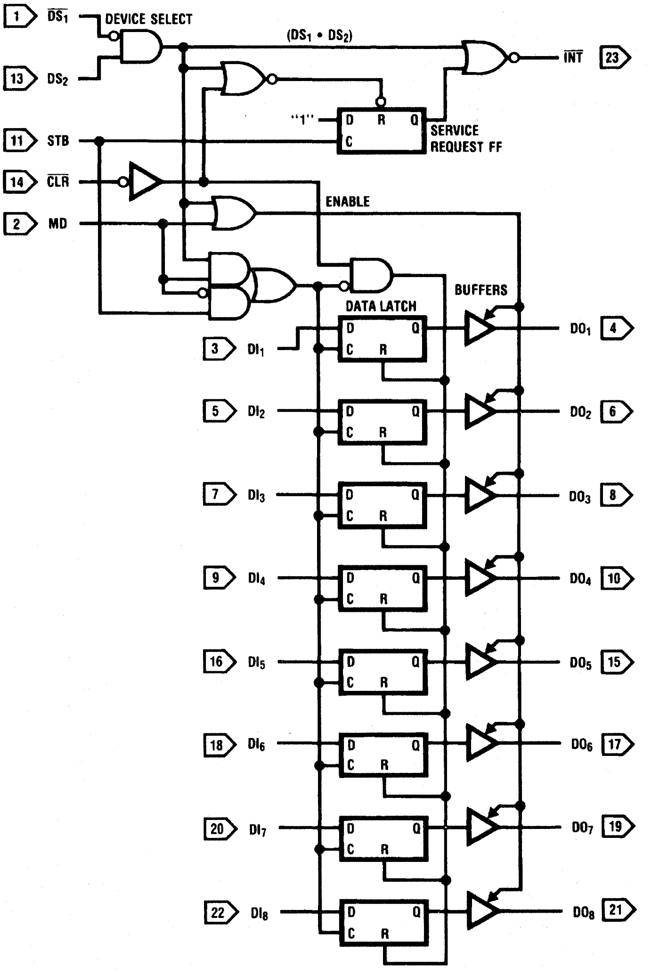


TL/C/5596–12

Top View

Order Number MM82PC12V
See NS Package Number V28A

Logic Diagram



TL/C/5596–11

Logic Table A

STB	MD	$DS_1 \bullet DS_2$	Data Out Equals
0	0	0	TRI-STATE
1	0	0	TRI-STATE
0	1	0	Data Latch
1	1	0	Data Latch
0	0	1	Data Latch
1	0	1	Data In
0	1	1	Data In
1	1	1	Data In

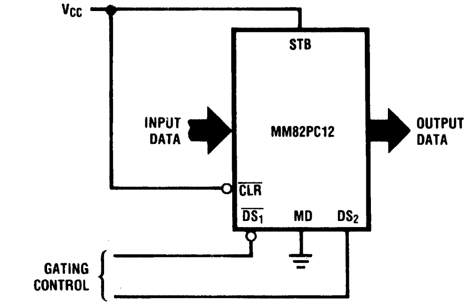
Note: $\overline{CLR}$ ↘ resets data latch to the output low state. The data latch clock is level sensitive, a low level clock latches the data.

Logic Table B

$\overline{CLR}$	$DS_1 \bullet DS_2$	STB	Q*	$\overline{INT}$
0 RESET	0	0	0	1
1	0	0	0	1
1	0	↘	1	0
1	1 RESET	0	0	0
1	0	0	0	1

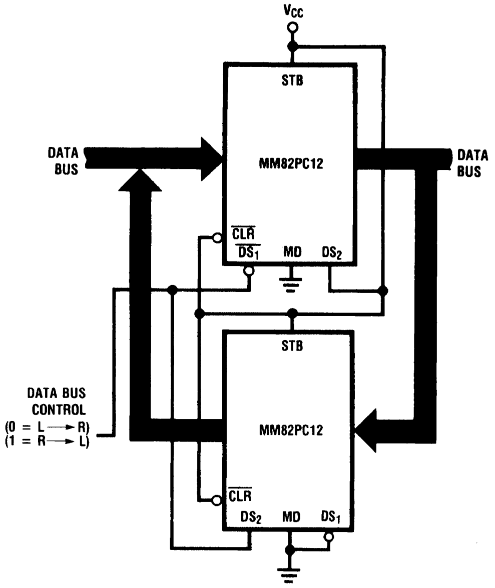
*Internal Service Request flip-flop.

Applications in Microcomputer Systems



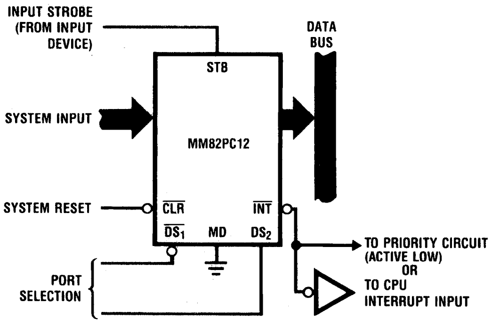
TL/C/5596-13

Gated Buffer (TRI-STATE)



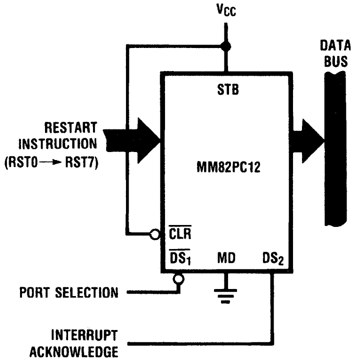
TL/C/5596-14

Bidirectional Bus Driver



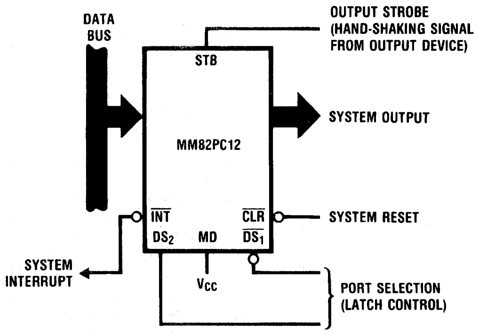
TL/C/5596-15

Interrupting Input Port



TL/C/5596-16

Interrupt Instruction Port


DATA BUS
OUTPUT STROBE (HAND-SHAKING SIGNAL FROM OUTPUT DEVICE)
STB
SYSTEM OUTPUT
MM82PC12
INT
CLR
SYSTEM RESET
DS2
MD
DS1
SYSTEM INTERRUPT
V_CC
PORT SELECTION (LATCH CONTROL)

TL/C/5596-17

Output Port (with Handshaking)

Ordering Information

MM82PC12XXX

- **/A+ = A+ Reliability Screening**
- **/883 = MIL-STD-883B Screening (Note 1)**

- **I = Industrial Temperature (−40°C to +85°C)**
- **M = Military Temperature (−55°C to +125°C)**
- **No Designation = Commercial Temperature (0°C to +70°C)**

- **N = Plastic Package**
- **J = Cerdip Package**
- **V = Plastic Leaded Chip Carrier (PCC) (Availability to be announced)**

TL/C/5596–18

Note 1: Do not specify a temperature option; all parts are screened to military temperature.

Section 5

Appendices/ Physical Dimensions

Section 5 Contents

The NS16550A: UART Design and Application Considerations

National Semiconductor Corp.
Application Note 491
Martin S. Michael
Daniel G. Durich

BACKGROUND

UARTs like other system components have evolved for many years to become faster, more integrated and less expensive. The rise in popularity of the personal computer with its focus and competition primarily centered on an architecture introduced by IBM®, has driven both UART performance and software compatibility issues. As transmission rates have increased, the amount of time the CPU has for other tasks while handling an active serial channel has been sharply reduced. One byte of data received at 1200 baud (8.3 ms) is received in 1⁄16th the time at 19.2 kbaud (520 μs). Software compatibility among the PC-based UARTs is critical due to the thousands of existing programs which use the serial channel and the new programs continually being offered.

Higher baud rates and compatibility requirements influence new UART designs. These two constraints result in UARTs that are capable of higher data rates, increasingly independent of CPU intervention and providing more autonomous features, while maintaining software compatibility. These development paths have been brought together in a new UART from National Semiconductor designated the NS16550A.

The NS16550A has all of the registers of its two predecessor parts (INS8250 and NS16450), so it can run all existing IBM PC, XT, AT, RT and compatible serial port software. In addition, it has a programmable mode which incorporates new high-performance features. Of course, all of these advanced features are useful in any asynchronous serial communications application regardless of the host architecture.

The reader is assumed to be familiar with the standard features of the NS16450, so this paper will concentrate mainly on the new features of the NS16550A. If the reader is unfamiliar with these UARTs it is advisable to start by reading their data sheets.

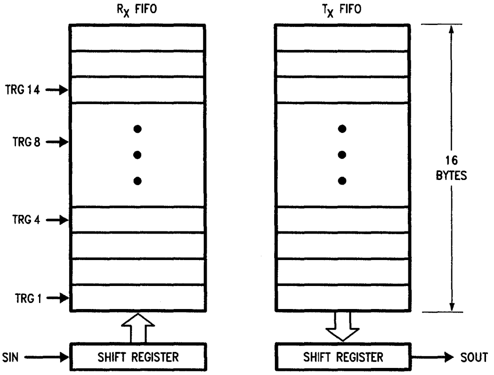
The first section reviews some of the design considerations and the operation of the NS16550A advanced features. The second section shows an NS16550A initialization routine written in 80286 assembly code with an explanation of the routine. The third section gives a detailed example of communications drivers written to interface two NS16550As on individual boards. These drivers are written for use with National Semiconductor's DB32032 evaluation boards, but can be ported to any NS32032-based system containing an NS32202 (ICU).

1.0 Design Considerations and Operation of the New UART Features

In order to optimize CPU/UART data transactions, the UART design takes into consideration the following constraints:

1. The CPU is usually much faster than the UART at transferring data. A high speed CPU could transfer a byte of data to/from the UART in a minimum of 280 ns. The UART would take over 1800 times longer to transmit/receive this data serially if it were operating at 19.2 kbaud.
2. There is a finite amount of wasted CPU time due to software overhead when stopping its current task to service the UART (context switching overhead).
3. The CPU may be required to complete a certain portion of its current task in a multitasking system before servicing the UART. This delay is the CPU latency time associated with servicing the interrupt. The amount of time that the receiver can accept continuous data after it requests service from the CPU constrains CPU latency time.

The design constraints listed above are met by adding two FIFOs and specialized transmitter/receiver support circuitry to the existing NS16450 design. The FIFOs are 16 bytes deep—one holds data for the transmitter, the other for the receiver (see *Figure 1*). Similarity between the FIFOs stops with their size, as each has been customized for special transmitter or receiver functions. Each has support circuitry to minimize software overhead when handling interrupts. The NS16550A **receiver** optimizes the CPU/UART data transaction via the following features:



TL/C/9313-1

FIGURE 1. Rx and Tx FIFOs

1. The depth of the Receiver (Rx) FIFO ensures that as many as 16 characters will be ready to transfer when the CPU services the Rx interrupt. Therefore, the CPU transfer rate is effectively buffered from the serial data rate.
2. The program can select the number of bytes required in the Rx FIFO (1, 4, 8 or 14) before the UART issues an interrupt. This allows the software to modify the interrupt trigger levels depending on its current task or loading. It also ensures that the CPU doesn't continually waste time switching context for only a few characters.

3. The Rx FIFO will hold 16 bytes regardless of which trigger level the CPU selects. This makes allowances for a variety of CPU latency times, as the FIFO continues to fill after the interrupt is issued.

The NS16550A **transmitter** optimizes the CPU/UART data transaction via the following features:

1. The depth of the Transmitter (Tx) FIFO ensures that as many as 16 characters can be transferred when the CPU services the Tx interrupt. Once again, this effectively buffers the CPU transfer rate from the serial data rate.
2. The Transmitter (Tx) FIFO is similar in structure to FIFOs the user may have previously set up in RAM. The Tx depth allows the CPU to load 16 characters each time it switches context to the service routine. This reduces the impact of the CPU time lost in context switching.
3. Since a time lag in servicing an asynchronous transmitter usually has no penalty, CPU latency time is of no concern to transmitter operation.

TX AND RX FIFO OPERATION

The Tx portion of the UART transmits data through SOUT as soon as the CPU loads a byte into the Tx FIFO. The UART will prevent loads to the Tx FIFO if it **currently** holds 16 characters. Loading to the Tx FIFO will again be enabled as soon as the next character is transferred to the Tx shift register. These capabilities account for the largely autonomous operation of the Tx.

The UART starts the above operations typically with a Tx interrupt. The NS16550A issues a Tx interrupt whenever the Tx FIFO is empty and the Tx interrupt is enabled, except in the following instance. Assume that the Tx FIFO is empty and the CPU starts to load it. When the first byte enters the FIFO, the Tx FIFO empty interrupt will transition from active to inactive. Depending on the execution speed of the service routine software, the UART may be able to transfer this byte from the FIFO to the shift register before the CPU loads another byte. If this happens, the Tx FIFO will be empty again and typically the UART's interrupt line would transition to the active state. This could cause a system with an interrupt control unit to record a Tx FIFO empty condition, even though the CPU is currently servicing that interrupt. Therefore, after the first byte has been loaded into the FIFO the UART will wait one serial character transmission time before issuing a new Tx FIFO empty interrupt.

This one character Tx interrupt delay will remain active until at least two bytes have been loaded into the FIFO, concurrently. When the Tx FIFO empties after this condition, the Tx interrupt will be activated without a one character delay.

Rx support functions and operation are quite different from those described for the transmitter. The Rx FIFO receives data until the number of bytes in the FIFO equals the selected interrupt trigger level. At that time if Rx interrupts are enabled, the UART will issue an interrupt to the CPU. The Rx FIFO will continue to store bytes until it holds 16 of them. It will not accept any more data when it is full. Any more data entering the Rx shift register will set the Overrun Error flag. Normally, the FIFO depth and the programmable trigger levels will give the CPU ample time to empty the Rx FIFO before an overrun occurs.

One side-effect of having a Rx FIFO is that the selected interrupt trigger level may be above the data level in the FIFO. This could occur when data at the end of the block contains fewer bytes than the trigger level. No interrupt would be issued to the CPU and the data would remain in the UART. To prevent the software from having to check for this situation the NS16550A incorporates a timeout interrupt.

The timeout interrupt is activated when there is at least one byte in the Rx FIFO, and neither the CPU nor the Rx shift register has accessed the Rx FIFO within 4 character times of the last byte. The timeout interrupt is cleared or reset when the CPU reads the Rx FIFO or another character enters it.

These FIFO related features allow optimization of CPU/UART transactions and are especially useful given the higher baud rate capability (256 kbaud). However, in order to eliminate most CPU interactions, the UART provides DMA request signals. Two DMA modes are supported: single-transfer and multi-transfer. These modes allow the UART to interface to higher performance DMA units, which can interleave their transfers between CPU cycles or execute multiple byte transfers.

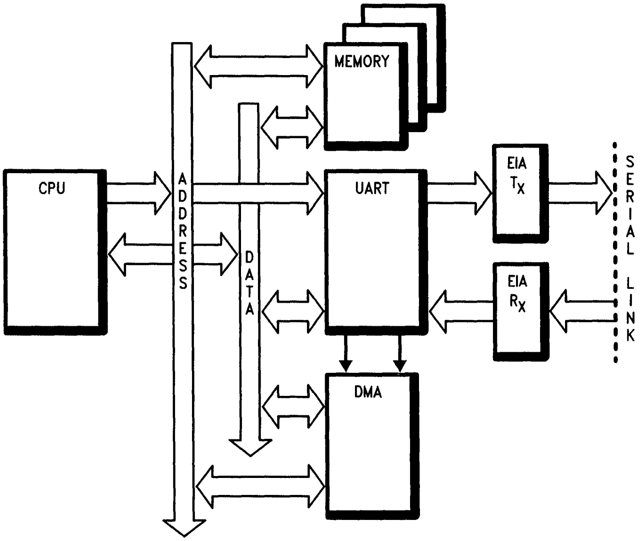
In single-transfer mode the receiver DMA request signal (Rx RDY) goes active whenever there is at least one character in the Rx FIFO. It goes inactive when the Rx FIFO is empty. The transmitter DMA request signal (Tx RDY) goes active when there are no characters in the Tx FIFO. It goes inactive when there is at least one character in the Tx FIFO. Therefore, in single-transfer mode active and inactive DMA signals are issued on a one byte basis.

In multi-transfer mode Rx RDY goes active whenever the trigger level or the timeout has been reached. It goes inactive when the Rx FIFO is empty. Tx RDY goes active when there is at least one unfilled position in the Tx FIFO. It goes inactive when the Tx FIFO is completely full. Therefore in multi-transfer mode active and inactive DMA signals are issued as the FIFO fills and empties. With 2 DMA channels (one for each Rx and Tx) assigned to it, the NS16550A could run somewhat independently of the CPU when the DMA unit transfers data composed of blocks with checksums.

SYSTEM OPERATION: THE NS16550A VS THE NS16450

Consider the typical system interface block diagram in *Figure 2.* This is a simple diagram, but it includes all of the components that typically interact with a UART. The advantages of the NS16550A over the NS16450 can be illustrated by comparing some of the system constraints when each UART is substituted into this basic system.

Both RS-232C and RS-422A interfaces can be used with either UART, however, the NS16550A can drive these interfaces up to 256 kbaud. Regarding the RS-422A specifica-



TL/C/9313-2

FIGURE 2. Typical System Interface

tion (max. 10 Mbaud) this is significantly faster than the NS16450 (max. 56 kbaud).

The NS16450 has no DMA request signals, so the DMA unit would not interact with the NS16450. The NS16550A, however, has DMA request signals and two modes of data transfer, as previously described, to interface with a variety of DMA units.

The greatest advantages of the NS16550A over the NS16450 are seen when considering the CPU/UART interface. Some characteristics of the transactions occurring between the CPU and the UART were previously cited. However, optimizing these transactions involves two issues:

1. Decreasing the amount of time the CPU interacts with the UART.
2. Increasing the amount of data transferred between the CPU and UART during their interaction time.

These optimization criteria are directly opposed to each other, but various features on the NS16550A have improved both.

One of the more obvious ways to decrease the CPU/UART interaction time is to decrease the time it takes for the transaction to occur. The NS16550A has an access cycle time that is almost 25% shorter than the NS16450. In addition, other timing parameters were made faster to simplify high speed CPU interactions.

The actual software required to transfer the data between the CPU and the UART is a small percentage of that required to support this transfer. However, each time a transfer occurs in the NS16450, this support software (overhead) must also be executed. With the NS16550A each time the UART needs service the CPU can theoretically transfer 16 bytes while only running through its overhead once. Tests have shown that this will increase the performance by a factor of 5 at the system level over the NS16450.

Another time savings for the CPU is a new feature of the UART interrupt structure. Unlike most other UARTs with Rx FIFOs, the NS16550A will issue an interrupt when there are characters below the interrupt trigger level after a preset time delay. This saves the extra time spent by the CPU to check for bytes that are at the end of a block, but won't reach the interrupt level.

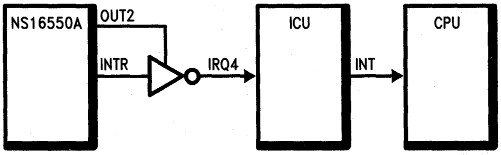
Since the NS16550A register set is identical to the NS16450 on power-up, all existing NS16450 software will run on it. The FIFOs are only enabled under program control.

All of this added performance is not without some tradeoffs. Two of the NS16450 pins, no connect (NC) and chip select out (CSOUT) have been replaced by the RxRDY and TxRDY pins. Most serial cards that currently use the NS16450 don't use these pins, so in those situations the NS16550A could be used as a plug-in upgrade. The software drivers for the NS16550A operating in FIFO mode need to be a little more sophisticated than for the NS16450. This will not cause a great penalty in CPU operating time as there is only one additional UART register to program and one to check during the initialization. One additional service routine is required to handle Rx timeout interrupts. This routine does not execute, except during intermittent transmissions or as described above.

All of these speed improvements and allowances for software constraints will make the NS16550A an optimal UART for both multi-tasking systems and multiport systems. Multitasking systems benefit from the increased time and flexibility offered to the CPU during context switching. Multiport systems, such as terminal concentrators, benefit from the on-board FIFOs and relatively autonomous functions of the UART.

SYSTEM INTERRUPT GENERATION

As a prelude to the topic of the next section (80286™-based system initialization) a review of a typical PC hardware interrupt path is given. This concerns only the interrupt path between the UART and the CPU (see *Figure 3*).



TL/C/9313–3

FIGURE 3. Typical PC Interrupt System Hardware

In order to enable interrupts from the UART to the CPU each hardware device must be correctly initialized. While initializing the hardware path, CPU interrupts are turned off to avoid false interrupts from this path. This initialization should be as short as possible to avoid other devices "stacking up" interrupts during this time.

After the NS16550A is initialized the bits 0–3 in the Interrupt Enable Register (IER) are set enabling all UART interrupts. Also, bit 3 in the Modem Control Register (MCR) is set to enable the buffer between the UART and the ICU.

The ICU has bit 4 of its Interrupt Mask Register (IMR) cleared, allowing interrupts occuring on IRQ4 to be transferred to the CPU via the group interrupt (INT). Finally, CPU interrupts are enabled again via the STI instruction.

The programmer should be aware that the ICU will be initialized for edge-triggered interrupts and that the UART always produces level active interrupts. This allows the system to get into a situation where the UART has multiple interrupts pending (signaled via a constantly high INTR), but the ICU fails to respond because it expects an edge for each pending interrupt. To avoid this situation, the programmer should disable all UART interrupts via the IER when entering each UART interrupt service routine and then reenable all UART interrupts that are to be used just before exiting each interrupt service routine.

SUMMARY

Up to this point the features of the NS16550A have been described, some of the design goals that resulted in these features have been reviewed, and a comparison has been given between it and the NS16450. Increases in bus speed and specialized functions make this part both faster from the hardware point of view and more efficient from the software point of view.

2.0 NS16550A Initialization

This initialization can be used on any 80286-based system; it enables both FIFOs and all interrupts on the UART. Additional procedures would have to be written to actually transfer data and service interrupts. These procedures would be similar in form to the 32000-based example in the next section, but the code would be different. The general flow of the initialization is shown in *Figure 4* and described below.

DETAILED SOFTWARE DESCRIPTION

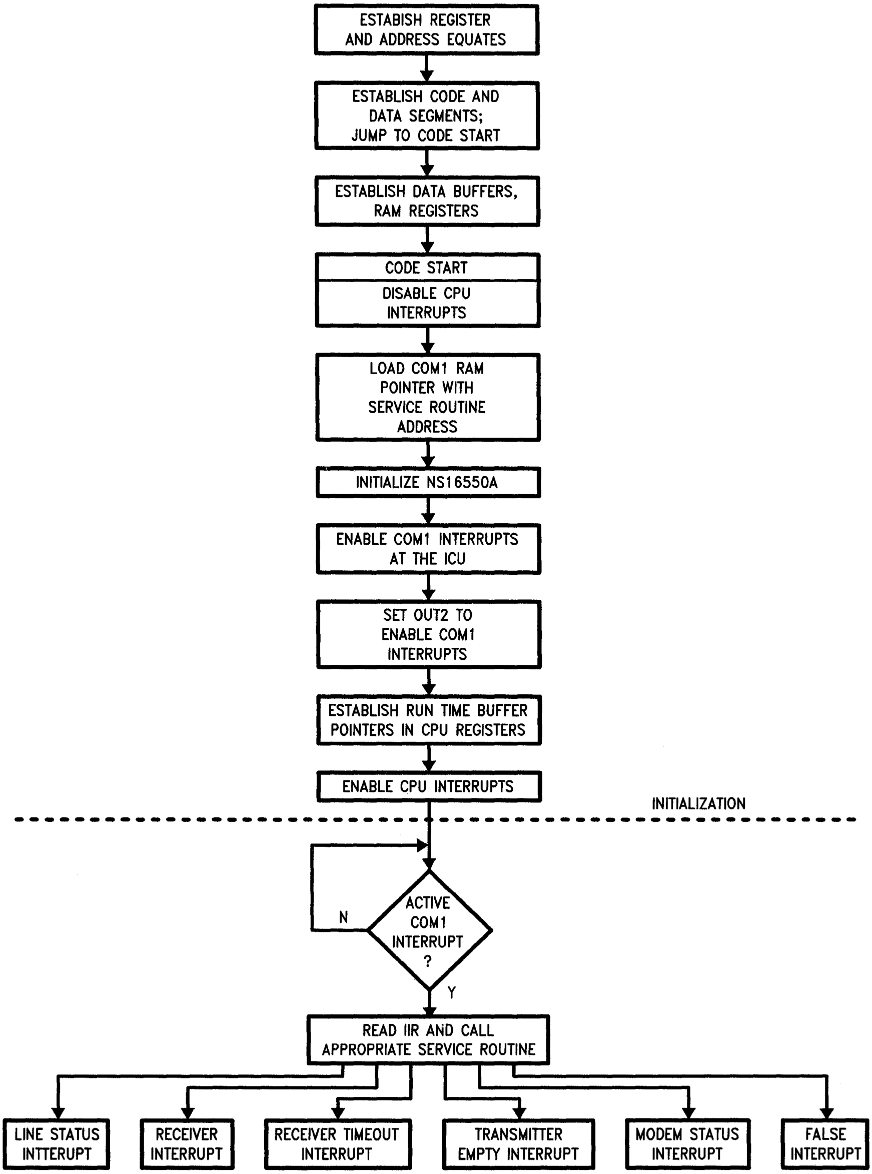
The first block in the initialization establishes abbreviations for the NS16550A registers and assigns addresses to them. The next three blocks establish code and data segments for the 80286. After jumping to the code start, the program disables CPU interrupts (CLI) until it has finished the initialization routine. Other interrupts may be active while CPU interrupts are masked, so the section of code following CLI should be as short as possible. The next block replaces the existing COM1 interrupt vector with the address of NS16550A interrupt handler (INTH in this case).

Initialization of the NS16550A is similar to the NS16450, except that there is one additional register to program which controls the FIFOs (Refer to the datasheet for a complete description). The sequence shown here sets bit 7 (DLAB) of the line control register (LCR), which enables access to the baud rate generator divisor. The divisor programmed is 0006 (19.2 kbaud) in this example. Programming the LCR again resets bit 7 (allowing access to the operational registers) and programs each frame for 7 data bits, one stop bit and even parity. The additional register that needs to be programmed in the NS16550A is the FIFO control register (FCR). The FCR data is 1100 0001. Bits 6 and 7 set the Rx FIFO interrupt trigger level at 14 characters. Bits 5 and 4 are reserved. Bit 3 keeps the DMA signal lines in mode 0. Setting bits 2 and 1 clear the Tx and Rx FIFOs, but this is done automatically when the FIFOs are first enabled by setting bit 0. Bit 0 of the FCR should ALWAYS BE SET whenever changes are to be made to the other bits of the FCR and the UART is to remain in FIFO Mode. When the FIFOs on the NS16550A are enabled bits 6 and 7 in the Interrupt Identification Register are set. Thus the program can distinguish between an NS16450 and an NS16550A, taking advantage of the FIFOs.

Sending a 0F to the Interrupt Enable Register enables all UART interrupts. The next two register accesses, reading the Line Status Register and the Modem Status Register, are optional. They are conservatively included in this initialization in order to defeat false interrupt indications in these registers caused by noise on the external lines.

The next block of code enables the interrupt signal to go beyond the UART through the system hardware. In many popular 80286-based personal computers, an interrupt control unit (ICU) has its mask register at I/O address 21H. To enable interrupts through this ICU for COM1 without disturbing other interrupts, the Interrupt Mask Register (IMR) is read. This data is combined with 1110 1111 via an AND instruction to unmask the COM1 interrupt and then loaded it back to the IMR. On these personal computers there is also a buffer on the interrupt line between the UART and ICU. This buffer is enabled by setting the OUT2 bit of the MODEM Control Register in the UART.

Before enabling CPU interrupts (STI) pointer registers to the data buffers of each service routine are loaded. After enabling CPU interrupts this program jumps to a holding loop to wait for an interrupt, whereas most programs would continue initializing other devices or jump to the system loop.


ESTABISH REGISTER AND ADDRESS EQUATES
ESTABLISH CODE AND DATA SEGMENTS; JUMP TO CODE START
ESTABLISH DATA BUFFERS, RAM REGISTERS
CODE START
DISABLE CPU INTERRUPTS
LOAD COM1 RAM POINTER WITH SERVICE ROUTINE ADDRESS
INITIALIZE NS16550A
ENABLE COM1 INTERRUPTS AT THE ICU
SET OUT2 TO ENABLE COM1 INTERRUPTS
ESTABLISH RUN TIME BUFFER POINTERS IN CPU REGISTERS
ENABLE CPU INTERRUPTS
INITIALIZATION
ACTIVE COM1 INTERRUPT ?
N
Y
READ IIR AND CALL APPROPRIATE SERVICE ROUTINE
LINE STATUS INTTERUPT
RECEIVER INTERRUPT
RECEIVER TIMEOUT INTERRUPT
TRANSMITTER EMPTY INTERRUPT
MODEM STATUS INTERRUPT
FALSE INTERRUPT

TL/C/9313-4

FIGURE 4. NS16550A Initialization and Driver Flowchart

```
        TITLE   550APP.ASM - NS16550A INITIALIZATION
;
;ESTABLISH NS16550A REGISTER ADDRESS/DATA EQUATES
;
;************ UART REGISTERS ************************
;
rxd     EQU 3F8H         ;RECEIVE DATA REG
txd     EQU 3F8H         ;TRANSMITT DATA REG
ier     EQU 3F9H         ;INTERRUPT ENABLE REG
dll     EQU 3F8H         ;DIVISOR LATCH LOW
dlh     EQU 3F9H         ;DIVISOR LATCH HIGH
iir     EQU 3FAH         ;INTERRUPT IDENTIFICATION REG
fcr     EQU 3FAH         ;FIFO CONTROL REG
lcr     EQU 3FBH         ;LINE CONTROL REG
mcr     EQU 3FCH         ;MODEM CONTROL REG
lsr     EQU 3FDH         ;LINE STATUS REG
msr     EQU 3FEH         ;MODEM STATUS REG
scr     EQU 3FFH         ;SCRATCH PAD REG
;
;**************** DATA EQUATES *****************
;
bufsize EQU 7CFH         ;TX AND RX BUFFER SIZE
dosrout EQU 25H          ;DOS ROUTINE SPECIFICATION
intnum  EQU 0CH          ;INTERRUPT NUMBER (0CH = COM1)
icumask EQU 0EFH         ;ICU INTERRUPT ENABLE MASK
divacc  EQU 80H          ;DIVISOR LATCH ACCESS CODE
lowdiv  EQU 06H          ;LOWER DIVISOR
uppdiv  EQU 00H          ;UPPER DIVISOR
dataspc EQU 1AH          ;DLAB = 0, 7 BITS, 1 STOP, EVEN
fifospc EQU 0C1H         ;FIFOS ENABLED, TRIG = 14, DMA MODE = 0
setout2 EQU 08H          ;SETTING OUT2 ENABLES INTRs TO THE ICU
intmask EQU 0FH          ;UART INTERRUPT ENABLE MASK
;
;*********** ESTABLISH CODE AND DATA SEGMENTS ******************
;
cseg    SEGMENT PARA PUBLIC "code"
        ORG     100H
        ASSUME  CS:cseg,DS:cseg
INIT:
        PUSH    CS
        POP     DS
        JMP     START
;
;********* ESTABLISH DATA BUFFERS AND RAM REGISTERS ********
;
msflag  DB      0
txflag  DB      0
sbuf    DB      bufsize DUP ("S")       ; STRING BUFFER
rbuf    DB      bufsize DUP ("R")       ; RECEIVE BUFFER
sbufe   EQU     sbuf + bufsize          ; END OF STRING BUFFER
rbufe   EQU     rbuf + bufsize          ; END OF RECEIVE BUFFER
;
START:
        CLI                       ;>>> DISABLE CPU INTERRUPTS <<<
```

```
;
;****** LOAD NEW INTERRUPT SERVICE ROUTINE POINTER FOR COM1 ***
;
        PUSH    DS                 ;SAVE EXISTING DATA SEG
        MOV     AH,dosrout         ;DESIGNATE FUNCTION NUMBER
        MOV     AL,intnum          ;DESIGNATE INTERRUPT
        PUSH    CS                 ;ALIGN CODE SEG
        POP     DS                 ;WITH DATA SEG
        MOV     DX,OFFSET INTH     ;SPECIFY SERVICE ROUTINE OFFSET
        INT     21H                ;REPLACE EXISTING INTR VECTOR
        POP     DS                 ;RESTORE CURRENT DATA SEG
;
;****************  INITIALIZE NS16550A  **********************
;
;This enables both FIFOs for data transfers at 19.2 kbaud using
;7 bit data, 1 stop bit and even parity. The Rx FIFO interrupt
;trigger level is set at 14 bytes.
        MOV     AL,divacc          ;SET-UP ACCESS TO DIVISOR LATCH
        MOV     DX,lcr
        OUT     DX,AL
        MOV     AL,lowdiv          ;LOWER DIVISOR LATCH, 19.2 kbaud
        MOV     DX,dll
        OUT     DX,AL
        MOV     AL,uppdiv          ;UPPER DIVISOR LATCH
        MOV     DX,dlh
        OUT     DX,AL
        MOV     AL,dataspc         ;DLAB = 0, 7 BITS, 1 STOP, EVEN
        MOV     DX,lcr
        OUT     DX,AL
        MOV     AL,fifospc         ;FIFOS ENABLED, TRIGGER = 14,
        MOV     DX,fcr             ;DMA MODE = 0
        OUT     DX,AL
        MOV     AL,intmask         ;ENABLE ALL UART INTERRUPTS
        MOV     DX,ier
        OUT     DX,AL
        MOV     DX,lsr             ;READ THE LSR TO CLEAR ANY FALSE
        IN      AL,DX              ;STATUS INTERRUPTS
        MOV     DX,msr             ;READ THE MSR TO CLEAR ANY FALSE
        IN      AL,DX              ;MODEM INTERRUPTS
;
;*************** ENABLE COM1 INTERRUPTS *********************
;
        IN      AL,21H             ;CHECK IMR
        AND     AL,icumask         ;ENABLE ALL EXISTING AND COM1
        OUT     21H,AL
        MOV     AL,setout2         ;SET OUT2 TO ENABLE INTR
        MOV     DX,mcr
        OUT     DX,AL
;
;*********  ESTABLISH RUN TIME BUFFER POINTERS IN REGISTERS ***
;
        MOV     SI,OFFSET sbuf
        MOV     DI,OFFSET rbuf
        MOV     BX,OFFSET sbuf
        MOV     BP,OFFSET rbuf
        STI     ;>>> ENABLE CPU INTERRUPTS <<<
```

3.0 Board to Board Communications with the NS16550A

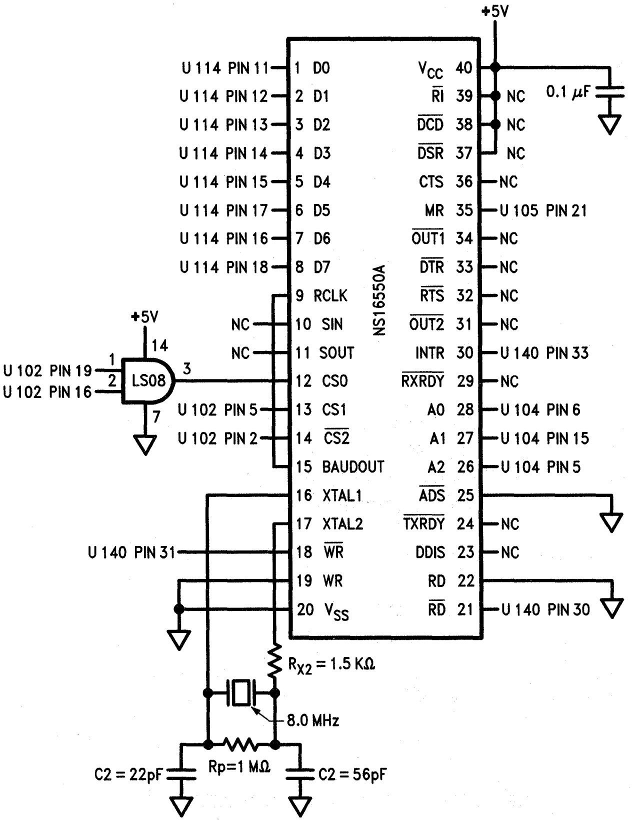
The following section describes the hardware and software for a fully asynchronous two board application. The two boards communicate simultaneously with each other via the NS16550As. Predetermined data is exchanged between the NS16550As and checked by the software for accuracy. Any data mismatches are flagged and stop the programs. Any data errors (i.e. overrun, parity, framing or break) will also stop the program. The NS16550A interface schematic, software flow chart and software are provided.

HARDWARE REQUIREMENTS

Running this application requires two NS32032-based boards. Each board must have one CPU, one ICU (NS32202), 256k of RAM (000000-03FFFF), the capability of running a monitor program (MON 32) and the capability of interfacing with a terminal. If MON 32 is not available, the display monitor service calls (SVC) must be altered to interface properly to the available terminal driver routines. In addition to these requirements, the NS16550A is enabled starting at address 0d00000.

The system described above was implemented on two DB32032 boards and used as an alpha site to test the NS16550A during its development. An NS16550A and appropriate decode logic were wirewrapped to each board (see *Figure 5*). As shown, an 8 MHz crystal is used to drive the baud rate generator, but for baud rates at or below 56 kbaud a 1.8432 MHz crystal can be substituted with changes to the divisor. Once this hardware is on both boards 5 connections between the NS16550As must be made—SIN to SOUT, SOUT to SIN, $\overline{\text{CTS}}$ to $\overline{\text{RTS}}$, $\overline{\text{RTS}}$ to $\overline{\text{CTS}}$, and GND to GND. Each DB32032 board has a port for attaching a terminal and a port available for downloading code. The applications software for these boards is downloaded from a VAX™ running the GNX™ debugger (V1.02). Once the downloads are complete to both boards the program D1APPS.EXE is started, then D2APPS.EXE is started.

If a VAX or the GNX debugger is not available the code can be loaded into PROMs and run directly.



TL/C/9313-5

FIGURE 5. NS16550A and DB32032 Board Interconnections

SOFTWARE OVERVIEW

The programs shown at the end of this application note are the assembly listings for D1APPS.ASM and D2APPS.ASM. These can be assembled, linked and loaded to form the executable (.EXE) files. The flowchart shown before them illustrates both programs.

Both programs are interrupt driven. D1APPS.EXE has its transmitter empty interrupt disabled until it receives its first 16 bytes from D2APPS.EXE. This allows the two programs to be started at different times. Data flow is controlled between the programs via $\overline{\text{RTS}}$ and $\overline{\text{CTS}}$ handshakes. D1APPS.EXE is started first and it loops until the first data from D2APPS.EXE arrives. As D1APPS.EXE exits its receiver interrupt routine, it enables its transmitter interrupt and begins to send bytes to D2APPS.EXE.

Transmission of a block of 16 bytes occurs when the Tx FIFO of the NS16550A is empty, the Tx interrupt is enabled and the receiver activates its clear to send ($\overline{\text{CTS}}$) signal. Each transmitter sends the next sequential block of data from a 256 byte buffer. When the bottom of the buffer is reached, the transmitter starts at the top of the buffer, again. The data transmitted from D1APPS.EXE to D2APPS.EXE is 00 to FF and from D2APPS.EXE to D1APPS.EXE is FF to 00. Since these are bench test programs for the NS16550A, the receiver subroutines compare the data they receive with the data they expect. This is done on a block-by-block basis and any mismatches result in both a message sent to the terminal and the program stopping.

DETAILED SOFTWARE DESCRIPTION

Initialization begins by equating NS16550A and ICU (NS32202) registers to the addresses in memory. The equates finish with a list of offsets associated with the static base register. These offsets give the starting locations for the RAM areas assigned to be data buffers. These include the UART interrupt entry offset (irl_mod); the string (sbuf), receive (rbuf), compare (cbuf) buffers and the interrupt table offset (intable).

At the code start (START::) the processor is put in the supervisor mode so that the interrupt dispatch table can be transferred from ROM to RAM. This transfer is essential in order to change the starting address of the UART interrupt service routine. To do this the interrupt service routine offset from the code start is calculated (isr-start). Combining this with the module table address (set-up by the linker, i.e., 9020) results in the interrupt table descriptor entry for UART interrupt service routine (isrent).

The next two sections of code load the data to be transmitted and compared into the RAM buffers sbuf and cbuf, respectively. The two programs differ at this point—D1APPS.EXE transmits 00 to FF and compares FF to 00 sequentially. D2APPS.EXE transmits FF to 00 and compares 00 to FF sequentially.

The NS16550A initialization starts with setting the divisor latch access bit, so the divisor can be loaded. It then determines the serial data format and disables all UART interrupts. The NS16550A initialization finishes by enabling and resetting the FIFOs and programming the receiver interrupt level for 14 bytes.

Next the ICU interrupt registers are set-up and interrupts are enabled. In program D1APPS.ASM the initialization finishes by enabling the receive data and line status interrupts. Since the transmitter FIFO empty interrupt is disabled D1APPS.EXE will stay in its hold loop until it receives data from D2APPS.EXE. D2APPS.EXE has its transmitter FIFO empty interrupt enabled at the end of its initialization, so it will send one block of 16 characters to D1APPS.EXE immediately.

When there are no interrupts pending and no service routines being executed, the programs run in a holding loop until the next interrupt.

Whenever the CPU enters the service routine (isr:) it checks the interrupts identification register (IIR) for the type of interrupt pending and branches to the appropriate subroutine. If the IIR value doesn't match a known interrupt condition, an invalid interrupt message is sent to the terminal and the program stops. Out of the five possible interrupts, two (line status and receiver timeout) have simple routines that only send a message to the terminal and then branch to the receiver data available routine. Modem status interrupts send a message to the CRT and then stop the program. Two robust interrupt service routines exist—one for the receiver and one for the transmitter.

The receiver interrupt service routine (rdai:) does the following:

1. Disables the $\overline{\text{RTS}}$ signal which stops the transmitter on the other board from sending more data.
2. Transfers all data from the UART Rx FIFO to the RAM receiver buffer (rbuf).
3. Branches to the compare subroutine when all data is transferred from the Rx FIFO.
4. Enables Tx interrupts in D1APPS.EXE.
5. Enables the $\overline{\text{RTS}}$ signal which allows the transmitter on the other board to send another block of data.

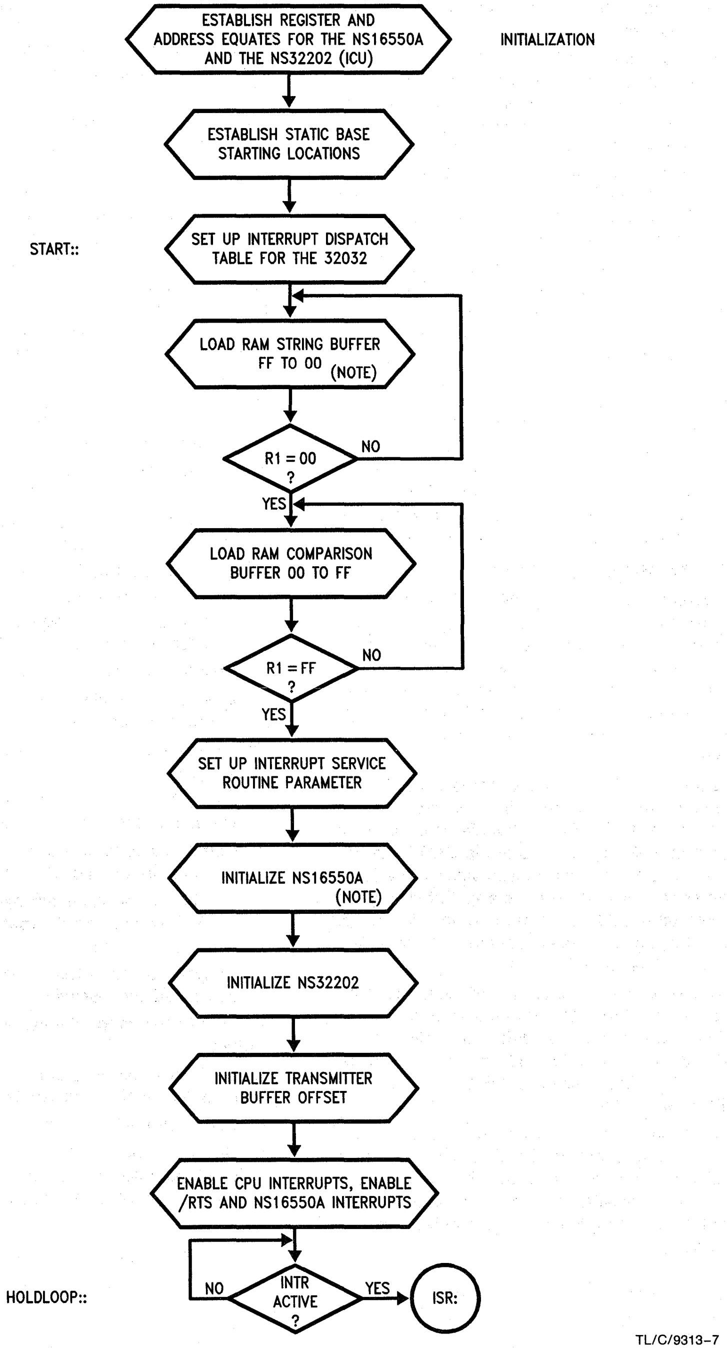
The compare interrupt service routine (compare:) does the following:

1. Aligns the receive buffer pointer to the last character taken in to the receive buffer (rbuf).
2. Compares each new byte in rbuf with the expected value (data stored in cbuf).
3. Sends a data mismatch message to the terminal and stops the program if the rbuf data fails to match the cbuf data.
4. Returns to rdai: when all of the new data in rbuf has compared successfully.

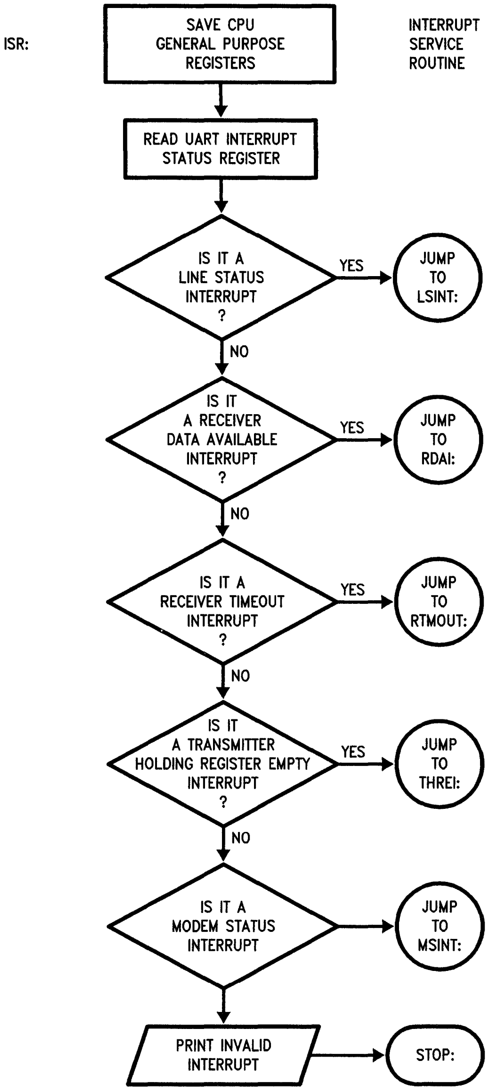
The transmitter interrupt service routine (threi:) does the following:

1. Decides whether to send 16 or 15 bytes in a block of data. **Note:** This decision is for testing purposes.
2. Sends one byte of data.
3. Checks for an active $\overline{\text{CTS}}$ condition. If it is active then it sends another byte of data. It continues to check and send a byte of data until all 15 or 16 bytes are sent.

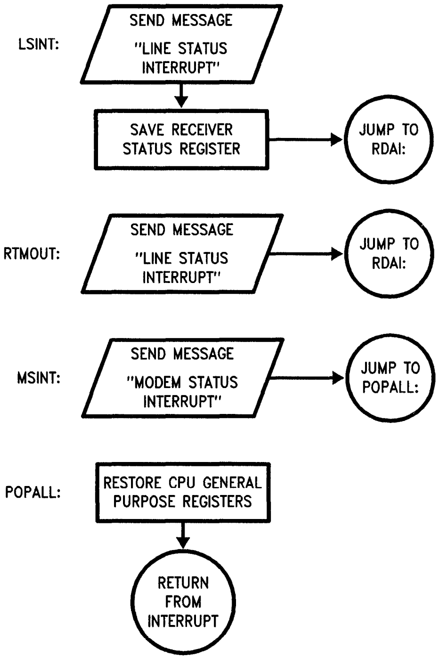
DIAPPS.ASM Flow Chart


ESTABLISH REGISTER AND ADDRESS EQUATES FOR THE NS16550A AND THE NS32202 (ICU)
INITIALIZATION
ESTABLISH STATIC BASE STARTING LOCATIONS
START::
SET UP INTERRUPT DISPATCH TABLE FOR THE 32032
LOAD RAM STRING BUFFER FF TO 00 (NOTE)
R1 = 00 ?
NO
YES
LOAD RAM COMPARISON BUFFER 00 TO FF
R1 = FF ?
NO
YES
SET UP INTERRUPT SERVICE ROUTINE PARAMETER
INITIALIZE NS16550A (NOTE)
INITIALIZE NS32202
INITIALIZE TRANSMITTER BUFFER OFFSET
ENABLE CPU INTERRUPTS, ENABLE /RTS AND NS16550A INTERRUPTS
HOLDLOOP::
NO
INTR ACTIVE ?
YES
ISR:
TL/C/9313-7

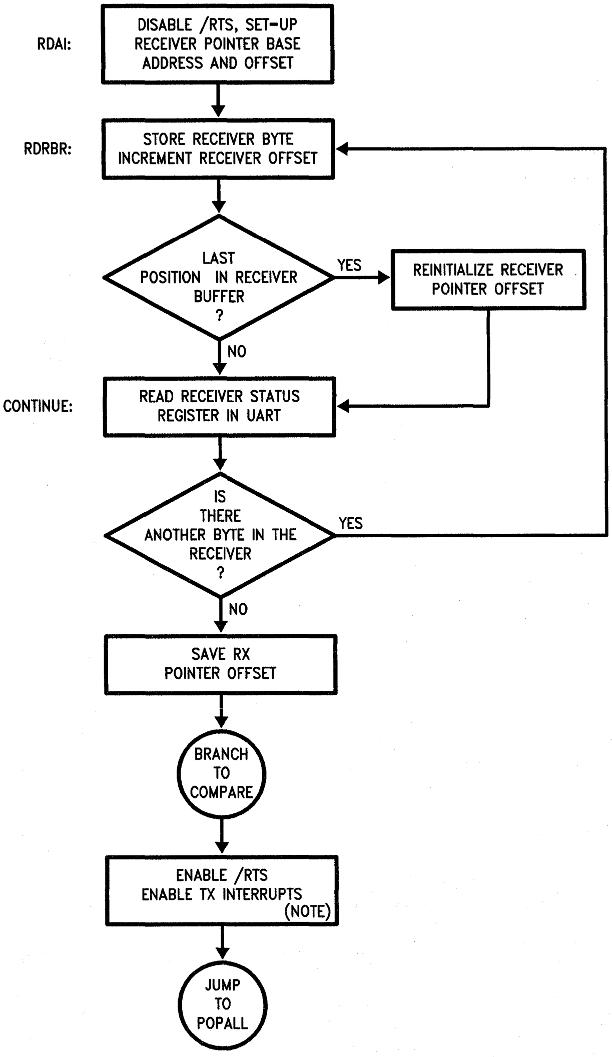
Note: This part of the software differs slightly in D2APPS.ASM


ISR:
SAVE CPU GENERAL PURPOSE REGISTERS
INTERRUPT SERVICE ROUTINE
READ UART INTERRUPT STATUS REGISTER
IS IT A LINE STATUS INTERRUPT ?
YES
JUMP TO LSINT:
NO
IS IT A RECEIVER DATA AVAILABLE INTERRUPT ?
YES
JUMP TO RDAI:
NO
IS IT A RECEIVER TIMEOUT INTERRUPT ?
YES
JUMP TO RTMOUT:
NO
IS IT A TRANSMITTER HOLDING REGISTER EMPTY INTERRUPT ?
YES
JUMP TO THREI:
NO
IS IT A MODEM STATUS INTERRUPT
JUMP TO MSINT:
PRINT INVALID INTERRUPT
STOP:

TL/C/9313-8

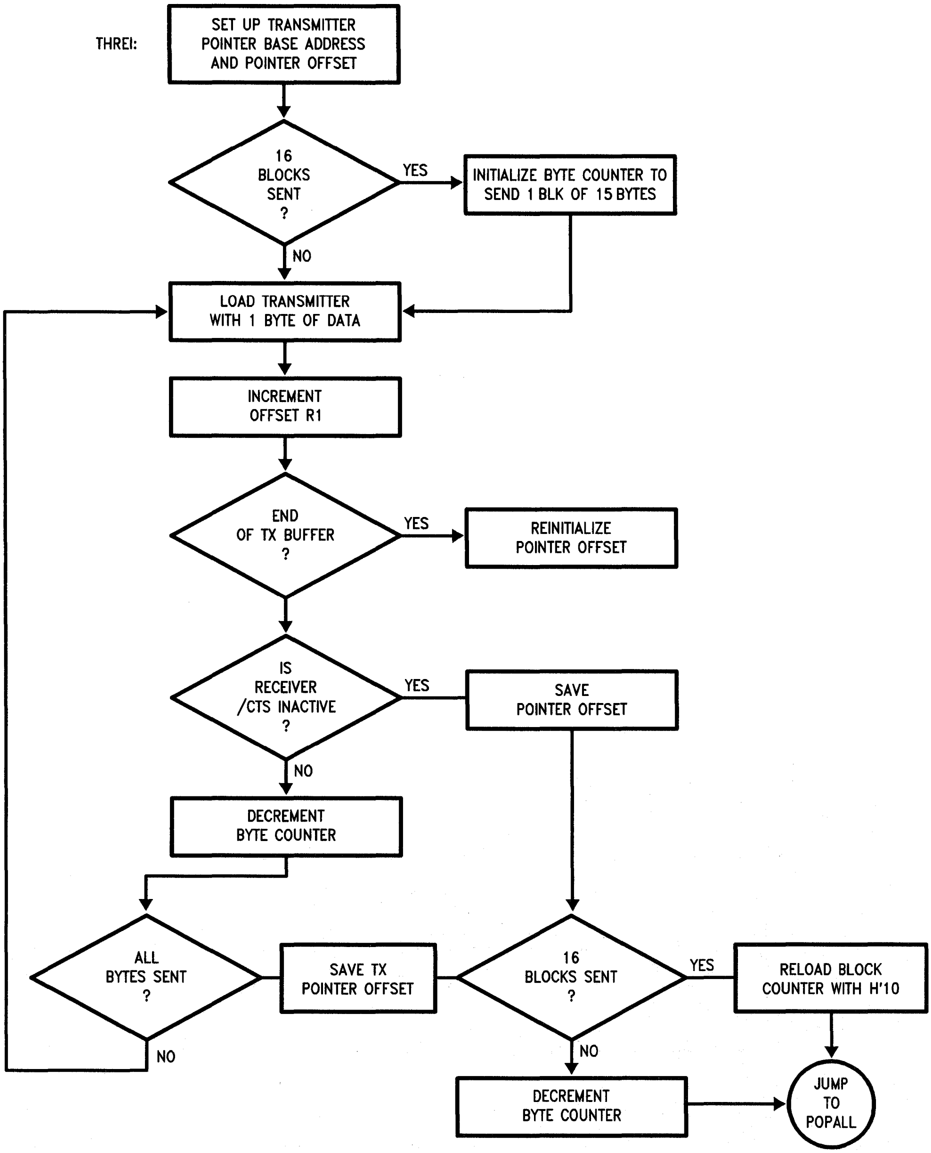

LSINT:
SEND MESSAGE "LINE STATUS INTERRUPT"
SAVE RECEIVER STATUS REGISTER
JUMP TO RDAI:
RTMOUT:
SEND MESSAGE "LINE STATUS INTERRUPT"
JUMP TO RDAI:
MSINT:
SEND MESSAGE "MODEM STATUS INTERRUPT"
JUMP TO POPALL:
POPALL:
RESTORE CPU GENERAL PURPOSE REGISTERS
RETURN FROM INTERRUPT

TL/C/9313-9

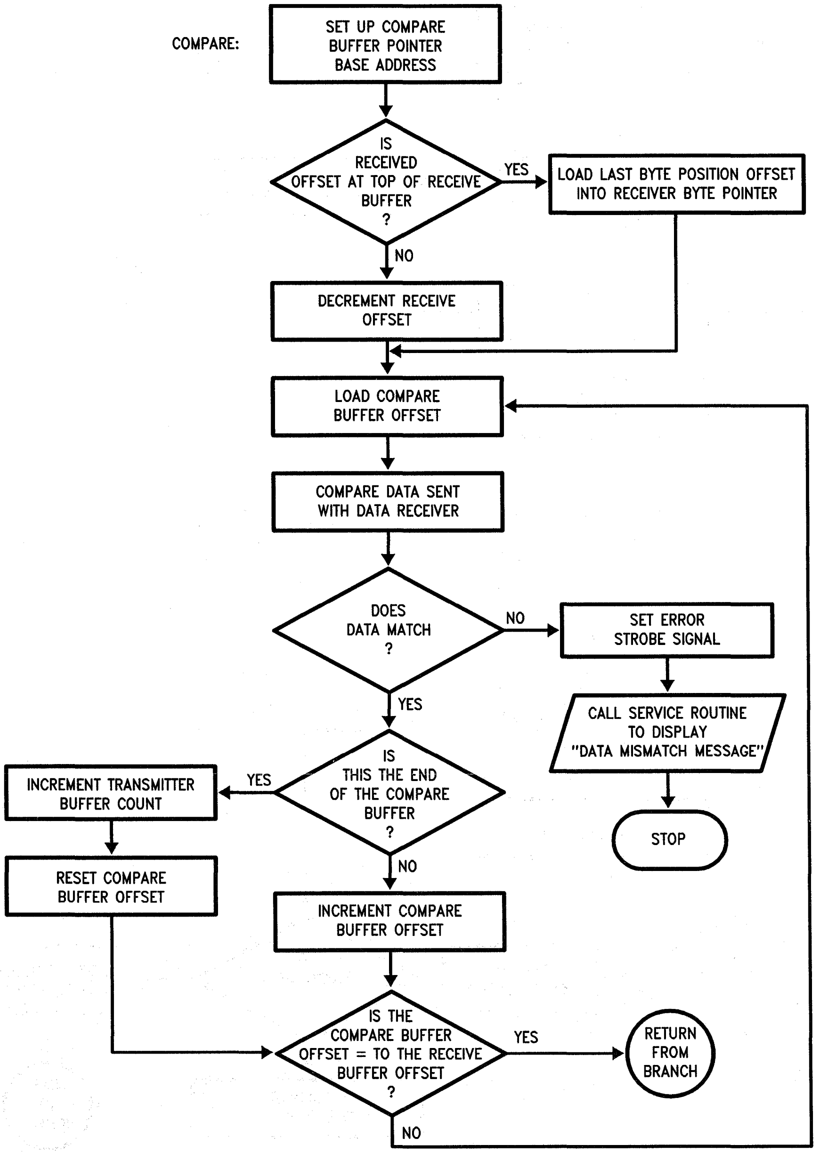

RDAI:
DISABLE /RTS, SET-UP RECEIVER POINTER BASE ADDRESS AND OFFSET
RDRBR:
STORE RECEIVER BYTE INCREMENT RECEIVER OFFSET
LAST POSITION IN RECEIVER BUFFER ?
YES
REINITIALIZE RECEIVER POINTER OFFSET
NO
CONTINUE:
READ RECEIVER STATUS REGISTER IN UART
IS THERE ANOTHER BYTE IN THE RECEIVER ?
YES
NO
SAVE RX POINTER OFFSET
BRANCH TO COMPARE
ENABLE /RTS ENABLE TX INTERRUPTS (NOTE)
JUMP TO POPALL

TL/C/9313–10

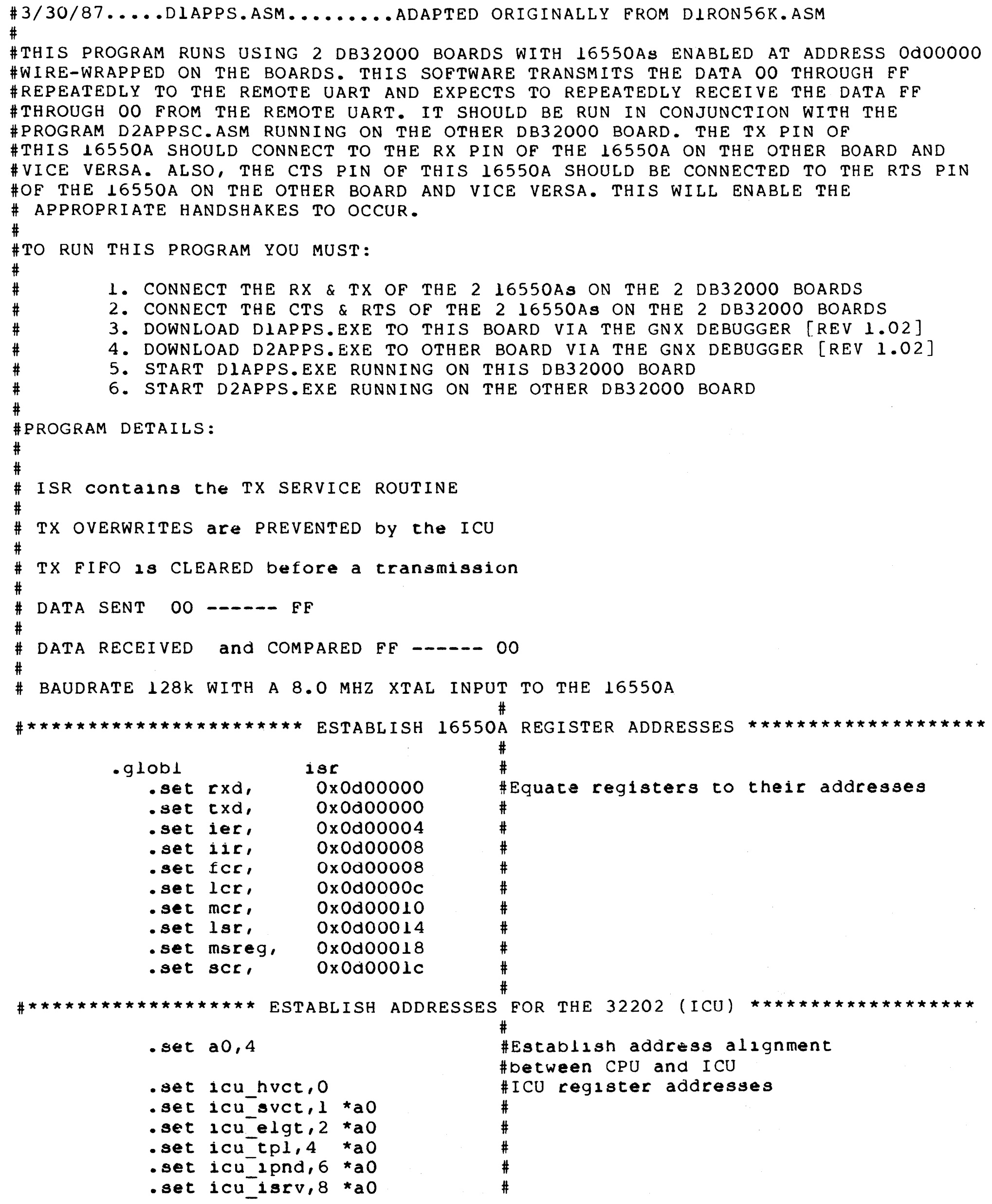
Note: This part of the software differs slightly in D2APPS.ASM


THREI:
SET UP TRANSMITTER POINTER BASE ADDRESS AND POINTER OFFSET
16 BLOCKS SENT ?
YES
INITIALIZE BYTE COUNTER TO SEND 1 BLK OF 15 BYTES
NO
LOAD TRANSMITTER WITH 1 BYTE OF DATA
INCREMENT OFFSET R1
END OF TX BUFFER ?
YES
REINITIALIZE POINTER OFFSET
IS RECEIVER /CTS INACTIVE ?
YES
SAVE POINTER OFFSET
NO
DECREMENT BYTE COUNTER
ALL BYTES SENT ?
NO
SAVE TX POINTER OFFSET
16 BLOCKS SENT ?
YES
RELOAD BLOCK COUNTER WITH H'10
NO
DECREMENT BYTE COUNTER
JUMP TO POPALL

TL/C/9313-11


COMPARE:
SET UP COMPARE BUFFER POINTER BASE ADDRESS
IS RECEIVED OFFSET AT TOP OF RECEIVE BUFFER ?
YES
LOAD LAST BYTE POSITION OFFSET INTO RECEIVER BYTE POINTER
NO
DECREMENT RECEIVE OFFSET
LOAD COMPARE BUFFER OFFSET
COMPARE DATA SENT WITH DATA RECEIVER
DOES DATA MATCH ?
NO
SET ERROR STROBE SIGNAL
CALL SERVICE ROUTINE TO DISPLAY "DATA MISMATCH MESSAGE"
STOP
YES
IS THIS THE END OF THE COMPARE BUFFER ?
YES
INCREMENT TRANSMITTER BUFFER COUNT
RESET COMPARE BUFFER OFFSET
NO
INCREMENT COMPARE BUFFER OFFSET
IS THE COMPARE BUFFER OFFSET = TO THE RECEIVE BUFFER OFFSET ?
YES
RETURN FROM BRANCH
NO

TL/C/9313–12

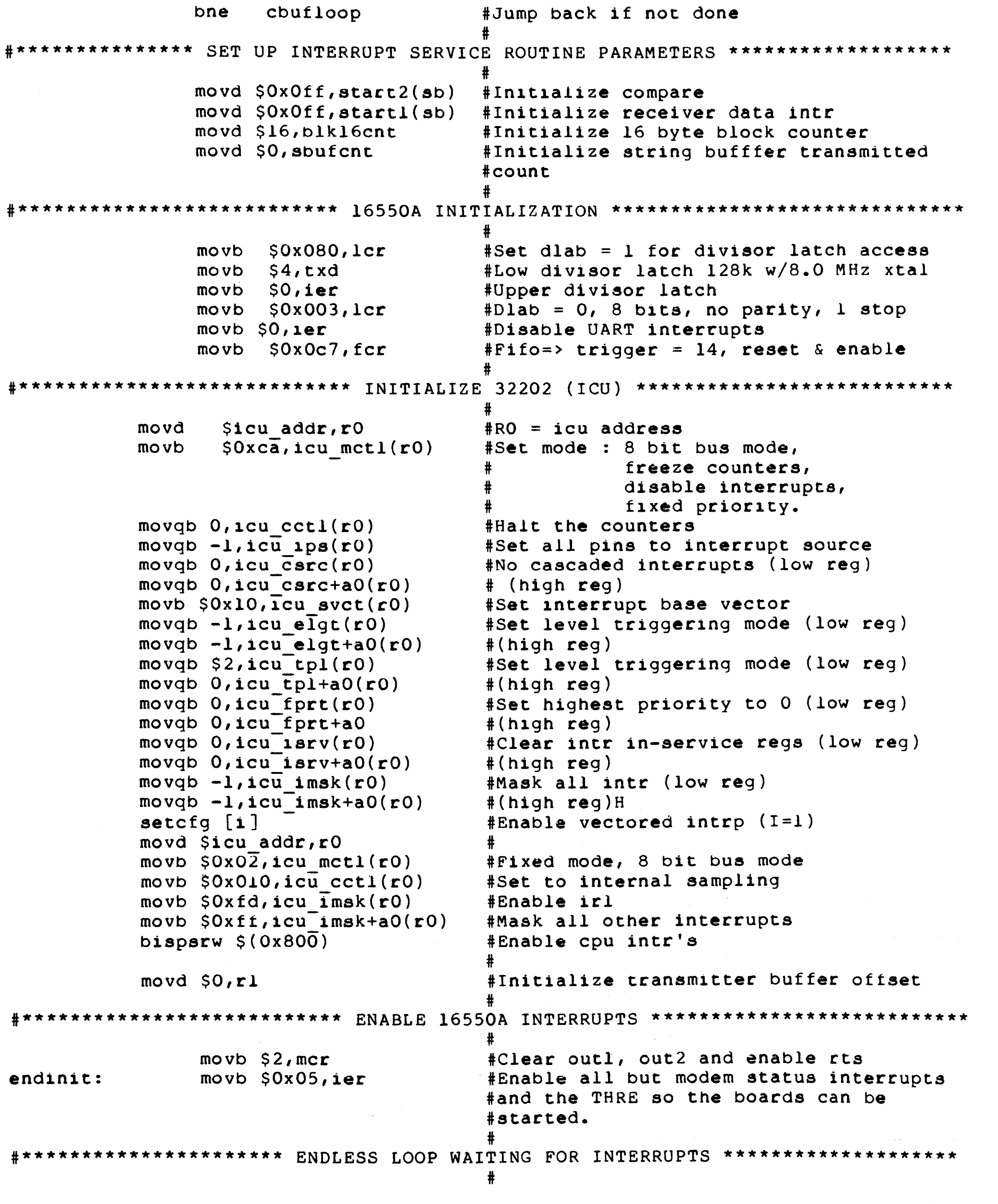


```
#3/30/87.....D1APPS.ASM.........ADAPTED ORIGINALLY FROM D1RON56K.ASM
#
#THIS PROGRAM RUNS USING 2 DB32000 BOARDS WITH 16550As ENABLED AT ADDRESS 0d00000
#WIRE-WRAPPED ON THE BOARDS. THIS SOFTWARE TRANSMITS THE DATA 00 THROUGH FF
#REPEATEDLY TO THE REMOTE UART AND EXPECTS TO REPEATEDLY RECEIVE THE DATA FF
#THROUGH 00 FROM THE REMOTE UART. IT SHOULD BE RUN IN CONJUNCTION WITH THE
#PROGRAM D2APPSC.ASM RUNNING ON THE OTHER DB32000 BOARD. THE TX PIN OF
#THIS 16550A SHOULD CONNECT TO THE RX PIN OF THE 16550A ON THE OTHER BOARD AND
#VICE VERSA. ALSO, THE CTS PIN OF THIS 16550A SHOULD BE CONNECTED TO THE RTS PIN
#OF THE 16550A ON THE OTHER BOARD AND VICE VERSA. THIS WILL ENABLE THE
# APPROPRIATE HANDSHAKES TO OCCUR.
#
#TO RUN THIS PROGRAM YOU MUST:
#
#        1. CONNECT THE RX & TX OF THE 2 16550As ON THE 2 DB32000 BOARDS
#        2. CONNECT THE CTS & RTS OF THE 2 16550As ON THE 2 DB32000 BOARDS
#        3. DOWNLOAD D1APPS.EXE TO THIS BOARD VIA THE GNX DEBUGGER [REV 1.02]
#        4. DOWNLOAD D2APPS.EXE TO OTHER BOARD VIA THE GNX DEBUGGER [REV 1.02]
#        5. START D1APPS.EXE RUNNING ON THIS DB32000 BOARD
#        6. START D2APPS.EXE RUNNING ON THE OTHER DB32000 BOARD
#
#PROGRAM DETAILS:
#
#
# ISR contains the TX SERVICE ROUTINE
#
# TX OVERWRITES are PREVENTED by the ICU
#
# TX FIFO is CLEARED before a transmission
#
# DATA SENT  00 ------ FF
#
# DATA RECEIVED  and COMPARED FF ------ 00
#
# BAUDRATE 128k WITH A 8.0 MHZ XTAL INPUT TO THE 16550A
                                                #
#*********************** ESTABLISH 16550A REGISTER ADDRESSES ********************
                                                #
        .globl          isr                     #
           .set rxd,     0x0d00000              #Equate registers to their addresses
           .set txd,     0x0d00000              #
           .set ier,     0x0d00004              #
           .set iir,     0x0d00008              #
           .set fcr,     0x0d00008              #
           .set lcr,     0x0d0000c              #
           .set mcr,     0x0d00010              #
           .set lsr,     0x0d00014              #
           .set msreg,   0x0d00018              #
           .set scr,     0x0d0001c              #
                                                #
#******************* ESTABLISH ADDRESSES FOR THE 32202 (ICU) ******************
                                                #
           .set a0,4                            #Establish address alignment
                                                #between CPU and ICU
           .set icu_hvct,0                      #ICU register addresses
           .set icu_svct,1 *a0                  #
           .set icu_elgt,2 *a0                  #
           .set icu_tpl,4  *a0                  #
           .set icu_ipnd,6 *a0                  #
           .set icu_isrv,8 *a0                  #
```

TL/C/9313-13

```
           .set icu_imsk,10 *a0          #
           .set icu_csrc,12 *a0          #
           .set icu_fprt,14 *a0          #
           .set icu_mctl,16 *a0          #
           .set icu_ciptr,18 *a0         #
           .set icu_pdat,19  *a0         #
           .set icu_ips,20   *a0         #
           .set icu_pdir,21  *a0         #
           .set icu_cctl,22  *a0         #
           .set icu_cictl,23 *a0         #
                                         #
                                         #First ICU register address
                                         #
           .set icu_addr,0xfffe00        #
                                         #
#************************* STATIC BASE STARTING LOCATIONS **********************
                                         #
           .set irl_mod, 17*4            #
           .set irl_off, 17*4+2          #
           .set start2, 0x0              #The following are static base variables
           .set start1, 0x0a             #used as base pointers. Start1/2 = flags
           .set txflag, 0x14             #txflaf = flag, sbuf = area used to
           .set sbuf, 0x1e               #store data to be transmitted, rbuf =
           .set rbuf, 0x41e              #area used to store received data,
           .set cbuf, 0x61e              #cbuf = area used to store compare
           .set intable, 0x81e           #buffer, intable = base pointer to the
                                         #interrupt table
                                         #
#********************** SET UP DISPATCH TABLE FOR THE 32032 ********************
                                         #
start::         bicpsrw $(0x100)         #Clear intr's
                movd $0x0c,r0            #Set for monitor svc to move intbase
                movd $0x055555555,r1     #from ROM to ram because you have
                addr intable(sb),r2      #to change the address for the
                movd $0x0c,r3            #interrupt service routine.
                svc                      #Actual svc for move
                sprd intbase,r2          #Put base addr of intbase in r2
                movd isrent,irl_mod(r2)  #Put offset of isr into 1st location
                                         #of dispatch table
                                         #
#********************* LOAD TRANSMITTER BUFFER (00 to FF) **********************
                                         #
senddat:        addr  sbuf(sb),r0        #R0 contains string buffer ptr.
                movd  $0,r1              #R1 contains offset
                movb  $0,r2              #Init data reg.
sbufloop:       movb  r2,0(r0)[r1:b]     #Load char. to string buffer
                addqw 1,r1               #Increment offset ptr.
                addqw 1,r2               #Increment data
                cmpw  r1,$256            #Check for 256 chars. loaded
                bne   sbufloop           #Jump back if not done
                                         #
#*********************** LOAD COMPARISON BUFFER (FF TO 00)**********************
                                         #
compdat:        addr  cbuf(sb),r0        #R0 contains pointer
                movd  $0,r1              #R1 contains offset
                movb  $0x0ff,r2          #Init data reg.
cbufloop:       movb  r2,0(r0)[r1:b]     #Load char. to compare buffer
                addqw 1,r1               #Increment ptr. offset
                subb  $1,r2              #Decrement data
                cmpw  r1,$256            #Check for 256 chars. loaded
```

TL/C/9313-14

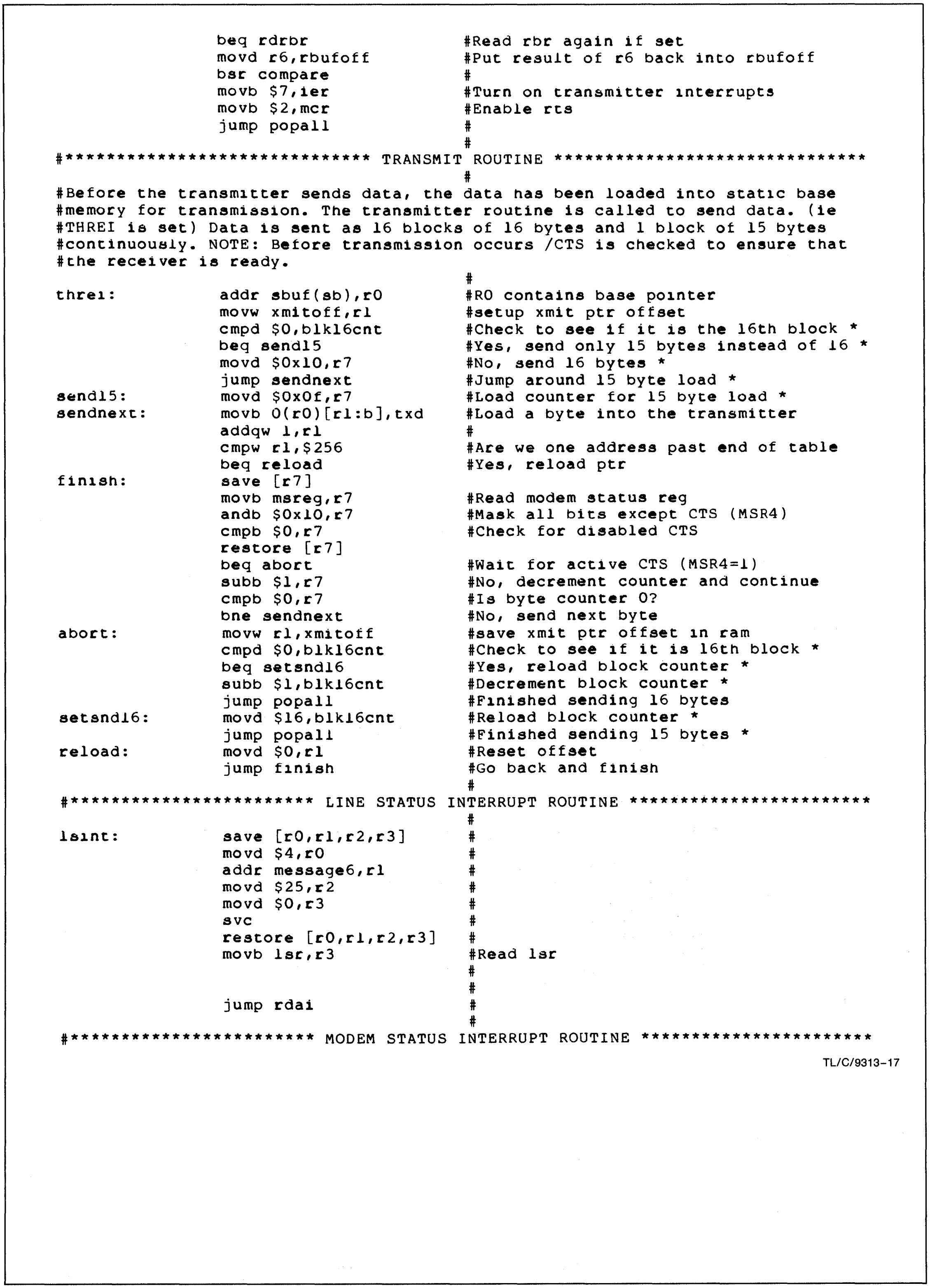


```
                bne     cbufloop            #Jump back if not done
                                            #
#*************** SET UP INTERRUPT SERVICE ROUTINE PARAMETERS ******************
                                            #
                movd $0x0ff,start2(sb)      #Initialize compare
                movd $0x0ff,start1(sb)      #Initialize receiver data intr
                movd $16,blk16cnt           #Initialize 16 byte block counter
                movd $0,sbufcnt             #Initialize string bufffer transmitted
                                            #count
                                            #
#*************************** 16550A INITIALIZATION ******************************
                                            #
                movb   $0x080,lcr           #Set dlab = 1 for divisor latch access
                movb   $4,txd               #Low divisor latch 128k w/8.0 MHz xtal
                movb   $0,ier               #Upper divisor latch
                movb   $0x003,lcr           #Dlab = 0, 8 bits, no parity, 1 stop
                movb $0,ier                 #Disable UART interrupts
                movb   $0x0c7,fcr           #Fifo=> trigger = 14, reset & enable
                                            #
#**************************** INITIALIZE 32202 (ICU) ***************************
                                            #
           movd    $icu_addr,r0             #R0 = icu address
           movb    $0xca,icu_mctl(r0)       #Set mode : 8 bit bus mode,
                                            #          freeze counters,
                                            #          disable interrupts,
                                            #          fixed priority.
           movqb 0,icu_cctl(r0)             #Halt the counters
           movqb -1,icu_ips(r0)             #Set all pins to interrupt source
           movqb 0,icu_csrc(r0)             #No cascaded interrupts (low reg)
           movqb 0,icu_csrc+a0(r0)          # (high reg)
           movb $0x10,icu_svct(r0)          #Set interrupt base vector
           movqb -1,icu_elgt(r0)            #Set level triggering mode (low reg)
           movqb -1,icu_elgt+a0(r0)         #(high reg)
           movqb $2,icu_tpl(r0)             #Set level triggering mode (low reg)
           movqb 0,icu_tpl+a0(r0)           #(high reg)
           movqb 0,icu_fprt(r0)             #Set highest priority to 0 (low reg)
           movqb 0,icu_fprt+a0              #(high reg)
           movqb 0,icu_isrv(r0)             #Clear intr in-service regs (low reg)
           movqb 0,icu_isrv+a0(r0)          #(high reg)
           movqb -1,icu_imsk(r0)            #Mask all intr (low reg)
           movqb -1,icu_imsk+a0(r0)         #(high reg)H
           setcfg [i]                       #Enable vectored intrp (I=1)
           movd $icu_addr,r0                #
           movb $0x02,icu_mctl(r0)          #Fixed mode, 8 bit bus mode
           movb $0x010,icu_cctl(r0)         #Set to internal sampling
           movb $0xfd,icu_imsk(r0)          #Enable ir1
           movb $0xff,icu_imsk+a0(r0)       #Mask all other interrupts
           bispsrw $(0x800)                 #Enable cpu intr's
                                            #
           movd $0,r1                       #Initialize transmitter buffer offset
                                            #
#**************************** ENABLE 16550A INTERRUPTS ****************************
                                            #
                movb $2,mcr                 #Clear out1, out2 and enable rts
endinit:        movb $0x05,ier              #Enable all but modem status interrupts
                                            #and the THRE so the boards can be
                                            #started.
                                            #
#*********************** ENDLESS LOOP WAITING FOR INTERRUPTS ********************
                                            #
```

TL/C/9313-15

```
holdloop:       nop                     #
                br holdloop             #
                                        #
#****************************** INTERRUPT HANDLER ******************************
                                        #
isr:            save [r0,r1,r2,r3,r4,r5,r6,r7]
                movb iir,r0             #R0- contains iir
                cmpb r0,$0x0c6          #
                beq lsint               #Line status interrupt
                cmpb r0,$0x0c4          #
                beq rdai                #Receiver interrupt
                cmpb r0,$0x0cc          #
                beq rtmout              #Rec timeout interrupt
                cmpb r0,$0x0c2          #
                beq threi               #THRE interrupt
                cmpb r0,$0x0c0          #
                beq msint               #Modem status interrupt
                                        #
                                        #
#************************** INVALID INTERRUPT ROUTINE **************************
                                        #
                save [r0,r1,r2,r3]      #
                movd $4,r0              #
                addr message2,r1        #
                movd $21,r2             #
                movd $0,r3              #
                svc                     #
                restore [r0,r1,r2,r3]   #
                                        #
                                        #
                jump stop               #Restore all registers
                                        #
                                        #
#********************* RECEIVER TIMEOUT INTERRUPT ROUTINE **********************
                                        #
rtmout:         jump rdai
                                        #
#*************************** RECEIVER INTERRUPT ROUTINE ************************
                                        #
#This portion of the program is reached by a positive test for the received data
#available interrupt. Once in this routine each byte is removed from the FIFO,
#placed in a designated static base memory location and the LSR is tested to see
#if the data ready (DR) bit is still set. Data is removed from the FIFO and
#placed in memory until the DR bit is no longer set. The data sent will be
#compared to known data, located in another designated static base location, by
#calling the compare subroutine.
                                        #
rdai:           movb $0,mcr             #Disable RTS; stop transmission
                addr rbuf(sb),r4        #r4 contains rbuf base address
                movd rbufoff,r6         #Put rbuf offset runner into r6
rdrbr:          movb rxd,0(r4)[r6:b]    #Store a byte in the receiver buffer
                cmpb $0x00,0(r4)[r6:b]  #Is it the last character
                addqw 1,r6              #Increment offset ptr.
                addqw 1,rbufoff         #Track r6
                bne continue            #
                movw $0,r6              #Reset pointer offset
                movw $0,rbufoff         #Reset rbufoff
continue:       movb lsr,r3             #Read lsr
                andb $01,r3             #Mask all but bit 0
                cmpb $01,r3             #
```

TL/C/9313-16

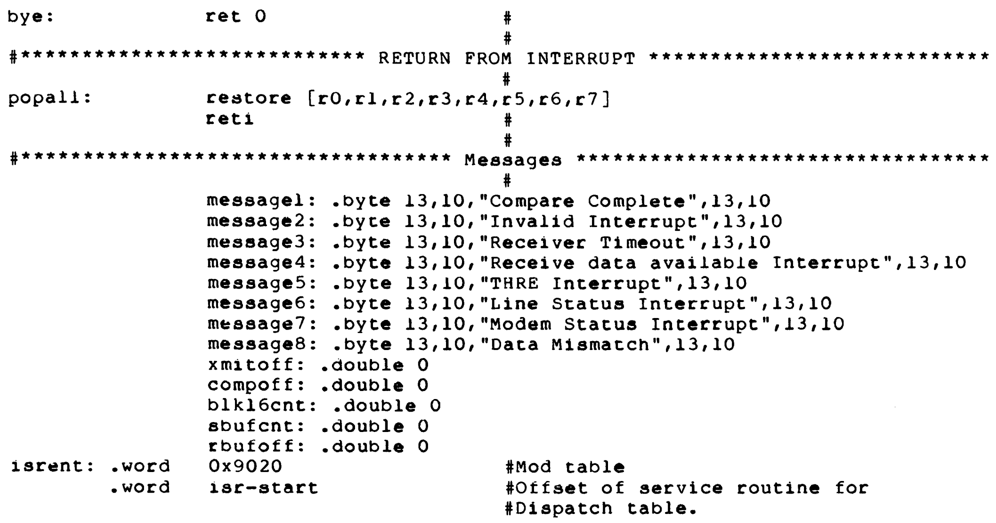


```
                beq rdrbr               #Read rbr again if set
                movd r6,rbufoff         #Put result of r6 back into rbufoff
                bsr compare             #
                movb $7,ier             #Turn on transmitter interrupts
                movb $2,mcr             #Enable rts
                jump popall             #
                                        #
#****************************** TRANSMIT ROUTINE ******************************
                                        #
#Before the transmitter sends data, the data has been loaded into static base
#memory for transmission. The transmitter routine is called to send data. (ie
#THREI is set) Data is sent as 16 blocks of 16 bytes and 1 block of 15 bytes
#continuously. NOTE: Before transmission occurs /CTS is checked to ensure that
#the receiver is ready.
                                        #
threi:          addr sbuf(sb),r0        #R0 contains base pointer
                movw xmitoff,r1         #setup xmit ptr offset
                cmpd $0,blk16cnt        #Check to see if it is the 16th block *
                beq send15              #Yes, send only 15 bytes instead of 16 *
                movd $0x10,r7           #No, send 16 bytes *
                jump sendnext           #Jump around 15 byte load *
send15:         movd $0x0f,r7           #Load counter for 15 byte load *
sendnext:       movb 0(r0)[r1:b],txd    #Load a byte into the transmitter
                addqw 1,r1              #
                cmpw r1,$256            #Are we one address past end of table
                beq reload              #Yes, reload ptr
finish:         save [r7]
                movb msreg,r7           #Read modem status reg
                andb $0x10,r7           #Mask all bits except CTS (MSR4)
                cmpb $0,r7              #Check for disabled CTS
                restore [r7]
                beq abort               #Wait for active CTS (MSR4=1)
                subb $1,r7              #No, decrement counter and continue
                cmpb $0,r7              #Is byte counter 0?
                bne sendnext            #No, send next byte
abort:          movw r1,xmitoff         #save xmit ptr offset in ram
                cmpd $0,blk16cnt        #Check to see if it is 16th block *
                beq setsnd16            #Yes, reload block counter *
                subb $1,blk16cnt        #Decrement block counter *
                jump popall             #Finished sending 16 bytes
setsnd16:       movd $16,blk16cnt       #Reload block counter *
                jump popall             #Finished sending 15 bytes *
reload:         movd $0,r1              #Reset offset
                jump finish             #Go back and finish
                                        #
#*********************** LINE STATUS INTERRUPT ROUTINE ***********************
                                        #
lsint:          save [r0,r1,r2,r3]      #
                movd $4,r0              #
                addr message6,r1        #
                movd $25,r2             #
                movd $0,r3              #
                svc                     #
                restore [r0,r1,r2,r3]   #
                movb lsr,r3             #Read lsr
                                        #
                                        #
                jump rdai               #
                                        #
#*********************** MODEM STATUS INTERRUPT ROUTINE **********************
```

TL/C/9313-17

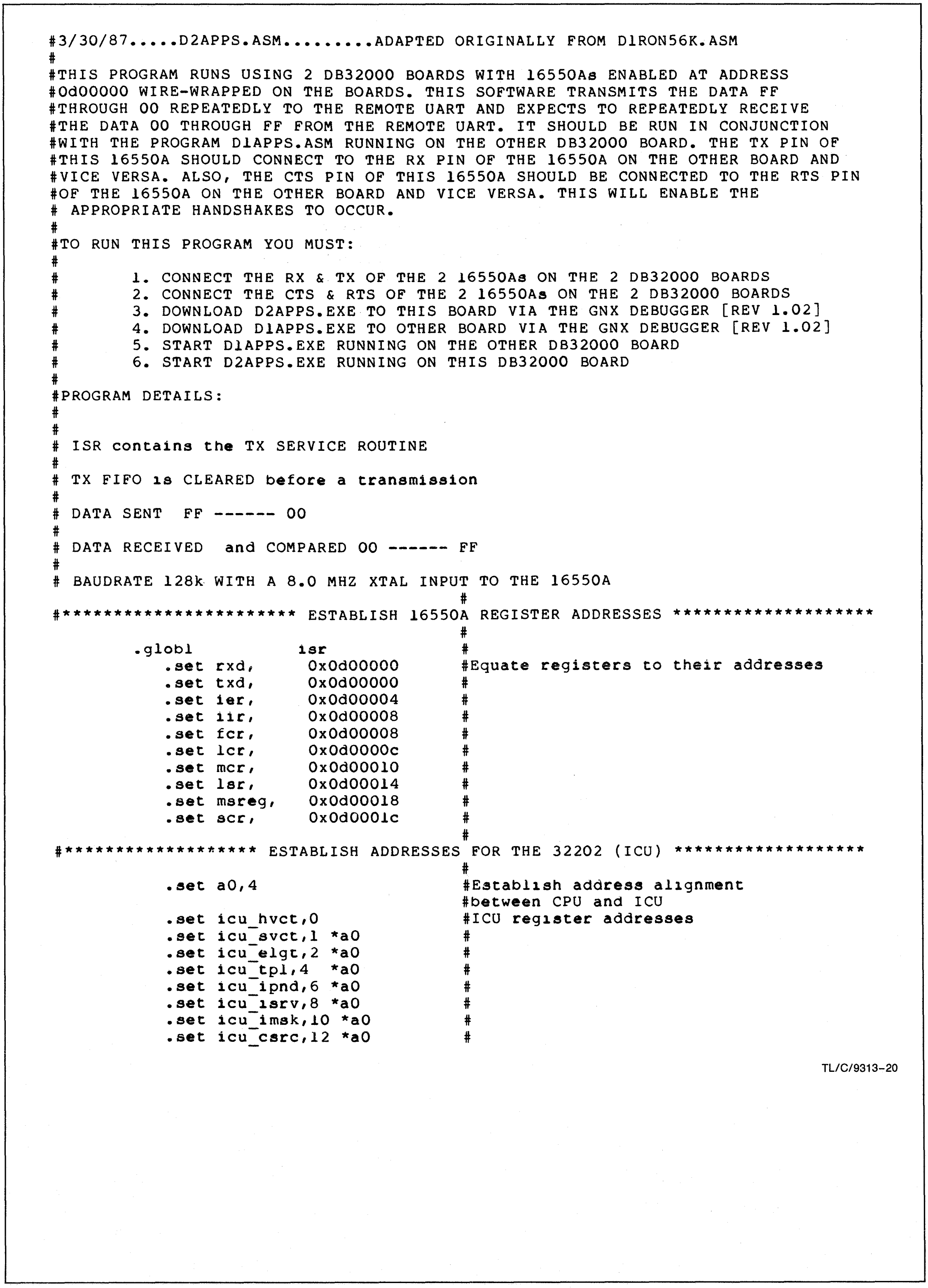
```
                                          #
msint:          save [r0,r1,r2,r3]        #
                movd $4,r0                #
                addr message7,r1          #
                movd $26,r2               #
                movd $0,r3                #
                svc                       #
                movb 0x0d00018,r0         #
                restore [r0,r1,r2,r3]     #
                jump popall               #
#****************************** COMPARE DATA ROUTINE *****************************
                                          #
#This subroutine is called by the receiver interrupt routine which has set the
#receiver offset (rbufoff) to point at the last byte received. This subroutine
#uses the compare offset (compoff) pointer as the pointer for both receive
#buffer data and compare buffer data. Each location is compared to ensure data
#sent is identical to data received. This is done until compoff equals rbufoff
#stopping the process and returning from the interrupt. NOTE: Data being
#received is known data and an exact copy is loaded into memory prior to any
#transmission.
                                          #
compare:        addr cbuf(sb),r1          #R1- base address of cbuf base
                cmpd $0,r6                #Check for potential invalid subtraction
                beq zeror6                #Jump around subtraction
                subd $1,r6                #
                jump compbyte             #Jump around subtraction fix
zeror6:         movd $0xff,r6             #
compbyte:       movd compoff,r5           #
                cmpb 0(r1)[r5:b],0(r4)[r5:b] #Compare data sent to data received
                bne wrong                 #Branch and set out1 if wrong
                                          #
                cmpb $0x00,0(r4)[r5:b]    #Check for end of buffer
                bne notend                #Branch and increment pointers
                jump reload1              #Test for having compared all bytes
                                          #
notend:         addd $1,compoff           #Increment pointer
notend1:        cmpd r5,r6                #
                beq bye                   #
                jump compbyte             #
                                          #
reload1:        addd $1,sbufcnt           #Increment transmiter cnt
                movd $0,compoff           #Reload offset of pointer
                jump notend1              #
                                          #
wrong:          nop                       #
                movb $0x0c,mcr            #Set out 2, for error strobe
                                          #
#************************* DATA MISMATCH MESSAGE *******************************
                                          #
                save [r0,r1,r2,r3]        #Save register for supervisor call
                movd $4,r0                #Value required by svc call
                addr message8,r1          #Mover address of message into r1
                movd $17,r2               #Number of characters into r2
                movd $0,r3                #Value required by svc call
                svc                       #Actual call
                restore [r0,r1,r2,r3]     #Restore registers
                                          #
stop:           nop                       #
                jump stop                 #Test point
                                          #
```

TL/C/9313-18



```
bye:            ret 0                     #
                                          #
#***************************** RETURN FROM INTERRUPT *****************************
                                          #
popall:         restore [r0,r1,r2,r3,r4,r5,r6,r7]
                reti                      #
                                          #
#************************************ Messages ************************************
                                          #
                message1: .byte 13,10,"Compare Complete",13,10
                message2: .byte 13,10,"Invalid Interrupt",13,10
                message3: .byte 13,10,"Receiver Timeout",13,10
                message4: .byte 13,10,"Receive data available Interrupt",13,10
                message5: .byte 13,10,"THRE Interrupt",13,10
                message6: .byte 13,10,"Line Status Interrupt",13,10
                message7: .byte 13,10,"Modem Status Interrupt",13,10
                message8: .byte 13,10,"Data Mismatch",13,10
                xmitoff: .double 0
                compoff: .double 0
                blk16cnt: .double 0
                sbufcnt: .double 0
                rbufoff: .double 0
isrent: .word   0x9020                    #Mod table
        .word   isr-start                 #Offset of service routine for
                                          #Dispatch table.
```

TL/C/9313-19



```
#3/30/87.....D2APPS.ASM.........ADAPTED ORIGINALLY FROM D1RON56K.ASM
#
#THIS PROGRAM RUNS USING 2 DB32000 BOARDS WITH 16550As ENABLED AT ADDRESS
#0d00000 WIRE-WRAPPED ON THE BOARDS. THIS SOFTWARE TRANSMITS THE DATA FF
#THROUGH 00 REPEATEDLY TO THE REMOTE UART AND EXPECTS TO REPEATEDLY RECEIVE
#THE DATA 00 THROUGH FF FROM THE REMOTE UART. IT SHOULD BE RUN IN CONJUNCTION
#WITH THE PROGRAM D1APPS.ASM RUNNING ON THE OTHER DB32000 BOARD. THE TX PIN OF
#THIS 16550A SHOULD CONNECT TO THE RX PIN OF THE 16550A ON THE OTHER BOARD AND
#VICE VERSA. ALSO, THE CTS PIN OF THIS 16550A SHOULD BE CONNECTED TO THE RTS PIN
#OF THE 16550A ON THE OTHER BOARD AND VICE VERSA. THIS WILL ENABLE THE
# APPROPRIATE HANDSHAKES TO OCCUR.
#
#TO RUN THIS PROGRAM YOU MUST:
#
#        1. CONNECT THE RX & TX OF THE 2 16550As ON THE 2 DB32000 BOARDS
#        2. CONNECT THE CTS & RTS OF THE 2 16550As ON THE 2 DB32000 BOARDS
#        3. DOWNLOAD D2APPS.EXE TO THIS BOARD VIA THE GNX DEBUGGER [REV 1.02]
#        4. DOWNLOAD D1APPS.EXE TO OTHER BOARD VIA THE GNX DEBUGGER [REV 1.02]
#        5. START D1APPS.EXE RUNNING ON THE OTHER DB32000 BOARD
#        6. START D2APPS.EXE RUNNING ON THIS DB32000 BOARD
#
#PROGRAM DETAILS:
#
#
# ISR contains the TX SERVICE ROUTINE
#
# TX FIFO is CLEARED before a transmission
#
# DATA SENT  FF ------ 00
#
# DATA RECEIVED  and COMPARED 00 ------ FF
#
# BAUDRATE 128k WITH A 8.0 MHZ XTAL INPUT TO THE 16550A
                                                  #
#*********************** ESTABLISH 16550A REGISTER ADDRESSES *******************
                                                  #
        .globl          isr                       #
           .set rxd,     0x0d00000                #Equate registers to their addresses
           .set txd,     0x0d00000                #
           .set ier,     0x0d00004                #
           .set iir,     0x0d00008                #
           .set fcr,     0x0d00008                #
           .set lcr,     0x0d0000c                #
           .set mcr,     0x0d00010                #
           .set lsr,     0x0d00014                #
           .set msreg,   0x0d00018                #
           .set scr,     0x0d0001c                #
                                                  #
#******************* ESTABLISH ADDRESSES FOR THE 32202 (ICU) ******************
                                                  #
           .set a0,4                              #Establish address alignment
                                                  #between CPU and ICU
           .set icu_hvct,0                        #ICU register addresses
           .set icu_svct,1 *a0                    #
           .set icu_elgt,2 *a0                    #
           .set icu_tpl,4  *a0                    #
           .set icu_ipnd,6 *a0                    #
           .set icu_isrv,8 *a0                    #
           .set icu_imsk,10 *a0                   #
           .set icu_csrc,12 *a0                   #
```

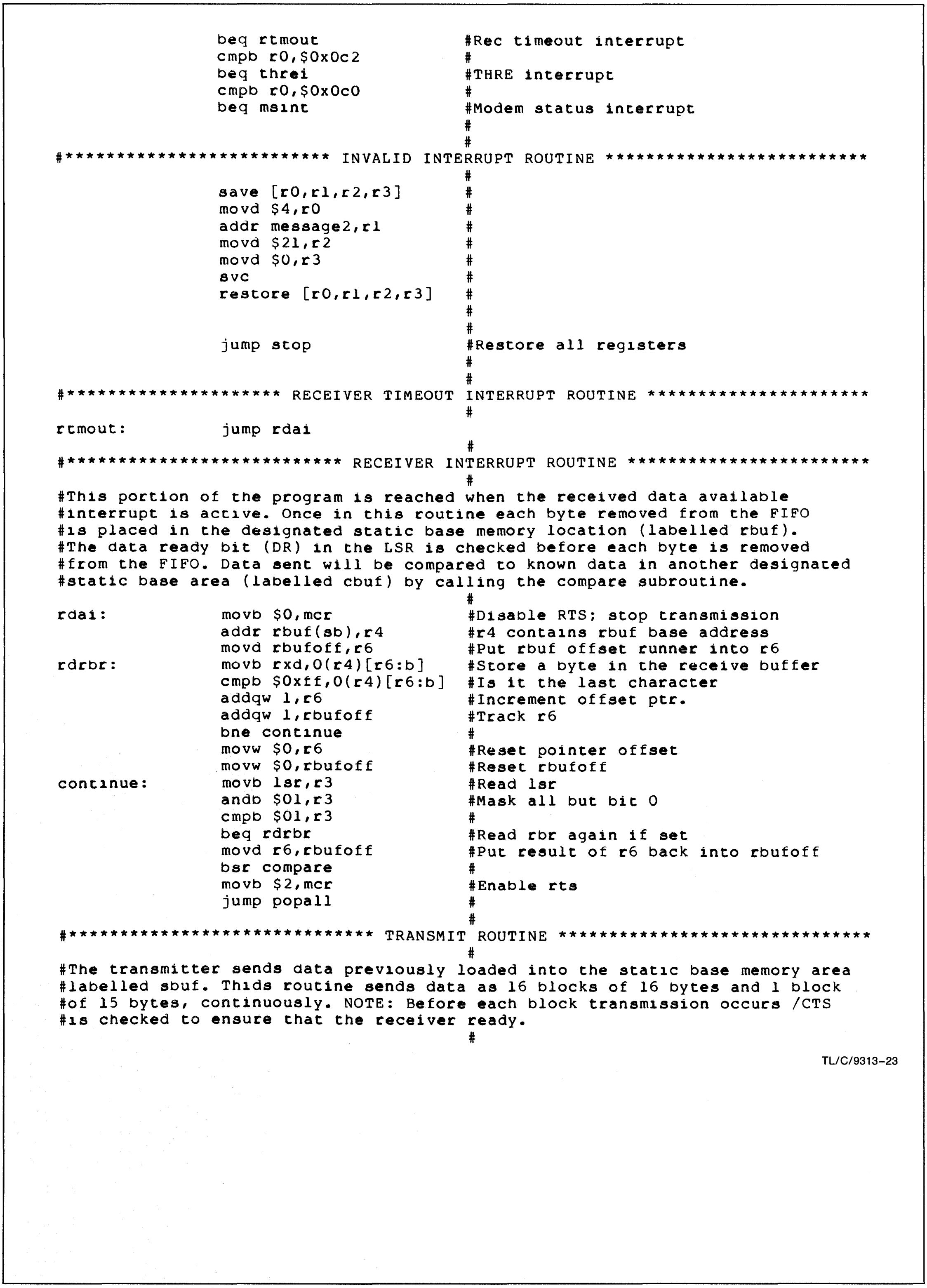
TL/C/9313-20

```
          .set icu_fprt,14 *a0        #
          .set icu_mctl,16 *a0        #
          .set icu_ciptr,18 *a0       #
          .set icu_pdat,19  *a0       #
          .set icu_ips,20   *a0       #
          .set icu_pdir,21  *a0       #
          .set icu_cctl,22  *a0       #
          .set icu_cictl,23 *a0       #
                                      #
                                      #First ICU register address
                                      #
          .set icu_addr,0xfffe00      #
                                      #
#************************** STATIC BASE STARTING LOCATIONS **********************
                                      #
          .set irl_mod, 17*4          #Dispatch table offset for IR1 entry
          .set sbuf, 0x1e             #sbuf = area used to
          .set rbuf, 0x41e            #store data to be transmitted, rbuf =
          .set cbuf, 0x61e            #area used to store received data,
          .set intable, 0x81e         #cbuf = area used to store compare
                                      #buffer, intable = base pointer to the
                                      #interrupt table
                                      #
#********************** SET UP DISPATCH TABLE FOR THE 32032 ********************
                                      #
start::        bicpsrw $(0x100)       #Clear intr's
               movd $0x0c,r0          #Set for monitor svc to move intbase
               movd $0x055555555,r1   #from ROM to ram because you have
               addr intable(sb),r2    #to change the address for the
               movd $0x0c,r3          #interrupt service routine.
               svc                    #Actual svc for move
               sprd intbase,r2        #Put base addr of intbase in r2
               movd isrent,irl_mod(r2) #Put offset of isr into 1st location
                                      #of dispatch table
                                      #
#********************* LOAD TRANSMITTER BUFFER (FF to 00) **********************
                                      #
senddat:       addr  sbuf(sb),r0      #R0 contains string buffer ptr.
               movd  $0,r1            #R1 contains offset
               movb  $0x0ff,r2        #Init data reg.
sbufloop:      movb  r2,0(r0)[r1:b]   #Load char. to string buffer
               addqw 1,r1             #Increment offset ptr.
               subb  $1,r2            #Increment data
               cmpw  r1,$256          #Check for 256 chars. loaded
               bne   sbufloop         #Jump back if not done
                                      #
#*********************** LOAD COMPARISON BUFFER (00 TO FF) *********************
                                      #
compdat:       addr  cbuf(sb),r0      #R0 contains pointer
               movd  $0,r1            #R1 contains offset
               movb  $0,r2            #Init data reg.
cbufloop:      movb  r2,0(r0)[r1:b]   #Load char. to compare buffer
               addqw 1,r1             #Increment ptr. offset
               addqw 1,r2             #Decrement data
               cmpw  r1,$256          #Check for 256 chars. loaded
               bne   cbufloop         #Jump back if not done
                                      #
#*************** SET UP INTERRUPT SERVICE ROUTINE PARAMETERS ******************
                                      #
               movd $16,blk16cnt      #Initialize 16 byte block counter
```

TL/C/9313-21

```
                                                            #
#**************************** 16550A INITIALIZATION *******************************
                                                            #
                  movb    $0x080,lcr            #Set dlab = 1 for divisor latch access
                  movb    $4,txd                #Low divisor latch 56k w/8.0 xtal
                  movb    $0,ier                #Upper divisor latch
                  movb    $0x003,lcr            #Dlab = 0, 8 bits, no parity, 1 stop
                  movb    $0,ier                #Disable UART interrupts
                  movb    $0x0c7,fcr            #Fifo=> trigger = 14, reset & enable
                                                            #
#***************************** INITIALIZE 32202 (ICU) ****************************
                                                            #
                                                            #
            movd    $icu_addr,r0                #R0 = icu address
            movb    $0xca,icu_mctl(r0)          #Set mode : 8 bit bus mode,
                                                #           freeze counters,
                                                #           disable interrupts,
                                                #           fixed priority.
            movqb 0,icu_cctl(r0)                #Halt the counters
            movqb -1,icu_ips(r0)                #Set all pins to interrupt source
            movqb 0,icu_csrc(r0)                #No cascaded interrupts (low reg)
            movqb 0,icu_csrc+a0(r0)             # (high reg)
            movb $0x10,icu_svct(r0)             #Set interrupt base vector
            movqb -1,icu_elgt(r0)               #Set level triggering (low reg)
            movqb -1,icu_elgt+a0(r0)            #(high reg)
            movqb $2,icu_tpl(r0)                #Set high polarity mode (low reg)
            movqb 0,icu_tpl+a0(r0)              #(high reg)
            movqb 0,icu_fprt(r0)                #Set highest priority to 0 (low reg)
            movqb 0,icu_fprt+a0                 #(high reg)
            movqb 0,icu_isrv(r0)                #Clear intr in-service regs (low reg)
            movqb 0,icu_isrv+a0(r0)             #(high reg)
            movqb -1,icu_imsk(r0)               #Mask all intr (low reg)
            movqb -1,icu_imsk+a0(r0)            #(high reg)H
            setcfg [i]                          #Enable vectored intrp (I=1)
            movd $icu_addr,r0                   #
            movb $0x02,icu_mctl(r0)             #Fixed mode, 8 bit bus mode
            movb $0x010,icu_cctl(r0)            #Set to internal sampling
            movb $0xfd,icu_imsk(r0)             #Enable irl
            movb $0xff,icu_imsk+a0(r0)          #Mask all other interrupts
            bispsrw $(0x800)                    #Enable cpu intr's
                                                            #
#**************************** ENABLE 16550A INTERRUPTS ***************************
                                                            #
                  movb $2,mcr                   #Clear out1, out2 and enable rts
endinit:          movb $0x07,ier                #Enable all but modem status interrupts
                                                            #
#********************** ENDLESS LOOP WAITING FOR INTERRUPTS *******************
                                                            #
holdloop:         nop                           #
                  br holdloop                   #
                                                            #
#******************************** INTERRUPT HANDLER *******************************
                                                            #
isr:              save [r0,r1,r2,r3,r4,r5,r6,r7]
                  movb iir,r0                   #R0- contains iir
                  cmpb r0,$0x0c6                #
                  beq lsint                     #Line status interrupt
                  cmpb r0,$0x0c4                #
                  beq rdai                      #Receiver interrupt
                  cmpb r0,$0x0cc                #
```

TL/C/9313-22



```
                beq rtmout               #Rec timeout interrupt
                cmpb r0,$0x0c2           #
                beq threi                #THRE interrupt
                cmpb r0,$0x0c0           #
                beq msint                #Modem status interrupt
                                         #
                                         #
#**************************** INVALID INTERRUPT ROUTINE ***************************
                                         #
                save [r0,r1,r2,r3]       #
                movd $4,r0               #
                addr message2,r1         #
                movd $21,r2              #
                movd $0,r3               #
                svc                      #
                restore [r0,r1,r2,r3]    #
                                         #
                                         #
                jump stop                #Restore all registers
                                         #
                                         #
#********************* RECEIVER TIMEOUT INTERRUPT ROUTINE **********************
                                         #
rtmout:         jump rdai
                                         #
#**************************** RECEIVER INTERRUPT ROUTINE ***********************
                                         #
#This portion of the program is reached when the received data available
#interrupt is active. Once in this routine each byte removed from the FIFO
#is placed in the designated static base memory location (labelled rbuf).
#The data ready bit (DR) in the LSR is checked before each byte is removed
#from the FIFO. Data sent will be compared to known data in another designated
#static base area (labelled cbuf) by calling the compare subroutine.
                                         #
rdai:           movb $0,mcr              #Disable RTS; stop transmission
                addr rbuf(sb),r4         #r4 contains rbuf base address
                movd rbufoff,r6          #Put rbuf offset runner into r6
rdrbr:          movb rxd,0(r4)[r6:b]     #Store a byte in the receive buffer
                cmpb $0xff,0(r4)[r6:b]   #Is it the last character
                addqw 1,r6               #Increment offset ptr.
                addqw 1,rbufoff          #Track r6
                bne continue             #
                movw $0,r6               #Reset pointer offset
                movw $0,rbufoff          #Reset rbufoff
continue:       movb lsr,r3              #Read lsr
                andb $01,r3              #Mask all but bit 0
                cmpb $01,r3              #
                beq rdrbr                #Read rbr again if set
                movd r6,rbufoff          #Put result of r6 back into rbufoff
                bsr compare              #
                movb $2,mcr              #Enable rts
                jump popall              #
                                         #
#****************************** TRANSMIT ROUTINE *******************************
                                         #
#The transmitter sends data previously loaded into the static base memory area
#labelled sbuf. Thids routine sends data as 16 blocks of 16 bytes and 1 block
#of 15 bytes, continuously. NOTE: Before each block transmission occurs /CTS
#is checked to ensure that the receiver ready.
                                         #
```

TL/C/9313-23

```
threi:          addr sbuf(sb),r0        #R0 contains base pointer
                movw xmitoff,r1         #setup xmit ptr offset
                cmpd $0,blk16cnt        #Check to see if it is the 16th block
                beq send15              #Yes, send only 15 bytes instead of 16
                movd $0x10,r7           #No, send 16 bytes
                jump sendnext           #Jump around 15 byte load
send15:         movd $0x0f,r7           #Load counter for 15 byte load
sendnext:       movb 0(r0)[r1:b],txd    #Load a byte into the transmitter
                addqw 1,r1              #
                cmpw r1,$256            #Are we one address past end of table
                beq reload              #Yes, reload ptr
finish:         save [r7]
                movb msreg,r7           #Read modem status reg
                andb $0x10,r7           #Mask all bits except CTS (MSR4)
                cmpb $0,r7              #Check for disabled CTS
                restore [r7]
                beq abort               #Leave on inactive CTS (MSR4=0)
                subb $1,r7              #No, decrement counter and continue
                cmpb $0,r7              #Is byte counter 0?
                bne sendnext            #No, send next byte
abort:          movw r1,xmitoff         #save xmit ptr offset in ram
                cmpd $0,blk16cnt        #Check to see if it is 16th block
                beq setsnd16            #Yes, reload block counter
                subb $1,blk16cnt        #Decrement block counter
                jump popall             #Finished sending 16 bytes
setsnd16:       movd $16,blk16cnt       #Reload block counter
                jump popall             #Finished sending 15 bytes
reload:         movd $0,r1              #Reset offset
                jump finish             #Go back and finish
                                        #
#************************ LINE STATUS INTERRUPT ROUTINE ************************
                                        #
lsint:          save [r0,r1,r2,r3]      #
                movd $4,r0              #
                addr message6,r1        #
                movd $25,r2             #
                movd $0,r3              #
                svc                     #
                restore [r0,r1,r2,r3]   #
                movb lsr,r3             #Read lsr
                jump rdai               #
                                        #
#************************ MODEM STATUS INTERRUPT ROUTINE ***********************
                                        #
msint:          save [r0,r1,r2,r3]      #
                movd $4,r0              #
                addr message7,r1        #
                movd $26,r2             #
                movd $0,r3              #
                svc                     #
                movb 0x0d00018,r0       #
                restore [r0,r1,r2,r3]   #
                jump popall             #
#****************************** COMPARE DATA ROUTINE ***************************
                                        #
#The receiver subroutine branches to this subroutine after it has removed all of
#the data from the Rx FIFO. The receive offset (rbufoff) is changed to point to
#the last byte received in rbuf. The compare offset (compoff) points to each
#byte in the receive buffer and its associated byte in the compare register.
#Compoff is incremented after each successful comparison and the comparisons
```

TL/C/9313-24

```
#end when compoff equals rbufoff. NOTE: Data being received by this test program
#is known data and a copy of it is loaded into cbuf before transmissions begin.
                                                  #
compare:          addr cbuf(sb),rl               #Rl- base address of cbuf base
                  cmpd $0,r6                      #Check for potential invalid subtraction
                  beq zeror6                      #Jump around subtraction
                  subd $1,r6                      #
                  jump compbyte                   #Jump around subtraction fix
zeror6:           movd $0xff,r6                   #
compbyte:         movd compoff,r5                 #
                  cmpb 0(rl)[r5:b],0(r4)[r5:b] #Compare data sent to data received
                  bne wrong                       #Branch and set outl if wrong
                                                  #
                                                  #
                  cmpb $0xff,0(r4)[r5:b]          #Check for end of buffer
                  bne notend                      #Branch and increment pointers
                  jump reloadl                    #Test for having compared all bytes
                                                  #
                                                  #
notend:           addd $1,compoff                 #Increment pointer
notendl:          cmpd r5,r6                      #
                  beq bye                         #
                  jump compbyte                   #
                                                  #
reloadl:          addd $1,sbufcnt                 #Increment transmiter cnt
                  movd $0,compoff                 #Reload offset of pointer
                  jump notendl                    #
                                                  #
wrong:            movb $0x0c,mcr                  #Set out 2, for error strobe
                                                  #
#************************* DATA MISMATCH MESSAGE ********************************
                                                  #
                  save [r0,rl,r2,r3]              #Save register for supervisor call
                  movd $4,r0                      #Value required by svc call
                  addr message8,rl                #Mover address of message into rl
                  movd $17,r2                     #Number of characters into r2
                  movd $0,r3                      #Value required by svc call
                  svc                             #Actual call
                  restore [r0,rl,r2,r3]           #Restore registers
                                                  #
stop:             nop                             #
                  jump stop                       #Test point
                                                  #
bye:              ret 0                           #
                                                  #
#***************************** RETURN FROM INTERRUPT *****************************
                                                  #
popall:           restore [r0,rl,r2,r3,r4,r5,r6,r7]
                  reti                            #
                                                  #
#************************************ Messages ***********************************
                                                  #
                  messagel: .byte 13,10,"Compare Complete",13,10
                  message2: .byte 13,10,"Invalid Interrupt",13,10
                  message3: .byte 13,10,"Receiver Timeout",13,10
                  message4: .byte 13,10,"Receive data available Interrupt",13,10
                  message5: .byte 13,10,"THRE Interrupt",13,10
                  message6: .byte 13,10,"Line Status Interrupt",13,10
                  message7: .byte 13,10,"Modem Status Interrupt",13,10
                  message8: .byte 13,10,"Data Mismatch",13,10
```

TL/C/9313-25

AN-493

A Comparison of the INS8250, NS16450 and NS16550A Series of UARTs

National Semiconductor Corp.
Application Note 493
Martin S. Michael

National currently produces seven versions of the INS8250 UART. Functionally, these parts appear to be the same, however, there are differences that the designer and purchaser need to understand. For each version, this document provides a brief overview of their distinct characteristics, a detailed function and timing section, a discussion of software compatibility issues and the AC timing parameters.

1.0 Part Summary

The seven versions currently produced are designated INS8250, INS8250-B, INS8250A, NS16450, INS82C50A, NS16C450, and NS16550A. These devices are grouped below by process type.

NMOS DEVICES

1. INS8250: This is the original version produced by National. It is the same part as the INS8250-B, but with faster CPU bus timings.
2. INS8250-B: This is the slower speed (CPU bus timing) version of the INS8250. It is used by many popular 8088-based microcomputers.

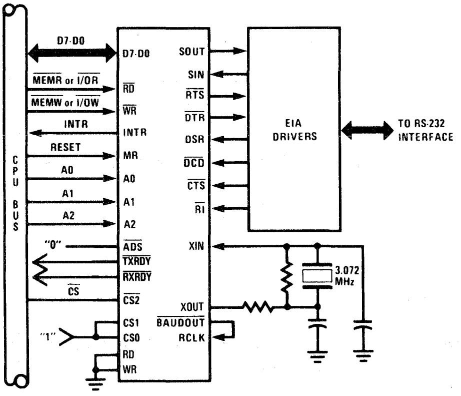
XMOS DEVICES

1. INS8250A: This is a revision of the INS8250 using the more advanced XMOS process. The INS8250A is better than the aforementioned parts due to the redesign (compare section 2.0 to 3.0) and the following process characteristics—closer threshold voltage control, more reliably implemented process topography and finer control over the active area critical dimensions. XMOS and CMOS parts should be used for all new designs. This part is used in many popular 8086-based microcomputers.
2. NS16450: This is the faster speed (CPU bus timing) version of the INS8250A. It is used by many popular 80286-based microcomputers.
3. NS16550A: This is the newest member of the UART family. It powers-up in the NS16450 mode and is completely compatible with all software written for the NS16450. It has advanced features such as on-board FIFOs, a DMA interface, faster CPU bus timings and a much higher maximum baud rate than the NS16450. The NS16550A should be used for all new non-CMOS designs, including those that were originally done with the NS16550. It is used in recent versions of popular 80286-based, 80386-based and ROMP-based microcomputers. Software written for the NS16550 is completely compatible with the NS16550A. Section 5.0 describes how the software can distinguish between the NS16550 and the NS16550A.
4. NS16550: This part powers-up in the NS16450 mode and is completely compatible with all software written for the NS16450. It has advanced features, such as a DMA interface. The on-board FIFOs are essentially non-functional. This part was issued on a limited basis. Any user that wants this part should order the NS16550A. Section 5.0 describes the differences between the NS16550 and the NS16550A in detail.

CMOS DEVICES

1. INS82C50A: This is a CMOS version of the INS8250A. It functions identically and for most AC parameters has the same timing specification as the INS8250A (see Section 4.0). It draws approximately 1/10 (10 mA) of the maximum operating current of the INS8250A.
2. NS16C450: This is a CMOS version of the NS16450. It functions identically and for most AC parameters has the same timing specification as the NS16450 (see Section 4.0). It draws approximately 1/12 (10 mA) of the maximum operating current of the NS16450.

Note: The XMOS and CMOS UARTs are not plug-in replacements for the INS8250/INS8250-B when used with ICUs that are in the popular edge-triggered configuration. However, there are easily implemented adjustments to the driving software or associated hardware that will allow these parts to be a plug-in replacement (see Section 6.0).



TL/C/9320–1

FIGURE 1. Connection Diagram

2.0 INS8250 and INS8250-B Functional Considerations

Designers using these NMOS parts should be well aware of the following considerations.

1. The Modem Status and Line Status registers are master-slave registers which transfer data from the master to the slave only when the INS8250, INS8250-B is not enabled. Thus, if the UART is never disabled:

   —The status registers are never updated.

   —The character in the transmit holding register will be transmitted repeatedly.

   —The CPU cannot read the current error status indication.

Recommendation: Disable the INS8250, INS8250-B between accesses.

2. At power-on the UART will occasionally transmit a random character. This occurs after release of the master reset and before it receives data from the CPU. The THRE and TSRE bits are unreliable at this time, due to their unpredictable state at power-on.

   Recommendation A: Use the following initialization routine:

   —Master reset.

   —Enable loopback mode (this causes any randomly sent characters to be sent to the receiver).

   —Load baud rate generator and initialize line control register.

   —Wait one character time and then clear the receiver buffer by reading it and clear any errors by reading the line status and modem status registers.

   —Disable the loopback mode.

The INS8250, INS8250-B is now initialized for normal operation and the THRE and TSRE bits are reliable. This procedure can be used with the INS8250A, NS16450 and INS82C50A, although it is unnecessary.

Recommendation B: Use one of the modem output lines to gate the transmitter data line.

3. When the transmitter interrupt is enabled, an interrupt occurs immediately regardless of the transmitter holding register's state. Furthermore, the first valid interrupt condition will probably be missed.

Recommendation: Use the following procedure to solve this problem:

   —Wait for the transmitter holding register to empty.

   —Disable microprocessor interrupts.

   —Write to the interrupt enable register. The interrupt indication that would normally appear at this time will be cleared by a previously stored reset, if the IIR has been read prior to this

Note: Whenever the IIR register is read and an active THRE interrupt will be cleared. If no THRE interrupt is active then the first THRE interrupt after the reading of IIR will be cleared).

   —Write to the interrupt enable register, again. Since there is no read of the IIR before this second write IER, there will be no stored reset to clear the normal THRE interrupt.

   —Enable microprocessor interrupts.

4. If data is not valid before and after WR or $\overline{WR}$ is active, then the bits of the internal register being addressed may change unpredictably. This could temporarily change any programmable UART function controlled by the addressed register. This situation exists because the INS8250, -B accepts data via fall-through latches that are enabled by the WR or $\overline{WR}$ going active rather than latched on the trailing edge of WR or $\overline{WR}$. Examples of this are glitches on the modem control lines or a temporary break on the serial output line while a command is written to the MCR or the LCR registers.

   Recommendation A: To avoid these problems the data must be valid just before, throughout and just after activation of WR or $\overline{WR}$.

   When using an 8088, 8086, 6800 or 8048 microprocessor, delay the leading edge of the write strobe until the data is stable. The above precaution is unnecessary when using the 8080, the NSC800™ or the Z80 microprocessors. Designs using a 32016 or 80286 should use the 16450, which avoids this problem by not having fall-through latches (see Section 3.0, Item 1).

Note: The temporary break caused by a spurious glitch on LCR6 can also be avoided by setting the loopback mode prior to writing to the line control register.

5. The transmitter generates start bits longer than the rest of the data by approximately 1 μs. This is due to a look-ahead circuit that sends the start bit while data is being transferred from the transmitter holding register to the transmitter shift register. At 56 kB this causes a 6.25% error.

   Recommendation: Be aware that the last stop bit will be reduced by an equivalent amount of time (approximately 1 μs).

6. If the CPU is slow in servicing the UART it could read current status (LSR) and then the next data byte (RBR), instead of the current data byte. An example of this type of failure would be losing a received character without an overrun indication. This occurs when the CPU reads the receiver buffer when another character from the shift register was being transferred to it. UART registers are updated as soon as the received data is available (i.e., the receive buffer register is updated as soon as all of the data bits have been received, the parity flag is updated as soon as the parity bit is received, the overrun flag is updated as soon as the stop bit is received, etc.).

   Recommendation: The CPU must read the buffer sooner.

7. The transmitter character may be erroneous, if the INS8250, -B transmits with 5 data bits and 1 and ½ stop bits.

   Recommendation: Use only 1 stop bit.

8. Writing a "1" to bit 1 of the Interrupt Enable Register (IER1), when the Transmitter Holding Register is not empty sets the THRE interrupt, regardless of the THRE status bit condition.

   Recommendation: Only set bit 1 in the Interrupt Enable Register (IER1) if the Transmitter Holding Register is empty.

9. When multiple interrupts are pending, the interrupt line (INTR) pulses low after each interrupt instead of remaining high continuously.

   Recommendation: This will not cause problems in normal operation, however, it is a condition necessary for compatibility in some popular 8088-based microcomputers that use an edge-triggered ICU (see Section 6.0).

10. Bit No. 6 (TSRE) of the line status register is set as soon as the transmitter shift register empties whether or not the transmitter holding register contains a character. Bit No. 6 is then reset when the transmitter shift register is reloaded.

    Recommendation: This will not cause problems in normal operation. However, it is a function tested on some popular 8088-based microcomputer systems diagnostic programs.

2.1 ADDITIONAL FUNCTIONAL CONSIDERATIONS

When using the INS8250-B in full duplex operation with the THRE interrupt enabled and either one or both of the higher priority interrupts enabled (Receiver Data Available, Receiver Line Status), the THRE interrupt indication may be lost. This is only possible if both data transmission and reception are occurring simultaneously. To avoid this problem use one of the three software aids listed below. The first two should be inserted in the Receiver Data Available and/or Receiver Line Status service routines. The last one should be inserted in the THRE service routine. Any of the following will result in successful operation given the above circumstance.

SOFTWARE AIDS

1. While inside the higher order interrupt service routines; test the THRE bit, if it is 1 then set IER1.
2. While inside the higher order interrupt service routines; test the THRE bit, if it is 1 then set a flag and service the transmitter as soon as you exit the routine.
3. Poll THRE (LSR5) instead of using the IIR.

3.0 INS8250A and NS16450 Function and Timing Considerations

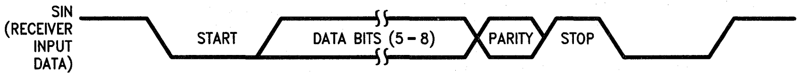
1. Chip select does not affect data transfers from the master register to the slave register. Therefore, the UART doesn't have to be deselected before it can offer valid status updates to the CPU.
2. The master reset (MR) input has a Schmitt Trigger circuit added to it.
3. A transmitter interrupt occurs only if the transmitter holding register is empty when bit 1 of the Interrupt Enable Register (IER) is set.
4. The UARTs latch data written to them on the trailing edge of the WR or $\overline{\text{WR}}$ signal, so data does not need to be valid for the total time write is active.
5. The loopback diagnostic function sets the modem control outputs $\overline{\text{RTS}}$, DTR, OUT1 and OUT2 to their inactive state (logic "1"), so they will send no spurious signals.
6. A one byte scratch pad register is included at location 111. This register is not on the INS8250 or -B.
7. When multiple interrupts are pending the interrupt line remains high rather than pulsing low after each interrupt is serviced. The INS8250A and NS16450 have level sensitive interrupts as opposed to edge-triggered interrupts. This requires a change in the UART driver software or associated hardware if the INS8250A, NS16450 is used with some popular microcomputers, and their edge-triggered ICUs (see Section 6.0).
8. Bit 6 of the line status register is set to 1 when both the transmitter holding and shift register are empty. This causes the INS8250A and NS16450 to be incompatible with some INS8250 software utilizing this bit.

3.1 TIMING CONSIDERATIONS

1. A start bit will be sent typically 16 clocks (1 bit time) after the WRTHR signal goes active.
2. The leading edge of WRTHR resets THRE and TEMT.
3. All of the line status errors and the received data flag (DR, data ready) are set during the time of the first stop bit.
4. TEMT is set 2 RCLK clock periods after the stop bit(s) are sent.
5. The modem control register updates the modem outputs on the trailing edge of WRMCR.

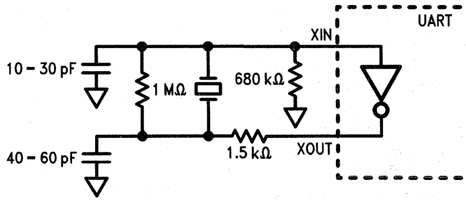
3.2 CRYSTAL REQUIREMENTS

There have been reports that certain types of 1.8432 MHz crystals have not been starting when used with the INS8250s (excluding the INS82C50A). The problem is with the smaller size versions of the crystal and their higher ESR values. In order to overcome this problem the following circuit should be used.



TL/C/9320–2

FIGURE 2. Serial Data Timing



TL/C/9320–3

FIGURE 3. The Oscillator Circuit

Crystal parameter for the above circuit are:

type	AT cut
resonance	fundamental (parallel)
load capacitor	20 pF – 32 pF
max. R_S	1k @ 1 MHz, 500 @ 5 MHz
cal. tolerance	+0.005% @ 25°C
drift tolerance	+0.005% @ 0°C – +70°C
overall tolerance	+0.01%

3.3 ADDITIONAL FUNCTIONAL CONSIDERATIONS

When using either the INS8250A or the NS16450 in full duplex operation with the THRE interrupt enabled and either one or both of the higher priority interrupts enable (Receiver Data Available, Receiver Line Status), the THRE interrupt indication may be lost. This is only possible if both data transmission and reception are occurring simultaneously. To avoid this problem use one of the three following software aids. The first two should be inserted in the Receiver Data Available and/or Receiver Line Status service routines. The last one should be inserted in the THRE service routine. Any of the following will result in successful operation given the above circumstance.

SOFTWARE AIDS

1. Disable and then reenable transmitter interrupts via IER1 after the last time the IIR is read in higher order interrupt service routines.
2. While inside the higher order interrupt service routines; test the THRE bit, if it is 1 then set a flag and service the transmitter as soon as you exit the routine.
3. Poll THRE (LSR5) instead of using the IIR.

4.0 INS82C50A and NS16C450 Function and Timing Considerations

All of the information presented in Sections 3.0 through 3.2 is applicable to the CMOS parts. In addition, the following items specify differences between XMOS and CMOS parts. They are applicable to the CMOS parts only:

1. Anytime a reset pulse is issued to the INS82C50A or NS16C450 the divisor latches must be rewritten with the appropriate divisors in order to start the baud rate generator.
2. t_{SI} is from 16 to 48 RCLK cycles in length

5.0 NS16550A and NS16550 Function and Timing Considerations

All of the information present in Sections 3.0 and 3.1 is applicable to the NS16550A and NS16550.

The primary difference between these two parts is in the operation of the FIFOs. The NS16550 will sometimes transfer extra characters when the CPU reads the RX FIFO. Due to the asynchronous nature of this failure there is no workaround and the NS16550 should NOT be used in the FIFO mode. The NS16550A has no problems operating in the FIFO mode and should be used on all new designs.

The programmer should note the difference in the function of bit 6 in the Interrupt Identification Register (IIR6). This bit is permanently at logical 0 in the NS16550. In the NS16550A this bit will be set to a 1 when the FIFOs are enabled. In both parts bit 7 of the IIR is set to a 1 when the FIFOs are enabled. Therefore, the program can distinguish when the FIFOs are enabled and whether the part is an NS16550A or an NS16550 by checking these two bits. In order to enable the FIFO mode and set IIR6 and IIR7 bit 0 of the FIFO Control Register (FCR0) should be set. Remember unless both bits IIR6 and IIR7 are set, the program should not transfer data via the FIFOs.

The following are improvements in the AC timings for the NS16550A over the NS16450:

1. t_{AR} changes from 60 ns to 30 ns.
2. t_{CSW} changes from 50 ns to 30 ns.
3. t_{CSR} changes from 50 ns to 30 ns.
4. RC changes from 360 ns to 280 ns.
5. t_{RC} changes from 175 ns to 125 ns.
6. t_{DS} changes from 40 ns to 30 ns.
7. t_{DH} changes from 40 ns to 30 ns.
8. Timing parameters specified by t_{SINT} will change in some cases when the FIFOs are enabled. Refer to the data sheet for specific changes.

6.0 Software Compatibility

The first part produced (INS8250-B) had some flaws and the first revision of that part (INS8250A) resolved those flaws. Between the time of the first part and the first revision, use of the INS8250-B in personal computers became quite common. Two of the conditions present in the INS8250-B are required in many of these personal computers (see Items 9 and 10 in Section 2.0). These two detect multiple pending interrupts from the INS8250-B and test the baud rate. These two conditions were eliminated in the revision part and all parts thereafter. Thus, the more recent UARTs require that one of the following recommendations or a similar change is made to the target system. Changing the software or hardware allows the more recent UARTs to replace the INS8250-B. If the target system services the UART via polling rather than interrupts, then all of the more recent parts will be plug-in replacements for the INS8250-B.

Note: The NS16550A has two pins with new functions (see the data sheet for specifics).

6.1 USING THE INS8250A, NS16450, INS82C50A, NS16C450 AND NS16550A WITH EDGED-TRIGGERED ICUs

Using these UARTs with an edge-triggered ICU as in some of the popular microcomputers requires a signal edge on the INTR pin for each pending UART interrupt. Otherwise, when multiple interrupts are pending the interrupt line will be constantly high active and the edge-triggered ICU will not request additional service for the UART.

6.2 CREATING AN INTERRUPT EDGE VIA SOFTWARE

This is done by disabling and then re-enabling UART interrupts via the Interrupt Enable Register (IER) before a specific UART interrupt handling routine (line status errors, received data available, transmitter holding register empty or modem status) is exited. To disable interrupts write H'00 to the IER. To re-enable interrupts write a byte containing ones to the IER bit positions whose interrupts are supposed to be enabled.

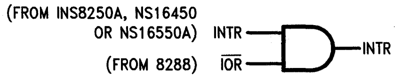
6.3 CREATING AN INTERRUPT EDGE IN HARDWARE

This is done externally to the UART. One approach is to connect the INTR pin of the UART to the input of an AND gate. The other input of this AND gate is connected to a signal that will always go low active when the UART is accessed (see *Figure 4*). The output of the AND gate is used as the interrupt to the ICU.

Note: This simple hardware recommendation will result in one invalid interrupt being generated, so the software routine should be able to handle this. The example shown below was tested using a modified asynchronous card in a few 8088-based microcomputer systems.

7.0 Acknowledgements

The editor expresses his gratitude to all of the applications, design and field applications engineers whose laboratory and field research have discovered most of the technical information used in this document.



TL/C/9320–4

FIGURE 4: Creating an INTR Edge in Hardware

AC Electrical Characteristics $T_A = 0°C$ to $+70°C$, $V_{CC} = 5V \pm 5\%$

Symbol	Parameter	Conditions	NS16550A		NS16450 NS16C450		INS8250A INS82C50A		INS8250		INS8250-B		Units
			Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	
t_{ADS}	Address Strobe Width		60		60		90		90		120		ns
t_{AH}	Address Hold Time		0		0		0		0		60		ns
t_{AR}	RD/$\overline{\text{RD}}$ Delay from Address	(Note 1)	30		60		80		110		110		ns
t_{AS}	Address Setup Time		60		60		90		110		110		ns
t_{AW}	WR/$\overline{\text{WR}}$ Delay from Address	(Note 1)	30		60		80		160		160		ns
t_{CH}	Chip Select Hold Time		0		0		0		0		60		ns
t_{CS}	Chip Select Setup Time		60		60		90		110		110		ns
t_{CSC}	Chip Select Output Delay from Select	(Notes 1, 8)		NA		100		125		200		200	ns
t_{CSR}	RD/$\overline{\text{RD}}$ Delay from Chip Select	(Note 1)	30		50		80		110		110		ns
t_{CSS}	Chip Select Output Delay from Strobe			NA		NA		NA	0	150	0	150	ns
t_{CSW}	WR/$\overline{\text{WR}}$ Delay from Select	(Note 1)	30		50		80		160		160		ns
t_{DH}	Data Hold Time		30		40		60		60		100		ns
t_{DS}	Data Setup Time		30		40		90		175		350		ns
t_{HZ}	RD/$\overline{\text{RD}}$ to Floating Data Delay	(Notes 3, 8)	0	100	0	100	0	100	0	150	0	150	ns
t_{MR}	Master Reset Pulse Width		5		5		10		25		25		ns
t_{RA}	Address Hold Time from RD/$\overline{\text{RD}}$	(Note 1)	20		20		20		10		10		ns
t_{RC}	Read Cycle Delay		125		175		500		1735		1735		ns
t_{RCS}	Chip Select Hold Time from RD/$\overline{\text{RD}}$	(Note 1)	20		20		20		50		50		ns
t_{RD}	RD/$\overline{\text{RD}}$ Strobe Width		125		125		175		175		350		ns
t_{RDA}	RD/$\overline{\text{RD}}$ Strobe Delay from $\overline{\text{ADS}}$		NA		NA		NA		0		0		ns
t_{RDD}	RD/$\overline{\text{RD}}$ Driver Enable/Disable	(Notes 3, 8)		60		60		75		150		250	ns
t_{RVD}	Delay from RD/$\overline{\text{RD}}$ to Data	(Note 8)		125		125		175		250		300	ns
t_{WA}	Address Hold Time from WR/$\overline{\text{WR}}$	(Note 1)	20		20		20		50		50		ns
t_{WC}	Write Cycle Delay		150		200		500		1785		1785		ns

Note 1: Applicable only when $\overline{\text{ADS}}$ is tied low.

Note 3: Charge and discharge time is determined by V_{OL}, V_{OH} and the external timing.

Note 8: Loading of 100 pF.

NA = Not Applicable.

AC Electrical Characteristics T_A = 0°C to +70°C, V_{CC} = 5V ±5% (Continued)

Symbol	Parameter	Conditions	NS16550A		NS16450 NS16C450		INS8250A INS82C50A		INS8250		INS8250-B		Units
			Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	
t_{WCS}	Chip Select Hold Time from WR/$\overline{\text{WR}}$	(Note 1)	20		20		20		50		50		ns
t_{WDA}	WR/$\overline{\text{WR}}$ Delay from Address		NA		NA		NA		50		50		ns
t_{WR}	WR/$\overline{\text{WR}}$ Strobe Width		100		100		175		175		350		ns
t_{XH}	Duration of Clock High Pulse	(Note 4)	55		140		140		140		140		ns
t_{XL}	Duration of Clock Low Pulse	(Note 4)	55		140		140		140		140		ns
RC	Read Cycle = $t_{AR} + t_{DIW} + t_{RC}$		280		360		755		2000		2205		ns
WC	Write Cycle = $t_{DDA} + t_{DOW} + t_{WC}$		280		360		755		2100		2305		ns
BAUD GENERATOR													
N	Baud Divisor		1	$2^{16}-1$	1	$2^{16}-1$	1	$2^{16}-1$	1	$2^{16}-1$	1	$2^{16}-1$	ns
t_{BHD}	Baud Output Positive Edge Delay	(Note 8)		175		175		250		250		250	ns
t_{BLD}	Baud Output Negative Edge Delay	(Note 8)		175		175		250		250		250	ns
t_{HW}	Baud Output Up Time	(Note 5)	75		250		250		330		330		ns
t_{LW}	Baud Output Down Time	(Note 6)	100		425		425		425		425		ns
RECEIVER (Note 2)													
t_{RINT}	Delay from RD/$\overline{\text{RD}}$ (RD RBR/RDLSR) to Reset Interrupt	(Note 8)		1000		1000		1000		1000		1000	ns
t_{SCD}	Delay from RCLK to Sample Time			2000		2000		2000		2000		2000	ns
t_{SINT}	Delay from Stop to Set Interrupt			1 RCLK		1 RCLK		1 RCLK		2000		2000	ns

Note 1: Applicable only when $\overline{\text{ADS}}$ is tied low.

Note 2: For the NS16550A in the FIFO Mode (FCR0 = 1) the trigger level and timeout interrupts, the receiver data available indication, the active RXRDY indication and the overrun error indication will be delayed 3 RCLKs. Status indicators (PE, FE, BI) will be delayed 3 RCLKs after the first byte has been received. For subsequently received bytes these indicators will be updated immediately after RDRBR goes inactive.

Note 4: The maximum external clock for the NS16550A is 8 MHz, NS16450 and INS8250A is 3.1 MHz and INS8250 and INS8250-B is 3.1 MHz. 100 pF load.

Note 5: The maximum external clock for the NS16550A is 8 MHz, NS16450 and INS8250A is 3.1 MHz and INS8250 and INS8250-B is 3.1 MHz. 100 pF load. This parameter is tested on the NS16550A and guaranteed by design on all other parts.

Note 6: The maximum external clock for the NS16550A is 8 MHz, NS16450 and INS8250A is 2.1 MHz and INS8250 and INS8250-B is 3.1 MHz. 100 pF load. This parameter is tested on the NS16550A and guaranteed by design on all other parts.

Note 8: Loading of 100 pF.

NA = Not Applicable.

AC Electrical Characteristics $T_A = 0°C$ to $+70°C$, $V_{CC} = 5V \pm 5\%$ (Continued)

Symbol	Parameter	Conditions	NS16550A		NS16450 NS16C450		INS8250A INS82C50A		INS8250		INS8250-B		Units
			Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	
TRANSMITTER													
t_{HR}	Delay from WR/$\overline{WR}$ (WR THR) to Reset Interrupt	(Note 8)		175		175		1000		1000		1000	ns
t_{IR}	Delay from RD/$\overline{RD}$ (RD IIR) to Reset Interrupt (THRE)	(Note 8)		250		250		1000		1000		1000	ns
t_{IRS}	Delay from Initial INTR Reset to Transmit Start	(Note 10)	8	24	24	40	24	40		16		16	Baudout Cycles
t_{SI}	Delay from Initial Write to Interrupt	(Notes 7, 9)	16	24	16	24	16	24		50		50	Baudout Cycles
t_{SS}	Delay from Stop to Next Start			NA		NA		NA		1000		1000	ns
t_{STI}	Delay from Stop to Interrupt (THRE)	(Note 7)	8	8	8	8	8	8		8		8	Baudout Cycles
t_{SXA}	Delay from Start to TXRDY Active	(Note 8)		8		NA		NA		NA		NA	Baudout Cycles
t_{WXI}	Delay from Write to TXRDY Inactive	(Note 8)		195		NA		NA		NA		NA	ns
MODEM CONTROL													
t_{MDO}	Delay from WR/$\overline{WR}$ (WR MCR) to Output	(Note 8)		200		200		1000		1000		1000	ns
t_{RIM}	Delay to Reset Interrupt from RD/$\overline{RD}$ (RD MSR)	(Note 8)		250		250		1000		1000		1000	ns
t_{SIM}	Delay to Set Interrupt from MODEM Input	(Note 8)		250		250		1000		1000		1000	ns

Note 7: This delay will be lengthened by 1 character time, minus the last stop bit time if the transmitter interrupt delay circuit is active.

Note 8: Loading of 100 pF.

Note 9: For both the NS16C450 and INS82C50A the value of t_{SI} will range from 16 to 48 baudout cycles.

Note 10: For both the NS16C450 and the INS82C50A the value of t_{IRS} will range from 24 to 40 baudout cycles.

NA = Not Applicable.

Comparison Study NSC800 vs. 8085/80C85 Z80®/Z80 CMOS

Introduction

The NSC800 is an 8-bit parallel processor with a Z80 compatible instruction set manufactured using National's micro-CMOS process. This process combines the speed of silicon gate NMOS with the low power inherent to CMOS.

The NSC800 has a 16-bit address bus which consists of the upper eight address bits (A8–A15) and the lower eight address bits (AD0–AD7). Address bits A0–A7 are time multiplexed on the 8-bit bidirectional address/data bus (AD0–AD7).

There are several advantages to using a multiplexed address/data bus. Multiplexing frees pins on the CPU and peripheral packages for other purposes, such as status outputs, DMA control lines, and multiple interrupts. This can reduce system component count. Fewer bus signal lines are required for device interconnections in most applications (16 lines for multiplexed bus systems vs. 24 lines for non-multiplexed systems). This reduces PC board complexity.

Peripherals of the NSC800 Family include:

NSC810A RAM I/O Timer
NSC831 I/O
NSC858 UART

In addition to the above parts, a complete family of low power speed compatible logic and interface parts is also available.

NSC800 vs. 8085

In terms of bus structure, the NSC800 is similar to the 8085. Both processors utilize a multiplexed bus and timing relationships are approximately the same. The 8085 does not guarantee that output data on AD0–AD7 are valid on both the leading and trailing edges of $\overline{WR}$. For the NSC800, data are valid on both the leading and trailing edges of $\overline{WR}$.

Both the NSC800 and the 8085 use ALE, S0, S1, and IO/$\overline{M}$ to indicate status. The lower eight address bits are guaranteed to be valid on the data bus at the trailing edge (high to low transition) of ALE (Address Latch Enable). This signal is used by the external system components to separate the address and data buses. When the only components utilized in the system are members of the NSC800 family (which contain on-chip demultiplexers), ALE needs only to be connected to the enable inputs. If non-NSC800 family components are used, ALE can be used to enable an 8-bit latch to perform the function of bus separation.

Decoding status bits S0 and S1, in conjunction with IO/$\overline{M}$, notifies the external system of the type of the ensuing M cycle. TABLE I shows a truth table of the encoded information. During a halt status the NSC800 will continue to refresh dynamic RAM.

TABLE I.
Machine Cycle Status - NSC800 and 8085

S0	S1	IO/M	Status
1	0	0	Memory Write
0	1	0	Memory Read
1	0	1	I/O Write
0	1	1	I/O Read
1	1	0	Opcode Fetch
0	1	0	Bus Idle*
0	0	0	Halt

*ALE not suppressed during Bus Idle

Direct Memory Access (DMA) control signals $\overline{BREQ}$ and $\overline{BACK}$ of the NSC800 perform the same functions as HOLD and HLDA on the 8085. The NSC800 allows simple wire ORing by using active low states for the DMA control signals. An active low on the $\overline{BREQ}$ (Bus Request) line, tested during the last T state of the current M cycle, initiates a DMA condition. The NSC800 will then respond with an active low $\overline{BACK}$ (Bus Acknowledge) signal causing the address, data and control buses (TRI-STATE® circuits) to go to the high impedance state, and notifies the interrupting device that the system bus is available for use. There is a difference in the timing relationship between these functions for the two processors. The 8085 responds with HLDA, one-half T state after it recognizes HOLD. The NSC800 responds with $\overline{BACK}$, one T state after it recognizes $\overline{BREQ}$.

During Input/Output cycles for peripherals, the NSC800 automatically inserts one wait state. This reduces the external hardware required for slow peripherals. The 8085 does not insert its own wait state during these I/O cycles. When they are needed, the 8085 user must design his system to contain the additional hardware required to do the wait state insertion. When more than one wait state is required, additional wait states can be added to the I/O cycles in a similar way on both the NSC800 and the 8085. On the NSC800, this is accomplished by bringing the $\overline{WAIT}$ control signal active low during T2 of an I/O or memory cycle. The 8085 is controlled in the same way through the use of the READY line.

The NSC800 instruction set is Z80 compatible and more powerful than the 8085's. The NSC800 does not support the RIM and SIM instructions of the 8085 (RIM and SIM can be emulated with I/O instructions), but has an improved instruction set for enhanced system performance. The NSC800 has two functions, $\overline{RFSH}$ and $\overline{PS}$, instead of the two serial I/O lines SOD and SID. $\overline{RFSH}$ (Refresh) is a status signal which indicates that an eight bit refresh address is present on the address/data bus (AD0–AD7). The refresh address occurs during T3 of each M1 (opcode fetch) cycle. The internal refresh counter is incremented after each instruction cycle. This counter output can be employed by the user's dynamic RAM refresh circuits. The $\overline{\text{PS}}$ (Power Save) control input, when active, causes the CPU to stop all internal clocks at the end of the current instruction, which reduces power consumption. The on-chip oscillator and CLK remain active for any required external timing. The NSC800 leaves all buses unchanged during this time, which has the effect of reducing power consumption on other CMOS parts in the system since the buses are not changing states. All internal registers and status conditions are maintained, and when $\overline{\text{PS}}$ subsequently goes high, the opcode fetch cycle begins in a normal fashion.

TABLE II indicates the major differences between the NSC800 and the 8085 presented in tabular form for quick reference.

TABLE II.
NSC800 vs. 8085/80C85 Comparison

Item	NSC800	8085	80C85
Power Consumption	50 mW @ 5V	850 mW @ 5V	50 mW @ 5V
Bus Drive Capacity	1 std. TTL (100 pF)	1 std. TTL (100 pF)	1 std. TTL (150 pF)
Dynamic RAM Refresh Counter	Yes, 8-bit	No	No
Automatic WAIT State on I/O	Yes	No	No
Number of instruction types	158	80	80
Number of Programmer Accessible Registers	22	10	10
Block I/O and Search	Yes	No	No

NSC800 vs. Z80/Z80 CMOS

The NSC800 contains the same complement of internal registers as the Z80 and maintains instruction set and opcode compatibility.

Machine cycle timing for the standard speed version of the NSC800 compares directly with the Z80. Although the software execution speeds are comparable, the NSC800 offers architectural advantages.

The bus structures of the NSC800 and the Z80 are quite different. The NSC800 uses a multiplexed address/data bus. The Z80 has separate address and data buses. As stated earlier, the separate bus structure requires additional signal lines for interconnection and gives up some package pins which could be used for other purposes.

The main differences between the NSC800 and the Z80, in addition to the bus structures, are the refresh counter, on-chip clock generation, and the interrupt capability.

1. The NSC800 contains an 8-bit refresh counter as opposed to a 7-bit refresh counter in the Z80. (This enables refresh of a 64K dynamic RAM system memory). The refresh timing of the NSC800 is functionally identical to that of the Z80.
2. The on-chip clock generation reduces the system component count. In place of an external clock generator chip, the NSC800 needs only a crystal or RC circuit to produce the system clock.
3. The NSC800 provides three interrupts that are not available on the Z80: $\overline{\text{RSTA}}$, $\overline{\text{RSTB}}$, $\overline{\text{RSTC}}$. This gives the NSC800 five levels of vectored, prioritized interrupts with no external logic. The general purpose interrupt ($\overline{\text{INTR}}$) and Non-maskable Interrupt ($\overline{\text{NMI}}$) are identical to the Z80. $\overline{\text{INTR}}$ has the same three modes of operation in both processors: Modes 0, 1, and 2. Upon initialization, the NSC800 is in mode 0 to maintain 8080 code compatibility. $\overline{\text{NMI}}$, when active, causes a restart to location X'66 as is the case with the Z80. Being a non-maskable interrupt, $\overline{\text{NMI}}$ cannot be disabled. The additional interrupts $\overline{\text{RSTA}}$, $\overline{\text{RSTB}}$, and $\overline{\text{RSTC}}$ cause restarts to locations X'3C, X'34, and X'2C respectively. The priority levels of the five interrupts are: $\overline{\text{NMI}}$ (highest), $\overline{\text{RSTA}}$, $\overline{\text{RSTB}}$, $\overline{\text{RSTC}}$, and $\overline{\text{INTR}}$ (lowest). For the NSC800, Interrupt acknowledge ($\overline{\text{INTA}}$) is provided on a dedicated output pin and need not be decoded externally, as is the case with the Z80. With the status outputs (S0, S1, IO/$\overline{\text{M}}$), early read/write information is obtainable. This is impossible to derive from the Z80.

Refer to TABLE III for comparison of the major differences between the NSC800 and the Z80.

TABLE III.
NSC800 vs. Z80/Z80 CMOS Comparison

Item	NSC800	Z80	Z80 CMOS
Power Consumption	50 mW @ 5V	750 mW @ 5V	75 mW @ 5V
Instruction Execution (Minimum)	1 μs	1 μs	1 μs
On-Chip Clock Generator	Yes	No	No
Number of On-Chip Vectored Interrupts	5	2	2
Early Read/Write Status	Yes	No	No
Dynamic RAM Refresh Counter	Yes, 8-bit	Yes, 7-bit	Yes, 7-bit

NSC800 Family Devices (microCMOS)

MM82PC08 8-Bit Bidirectional Transceiver

MM82PC12 Input/Output Port

Note: The above devices are pin for pin and function compatible with the standard TTL, CMOS or NMOS versions currently available.

SUMMARY

National's NSC800 has a Z80 compatible instruction set, which is more powerful than the 8085. NSC800 external hardware requirements are less because of on-chip automatic wait state insertion, clock generation and five levels of vectored prioritized interrupts.

The 8085 and the NSC800 have similar bus structures, and timing. The key advantages of the NSC800 over the 8085 are the larger instruction set, more registers accessible to programmers, low power consumption, and a dynamic RAM refresh counter.

The main advantages of the NSC800 compared to the Z80 are the multiplexed address/data bus, an 8-bit refresh counter for dynamic RAMs, on-chip clock generation, and five interrupts. The speed of the NSC800 and Z80 is the same but, the NSC800 has very low power consumption.

Software Comparison NSC800 vs. 8085, Z80®

Introduction

The NSC800 is an 8-bit parallel microprocessor fabricated using National's microCMOS process. This process allows fabrication of a microprocessor family that has the performance of silicon gate NMOS along with the low power inherent to CMOS. The NSC800 instruction set is a superset of the 8080's instruction set. It comprises over 900 operation codes falling into 158 instruction types. The instruction categories are:

- Load and Exchange
- Arithmetic and Logic
- Rotate and Shift
- Jump and Call
- Input/Output
- Bit manipulation (set, test, reset)
- Block Transfer and Search
- CPU control

The load instructions allow the movement of data into and out of the CPU, between internal registers, plus the capability to load immediate data into internal registers. The exchange instructions allow swapping of data between two registers.

The arithmetic and logic instructions operate on the data in the accumulator (primary working register) and in the other registers. Status flags are set or reset depending on the result of the particular operation executed. This group includes 8-bit and 16-bit operations.

The rotate and shift instructions allow any register or memory location to be rotated or shifted, left or right, with or without carry. These can be either an arithmetic or logic type.

The jump and call group includes several different types: one byte calls, two byte relative jumps, conditional branching, and three byte calls and jumps, which can reach any location in memory. Calls push the current contents of the Program Counter onto the stack before branching to the new program address to facilitate subroutine execution.

Input/Output instructions allow communications between the NSC800 and external peripheral devices. There are 255 (location X'BB is used for an interrupt mask) unique peripheral I/O locations available to the NSC800. I/O instructions can move data between any memory location or internal register and any I/O location. There are also block I/O instructions which allow moving data blocks of up to 256 bytes directly from memory to any peripheral location or from any peripheral location to a block of memory.

Bit manipulation instructions can set, test or reset any bit in the accumulator, any general purpose register or any memory location.

The block transfer instructions allow a single instruction to move any size block of memory to any other location in memory. Through the use of the block search instructions, any size block of memory can be searched for a particular byte of data.

Finally, the CPU control group allows user control over the various modes of CPU operation, such as enabling and disabling interrupts or setting modes of interrupt response.

The following sections will compare the instruction set of the NSC800 with those of the 8085 and the Z80.

NSC800 vs. 8085

The 8085 instruction set consists of 246 op codes falling into 80 instruction types. With the exception of RIM and SIM, the NSC800 is instruction and op code compatible with the 8085. The RIM and SIM instructions are not supported because the NSC800 does not have the SID and SOD serial I/O lines. The interrupt mask on the NSC800 is accessible by writing the mask word to I/O location X'BB. The bit positions for the interrupt enables are shown below:

Location X'BB Bit Assignments

Bit	**Interrupt Enable for**
7	N/A
6	N/A
5	N/A
4	N/A
3	$\overline{\text{RSTA}}$
2	$\overline{\text{RSTB}}$
1	$\overline{\text{RSTC}}$
0	$\overline{\text{INTR}}$

N/A = not used: a don't care bit.

As an example, to enable interrupts on the $\overline{\text{RSTA}}$ input, a logic '1' is written into bit 3 of I/O location X'BB. If the master interrupt enable has been set by executing the Enable Interrupt (EI) instruction, interrupts will now be accepted on $\overline{\text{RSTA}}$ only.

Other than the method of enabling and disabling individual interrupts and the RIM and SIM instructions themselves, the NSC800 instruction set is a superset of the 8085's instruction set.

The following benchmark demonstrates the code reduction and throughtput improvement obtained by using one of the special NSC800 instructions over the same function implemented with the limited 8085 instruction set. The function is to move a 512-byte block of data from one section of memory to another.

8085

Bytes		Mnemonics		Cycles
3		LXI	H,SOURCE	10
3		LXI	D,DEST	10
3		LXI	B,COUNT	10
1	LOOP:	MOV	A,M	7
1		STAX	D	7
1		INX	H	6
1		INX	D	6
1		DCX	B	6
1		MOV	A,C	4
1		ORA	B	4
3		JNZ	LOOP	10
Total: 19				Total: 80

NSC800

Bytes	Mnemonics		Cycles
3	LD	HL,SOURCE	10
3	LD	DE,DEST	10
3	LD	BC,COUNT	10
2	LDIR		21
Total: 11			Total: 51

The use of the LDIR instruction of the NSC800 results in a 47.5% increase in throughput and a 42% decrease in the number of bytes required to implement the function when compared with the 8085 implementation. The time required to make the move is approximately 2.69 ms for the NSC800 and approximately 5.12 ms for the 8085. Note that even though the 8085 runs at a faster cycle time (200 ns vs. 250 ns), the improved instruction set of the NSC800 produces an increase in system performance.

The NSC800 includes all 8085 flags plus some additional flags. The flag formats for the NSC800 and 8085 are:

NSC800 Flags (Z80 Flags)

7	6	5	4	3	2	1	0
S	Z	X	H	X	P/V	N	C

8085 Flags

7	6	5	4	3	2	1	0
S	Z	X	AC	X	P	X	CY

The differences between the flag registers on the NSC800 and the 8085 are identified below:

1. Bit position D1 (additional on the NSC800) contains an add/subtract flag that is used internally for proper operation of BCD instructions.

2. In the NSC800, the P/V flag will not match the 8085's P flag after an 8-bit arithmetic operation, since it acts as an overflow bit for the NSC800, but acts as a parity bit for these operations in the 8085.

3. Bit position D2 (changed for the NSC800) is a dual purpose flag; it indicates the parity of the result in the accumulator when logical operations are performed and also represents overflow when signed two's complement arithmetic operations are performed. An overflow occurs when the result of a two's complement operation within the accumulator is out of range.

4. For general Compare operations, the NSC800 uses the P/V flag as an overflow bit, while the 8085 uses the P flag for parity.

5. The H flag (bit position D4) on the NSC800 is functionally the same as the auxiliary carry on the 8085.

6. For Double Precision Addition, the NSC800 leaves the H flag undefined, while the 8085 does not affect the AC flag for this operation (DAD).

7. For Rotate operations, the NSC800 resets the H flag, while the 8085 leaves the AC flag unaffected for these operations.

8. When Complementing the Accumulator, the NSC800 sets the H flag (H = 1), while the 8085 leaves the AC flag unaffected.

9. When Complementing Carry, the NSC800 leaves the H flag undefined, while the 8085 leaves the AC flag unaffected.

10. When Setting the Carry, the NSC800 clears the H flag (H = 0), while the 8085 leaves the AC flag unaffected.

NSC800 vs. Z80

The instruction set and op codes of the NSC800 are identical to those of the Z80. Software written for the Z80 will run on the NSC800 without change, unless I/O location X'BB is used. Another location should be assigned since location X'BB is an on-chip write-only register used for the interrupt mask. Since the NSC800 executes code at the same cycle time as the Z80, any software timing loops will also remain the same, and no change is necessary. The NSC800 expanded interrupt capability is transparent to the user unless specifically evoked by the user software.

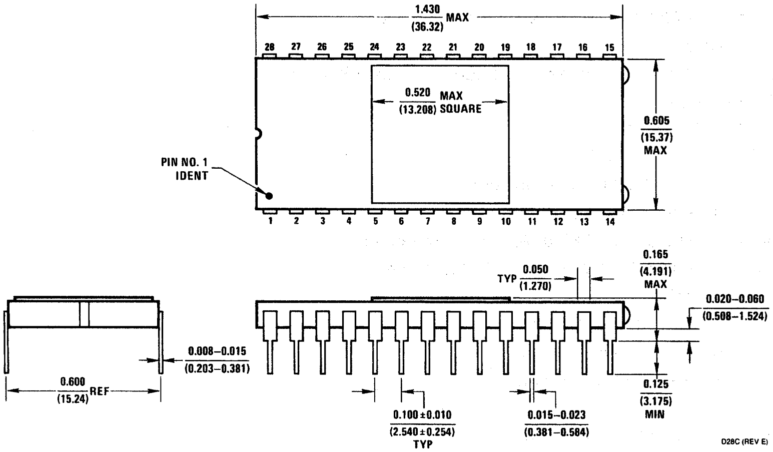
The NSC800 has 8-bit refresh rather than the 7-bit refresh scheme of the Z80. Therefore, the state of the 8th bit will be indeterminate since it is part of the R Register and so included in refresh operations.

The status flags on the NSC800 are identical to those on the Z80. There is no difference between the positions of the individual bits in the flag register, nor in the manner in which the flags are set or reset due to an arithmetic or logical operation. Testing of the flags is also the same.

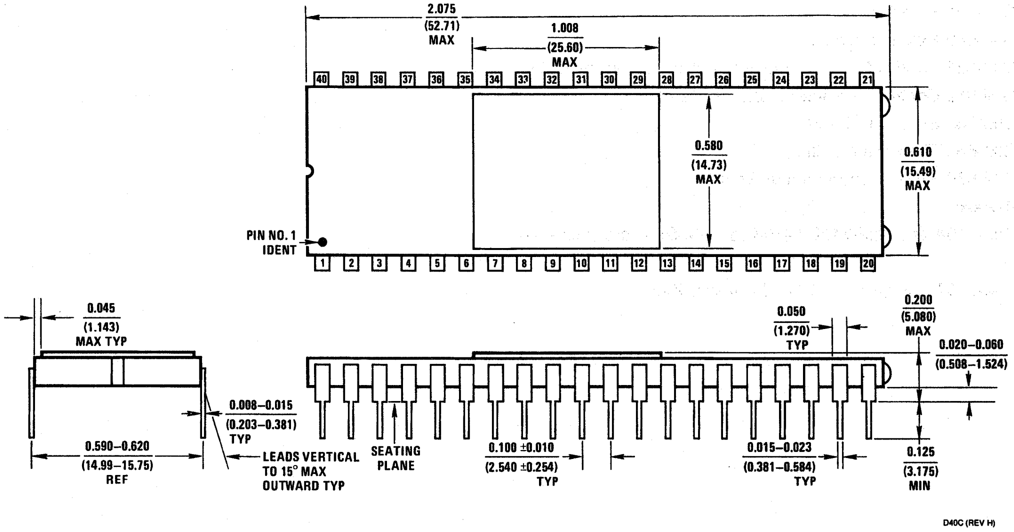
CMOS Industrial Microcomputers Products

National Semiconductor Corporation

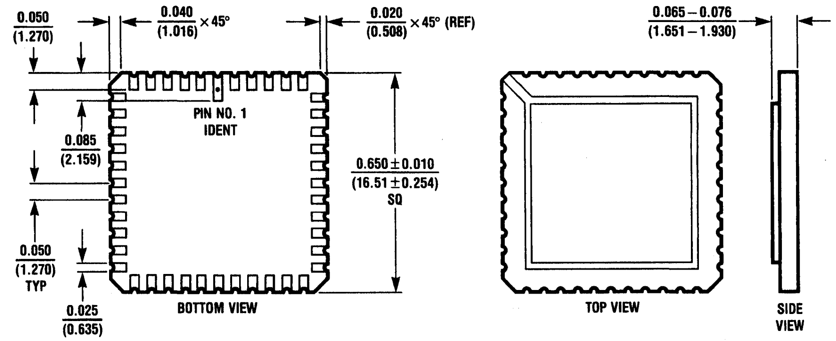
All dimensions are in inches (millimeters)



NS Package D28C

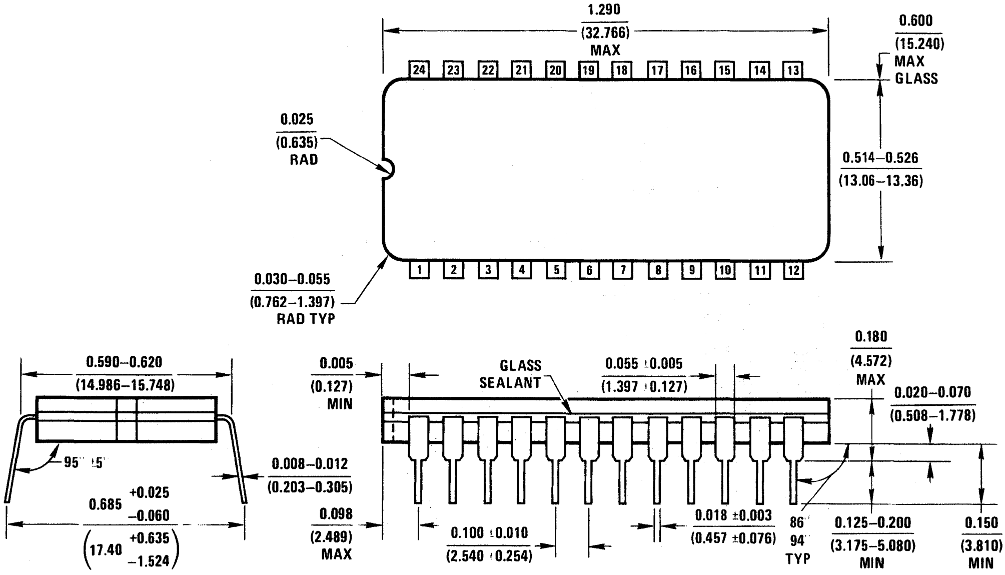


NS Package D40C



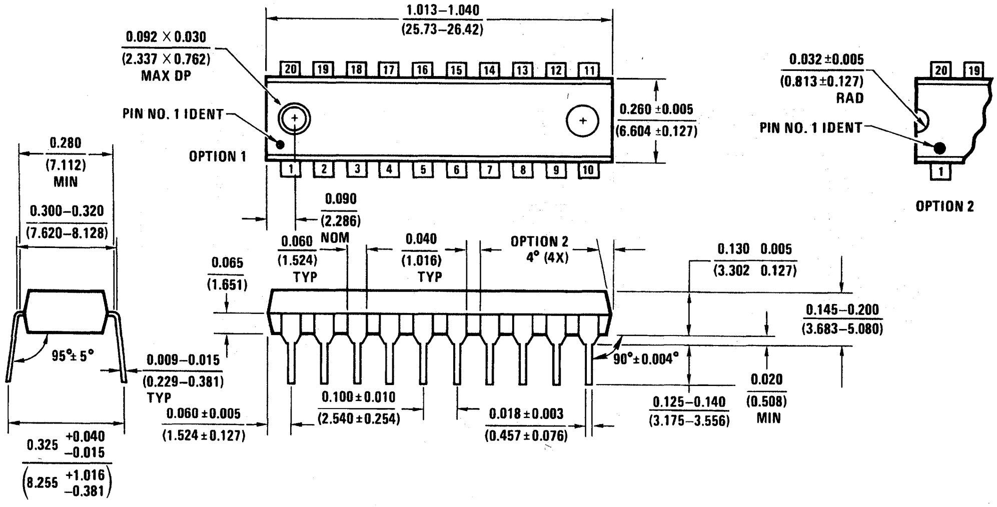
E44A (REV C)

NS Package E44A

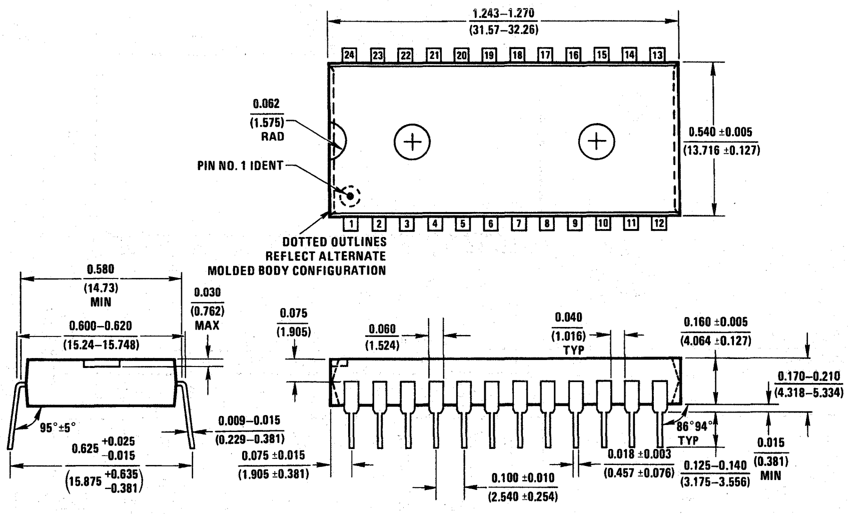


J24A (REV H)

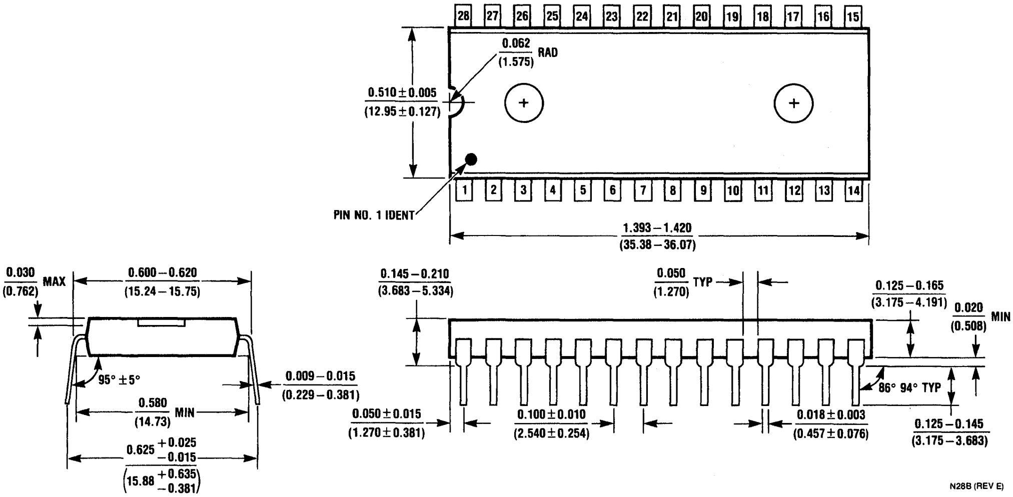
NS Package J24A



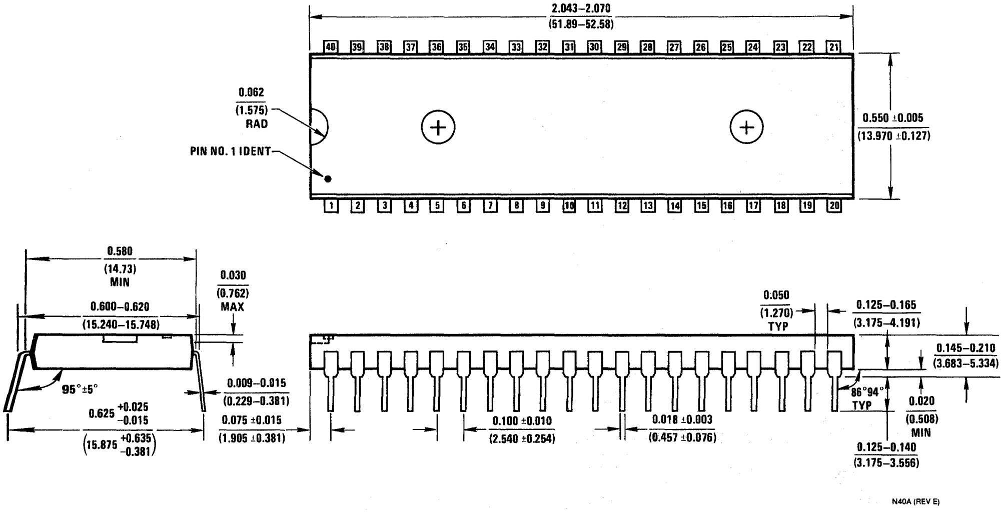
NS Package N20A



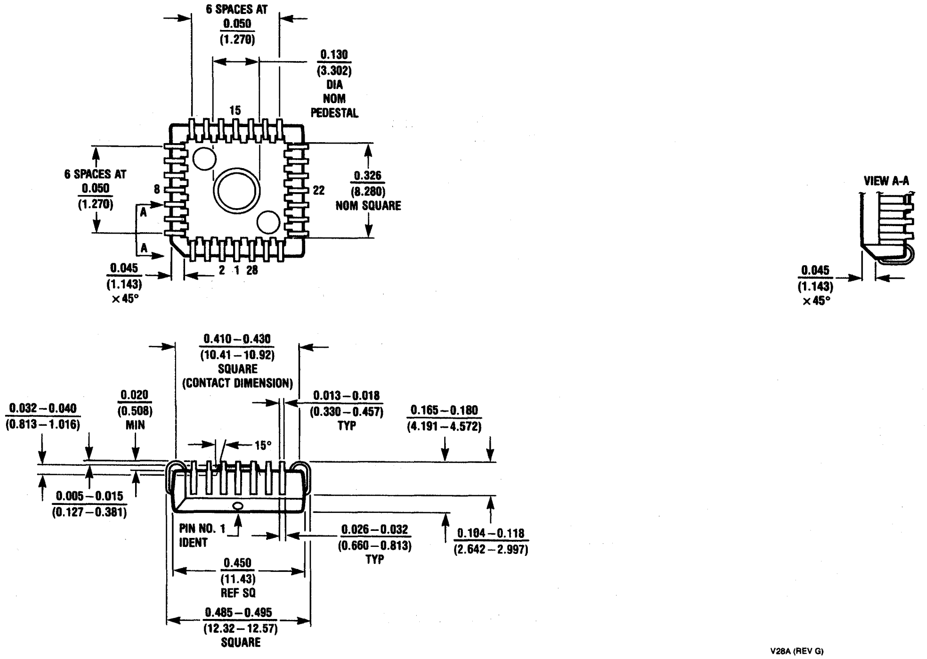
NS Package N24A



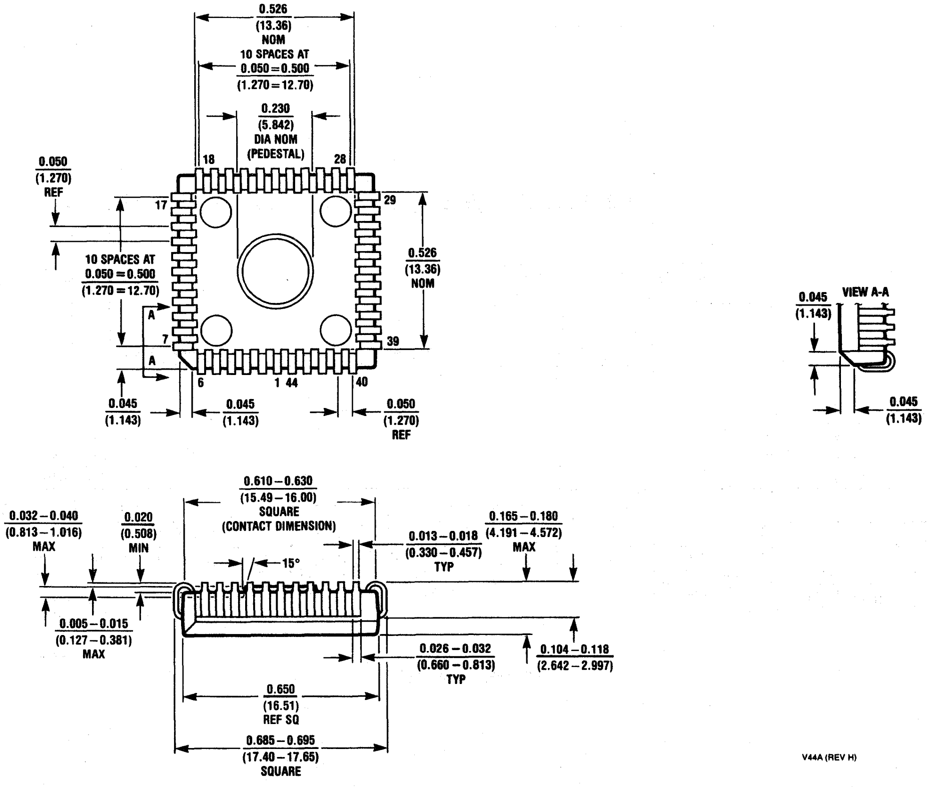
NS Package N28B



NS Package N40A



NS Package V28A



NS Package V44A

NOTES

NOTES

NOTES

National
Semiconductor
Corporation

Bookshelf of Technical Support Information

National Semiconductor Corporation recognizes the need to keep you informed about the availability of current technical literature.

This bookshelf is a compilation of books that are currently available. The listing that follows shows the publication year and section contents for each book.

Please contact your local National sales office for possible complimentary copies. A listing of sales offices follows this bookshelf.

We are interested in your comments on our technical literature and your suggestions for improvement.

Please send them to:

Technical Communications Dept. M/S 23-200
2900 Semiconductor Drive
P.O. Box 58090
Santa Clara, CA 95052-8090

For a recorded update of this listing plus ordering information for these books from National's Literature Distribution operation, please call (408) 749-7378.

ALS/AS LOGIC DATABOOK—1987

Introduction to Bipolar Logic • Advanced Low Power Schottky • Advanced Schottky

ASIC DESIGN MANUAL/GATE ARRAYS & STANDARD CELLS—1987

SSI/MSI Functions • Peripheral Functions • LSI/VLSI Functions • Design Guidelines • Packaging

DATA CONVERSION/ACQUISITION DATABOOK—1984

Selection Guides • Active Filters • Amplifiers • Analog Switches • Analog-to-Digital Converters
Analog-to-Digital Display (DVM) • Digital-to-Analog Converters • Sample and Hold • Sensors/Transducers
Successive Approximation Registers/Comparators • Voltage References

HYBRID PRODUCTS DATABOOK—1982

Operational Amplifiers • Buffers • Instrumentation Amplifiers • Sample & Hold Amplifiers • Comparators
Non-Linear Functions • Precision Voltage Regulators and References • Analog Switches
MOS Clock Drivers • Digital Drivers • A-D Converters • D-A Converters • Fiber-Optic Products
Active Filters & Telecommunication Products • Precision Networks • 883/RETS

INTERFACE DATABOOK—1986

Transmission Line Drivers/Receivers • Bus Transceivers • Peripheral/Power Drivers • Display Controllers/Drivers
Memory Support • Microprocessor Support • Level Translators/Buffers • Frequency Synthesis

INTERFACE/BIPOLAR LSI/BIPOLAR MEMORY/PROGRAMMABLE LOGIC DATABOOK—1983

Transmission Line Drivers/Receivers • Bus Transceivers • Peripheral/Power Drivers
Level Translators/Buffers • Display Controllers/Drivers • Memory Support • Dynamic Memory Support
Microprocessor Support • Data Communications Support • Disk Support • Frequency Synthesis
Interface Appendices • Bipolar PROMs • Bipolar and ECL RAMs • 2900 Family/Bipolar Microprocessor
Programmable Logic

INTUITIVE IC CMOS EVOLUTION—1984

Thomas M. Frederiksen's new book targets some of the most significant transitions in semiconductor technology since the change from germanium to silicon. *Intuitive IC CMOS Evolution* highlights the transition in the reduction in defect densities and the development of new circuit topologies. The author's latest book is a vital aid to engineers, and industry observers who need to stay abreast of the semiconductor industry.

INTUITIVE IC OP AMPS—1984

Thomas M. Frederiksen's new book, *Intuitive IC Op Amps,* explores the many uses and applications of different IC op amps. Frederiksen's detailed book differs from others in the way he focuses on the intuitive groundwork in the basic functioning concepts of the op amp. Mr. Frederiksen's latest book is a vital aid to engineers, designers, and industry observers who need to stay abreast of the computer industry.

LINEAR APPLICATIONS HANDBOOK—1986

The purpose of this handbook is to provide a fully indexed and cross-referenced collection of linear integrated circuit applications using both monolithic and hybrid circuits from National Semiconductor.

Individual application notes are normally written to explain the operation and use of one particular device or to detail various methods of accomplishing a given function. The organization of this handbook takes advantage of this innate coherence by keeping each application note intact, arranging them in numerical order, and providing a detailed Subject Index.

LINEAR 1 DATABOOK—1988

Voltage Regulators • Operational Amplifiers • Buffers • Voltage Comparators • Instrumentation Amplifiers • Surface Mount

LINEAR 2 DATABOOK—1988

Active Filters • Analog Switches/Multiplexers • Analog-to-Digital • Digital-to-Analog • Sample and Hold
Sensors • Voltage References • Surface Mount

LINEAR 3 DATABOOK—1988

Audio Circuits • Radio Circuits • Video Circuits • Motion Control • Special Functions • Surface Mount

LINEAR SUPPLEMENT DATABOOK—1984

Amplifiers • Comparators • Voltage Regulators • Voltage References • Converters • Analog Switches
Sample and Hold • Sensors • Filters • Building Blocks • Motor Controllers • Consumer Circuits
Telecommunications Circuits • Speech • Special Analog Functions

LOGIC DATABOOK VOLUME I—1984

CMOS AC Switching Test Circuits and Timing Waveforms • CMOS Application Notes • MM54HC/MM74HC
MM54HCT/MM74HCT • CD4XXX • MM54CXXX/MM74CXXX • LSI/VLSI

LS/S/TTL DATABOOK—1987

Introduction to Bipolar Logic • Low Power Schottky • Schottky • TTL • Low Power

MASS STORAGE HANDBOOK—1986

Disk Interface Design Guide and User Manual • Winchester Disk Support • Winchester Disk Data Controller
Floppy Disk Support • Drive Interface Support Circuits

MEMORY SUPPORT HANDBOOK—1986

Dynamic Memory Control • Error Checking and Correction • Microprocessor Interface and Applications
Memory Drivers and Support

NON-VOLATILE MEMORY DATABOOK—1987

CMOS EPROMs • EEPROMs • Bipolar PROMs

SERIES 32000 DATABOOK—1986

Introduction • CPU-Central Processing Unit • Slave Processors • Peripherals • Data Communications and LAN's
Disk Control and Interface • DRAM Interface • Development Tools • Software Support • Application Notes

RANDOM ACCESS MEMORY DATABOOK—1987

Static RAMs • TTL RAMs • TTL FIFOs • ECL RAMs

RELIABILITY HANDBOOK—1986

Reliability and the Die • Internal Construction • Finished Package • MIL-STD-883 • MIL-M-38510
The Specification Development Process • Reliability and the Hybrid Device • VLSI/VHSIC Devices
Radiation Environment • Electrostatic Discharge • Discrete Device • Standardization
Quality Assurance and Reliability Engineering • Reliability and Documentation • Commercial Grade Device
European Reliability Programs • Reliability and the Cost of Semiconductor Ownership
Reliability Testing at National Semiconductor • The Total Military/Aerospace Standardization Program
883B/RETS™ Products • MILS/RETS™ Products • 883/RETS™ Hybrids • MIL-M-38510 Class B Products
Radiation Hardened Technology • Wafer Fabrication • Semiconductor Assembly and Packaging
Semiconductor Packages • Glossary of Terms • Key Government Agencies • AN/ Numbers and Acronyms
Bibliography • MIL-M-38510 and DESC Drawing Cross Listing

TELECOMMUNICATIONS—1987

Line Card Components • Integrated Services Digital Network Components • Modems
Analog Telephone Components • Application Notes

THE SWITCHED-CAPACITOR FILTER HANDBOOK—1985

Introduction to Filters • National's Switched-Capacitor Filters • Designing with Switched-Capacitor Filters
Application Circuits • Filter Design Program • Nomographs and Tables

TRANSISTOR DATABOOK—1982

NPN Transistors • PNP Transistors • Junction Field Effect Transistors • Selection Guides • Pro Electron Series
Consumer Series • NA/NB/NR Series • Process Characteristics Double-Diffused Epitaxial Transistors
Process Characteristics Power Transistors • Process Characteristics JFETs • JFET Applications Notes

VOLTAGE REGULATOR HANDBOOK—1982

Product Selection Procedures • Heat Flow & Thermal Resistance • Selection of Commercial Heat Sink
Custom Heat Sink Design • Applications Circuits and Descriptive Information • Power Supply Design
Data Sheets

48-SERIES MICROPROCESSOR HANDBOOK—1980

The 48-Series Microcomputers • The 48-Series Single-Chip System • The 48-Series Instruction Set
Expanding the 48-Series Microcomputers • Applications for the 48-Series • Development Support
Analog I/O Components • Communications Components • Digital I/O Components • Memory Components
Peripheral Control Components

National
Semiconductor
Corporation

National Semiconductor Corporation
2900 Semiconductor Drive
P.O. Box 58090
Santa Clara, CA 95052-8090
Tel: (408) 721-5000
TWX: (910) 339-9240

SALES OFFICES (Continued)

INTERNATIONAL OFFICES

Electronica NSC de Mexico SA
Juventino Rosas No. 118-2
Col Guadalupe Inn
Mexico, 01020 D.F. Mexico
Tel: (905) 524-9402

National Semicondutores Do Brasil Ltda.
Av. Brig. Faria Lima, 830
8 Andar
01452 Sao Paulo, SP, Brasil
Tel: (55/11) 212-5066
Telex: 391-1131931 NSBR BR

National Semiconductor GmbH
Industriestrasse 10
D-8080 Furstenfeldbruck
West Germany
Tel: 49-08141-103-0
Telex: 527 649

National Semiconductor (UK) Ltd.
301 Harpur Centre
Horne Lane
Bedford MK40 ITR
United Kingdom
Tel: (02 34) 27 00 27
Telex: 826 209

National Semiconductor Benelux
Vorstlaan 100
B-1170 Brussels
Belgium
Tel: (02) 6725360
Telex: 61007

National Semiconductor (UK) Ltd.
1, Bianco Lunos Alle
DK-1868 Fredriksberg C
Denmark
Tel: (01) 213211
Telex: 15179

National Semiconductor
Expansion 10000
28, rue de la Redoute
F-92260 Fontenay-aux-Roses
France
Tel: (01) 46 60 81 40
Telex: 250956

National Semiconductor S.p.A.
Strada 7, Palazzo R/3
20089 Rozzano
Milanofiori
Italy
Tel: (02) 8242046/7/8/9

National Semiconductor AB
Box 2016
Stensatravagen 13
S-12702 Skarholmen
Sweden
Tel: (08) 970190
Telex: 10731

National Semiconductor
Calle Agustin de Foxa, 27
28036 Madrid
Spain
Tel. (01) 733-2958
Telex: 46133

National Semiconductor Switzerland
Alte Winterthurerstrasse 53
Postfach 567
Ch-8304 Wallisellen-Zurich
Tel: (01) 830-2727
Telex: 59000

National Semiconductor
Kauppakartanonkatu 7
SF-00930 Helsinki
Finland
Tel: (0) 33 80 33
Telex: 126116

National Semiconductor Japan Ltd.
4-403 Ikebukuro, Toshima-ku
Tokyo 171, Japan
Tel: (03) 988-2131
Fax: 011-81-3-988-1700

National Semiconductor Hong Kong Ltd.
Southeast Asia Marketing
Austin Tower, 4th Floor
22-26A Austin Avenue
Tsimshatsui, Kowloon, H.K.
Tel: 852 3-7243645
Cable: NSSEAMKTG
Telex: 52996 NSSEA HX

National Semiconductor (Australia) PTY, Ltd.
Bldg. F, 4th Floor,
MacQuarie Tech Centre
11-17 Khartoum Rd.
North Ryde, N.S.W. 2113
Sydney, Australia
Tel: 61-2-887-4455
Telex: AA27173

National Semiconductor (PTE), Ltd.
200 Cantonment Road 13-01
Southpoint
Singapore 0208
Tel: 2252226
Telex: RS 33877

National Semiconductor (Far East) Ltd.
Taiwan Branch
P.O. Box 68-332 Taipei
7th Floor, Nan Shan Life Bldg.
302 Min Chuan East Road,
Taipei, Taiwan R.O.C.
Tel: (86) 02-501-7227
Telex: 22837 NSTW
Cable: NSTW TAIPEI

National Semiconductor (Far East) Ltd.
Korea Office
Room 612,
Korea Fed. of Small Bus. Bldg.
16-2, Yoido-Dong,
Youngdeungpo-Ku
Seoul, Korea
Tel: (02) 784-8051/3
Telex: K24942 NSRK

TL/2154 RRD-RRD45M097/Printed in U.S.A.