National Semiconductor

October 1980

INS8070-Series Microprocessor Family

General Description

The INS8070-Series of microprocessors (hereinafter referred to as the 70-Series Family) is intended for use in systems requiring the economy of a single chip, the flexibility of multiprocessing bus architecture and the power of 16-bit arithmetic operations.

Designed for simplicity and low-cost implementation for stand-alone multiprocessing and DMA systems, the members of the 70-Series family include:

INS8070 - 64 Bytes of RAM, no ROM
INS8072 - 64 Bytes of RAM, 2.5K ROM

The 70-Series family comprises bus-oriented microprocessors with on-board ROM and RAM and built-in multi-processing logic. Low-cost systems containing data may occupy the same bus and still run at full speed, as their external bus requirements are confined to the occasional access of data. Such systems include terminals, intelligent peripherals and multiprocessing systems (e. g., the solution of polynomials in real-time).

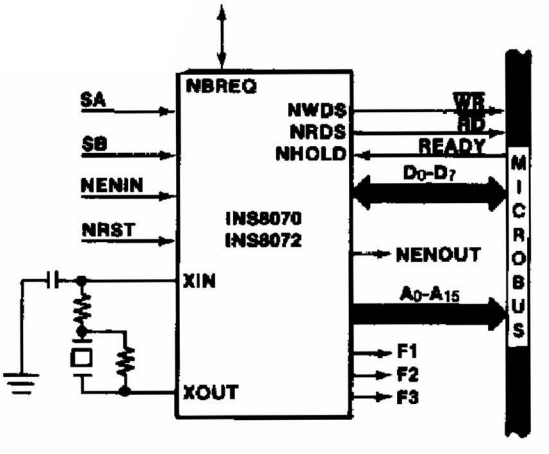
For very high-volume users, the instruction set, ROM and RAM size may be tailored to fit the necessary applications.

The 70-Series family is fabricated using National's Standard N-channel, silicon gate, depletion load, MOS technology. All the devices are housed in standard 40-pin dual-in-line packages.

Features

- On-Board ROM
- On-Board RAM (Executable)
- 8-Bit Data Bus
- 16-Bit Address Bus (64K Addressing Capability)
- Comprehensive Set of 8 and 16-Bit Arithmetic, Logic and Stack Manipulation Instructions
- Hardware 16 x 16 Bit Multiply (37 μs) and Divide (42 μs)
- Built-in Multiprocessing and DMA Logic
- Interfaces with Memory and Standard INS8080 Peripherals at Any Clock Speed
- On-Chip Clock Generation
- Single Instruction Character Search and Single Instruction ASCII to Decimal Conversion
- Full Hardware and Software Development Systems Available on STARPLEX ™ and ISE ™
- Single 5-Volt Supply
- MICROBUS™ Compatible

70-Series Family to MICROBUS Configuration



STARPLEX, ISE are trademarks of National Semiconductor Corp.

μPG-FL5M110/Printed in U.S.A.

Absolute Maximum Ratings

Max. Voltage to Any Input with Respect to GND .... -0.5V to +7.0V
Operating Temperature..... 0° C to +70° C
Storage Temperature..... -65° C to 70° C

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.*

DC Electrical Characterisitics

(T_A = 0° C to +70° C, V_{CC} = +5V ±5%)

Parameter	Min.	Typ	Max	Units	Conditions
INPUT SPECIFICATIONS					
All Input Pins Except V_{CC} and GND, NRST, and Sense Inputs SA and SB (See Note 1.)					
Logic 1 Input Voltage	2.0			V	
Logic 0 Input Voltage			0.8	V	
Inputs with Schmitt Trigger Input Stage (NRST, SA, SB)					
V_T + Positive Going Threshold Voltage			2.4	V	
V_T - Negative Going Threshold Voltage	0.8			V	
Hysteresis (V_T+ - V_T-)	0.2			V	
Input Capacitance (All pins except V_{CC} and GND)		6		pF	
Supply Current I_{CC}		100 mA		mA	T_A = 25° C Outputs unloaded

NOTE 1: *SA and SB are not buffered by the Schmitt Trigger in the interrupt path and meet the normal Input Voltage specifications when used as Interrupt Inputs.*

OUTPUT SPECIFICATIONS					
TRI-STATE® Pins (NWDS, NRDS, DB7-DB0, AD15-AD00)					
Logic 1 Output Voltage	2.4			V	I_{OUT} = -100 μA
Logic 0 Output Voltage			0.4	V	I_{OUT} = 1.6 mA
TRI-STATE Output Current			±10	μA	0 < V_{OUT} < V_{CC}
FLAG 1-3, NENOUT					
Logic 1 Output Voltage	V_{CC} -1			V	I_{OUT} = -100 μA
Logic 1 Output Voltage	1.5			V	I_{OUT} = -1 mA
Logic 0 Output Voltage			0.4	V	I_{OUT} = 1.6 mA
NBREQ					
Logic 0 Output Voltage			0.4	V	I_{OUT} = 1.6 mA
Logic 1 Output Current			±10	μA	0<V_{OUT}<V_{CC}
XOUT					
Logic 1 Output Voltage	2.4			V	I_{OUT} = -100 μA
Logic 0 Output Voltage			0.4	V	I_{OUT} = 1.6 mA

AC Electrical Characteristics

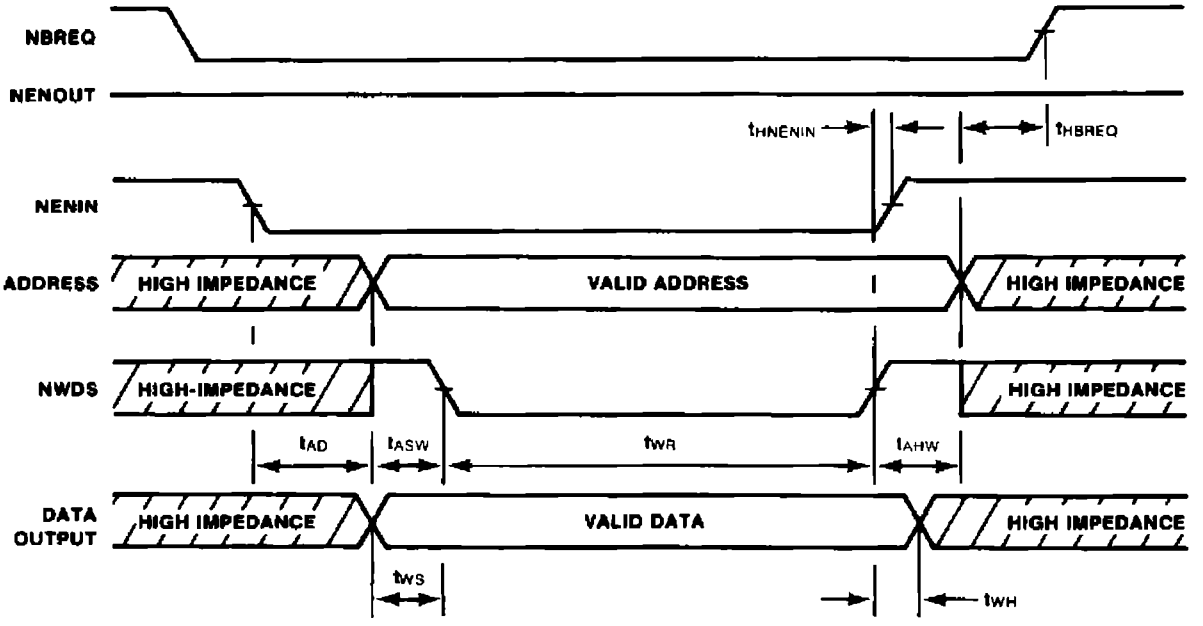
(T_A = +25°C, V_{CC} = +5V ±5%, GND = 0V)

Parameter	Min.	Typ	Max.	Unit	Test Conditions (See Note)
EXTERNAL MEMORY f_x (Input Frequency)	1.0		4.0	MHz	Note: tx is the period (1) fx of the XIN input. When f_x = 4MHz, tx = 250ns.
Read Cycle t_{RD}	4tx-50	4tx		ns	
t_{ASR}	1x-150	tx		ns	
t_{AHR}	tx- 75	tx		ns	
t_{DS}	400		600	ns	
t_{DH}		0		ns	
t_{AD}		2tx		ns	
t_{HBREQ}	-50	25		ns	
t_{HNENIN}	0			ns	
Write Cycle t_{WR}	3tx-100	3tx		ns	
t_{ASW}	2tx-150	2tx		ns	
t_{AHW}	tx- 50	tx		ns	
t_{WS}	2tx-200	2tx		ns	
t_{WH}	tx- 50	tx		ns	
t_{AD}		2tx		ns	
t_{HBREQ}	-50	25		ns	
t_{HNENIN}	0			ns	
EXTERNAL READ/WRITE CYCLE EXTENSION t_{D1} (HOLD) Read Cycle			2tx	ns	See *Figure 3*
Write Cycle			tx	ns	
t_W (HOLD)			∞	ns	
t_S (HOLD)	2tx			ns	
t_{D2} (HOLD)		3tx		ns	
t_{D3} (HOLD)		4tx		ns	
Capacitances Output Load (Xout)		30		pF	
All Other Output Pins		75		pF	

Timing Waveforms

Timing Waveforms

Write Cycle

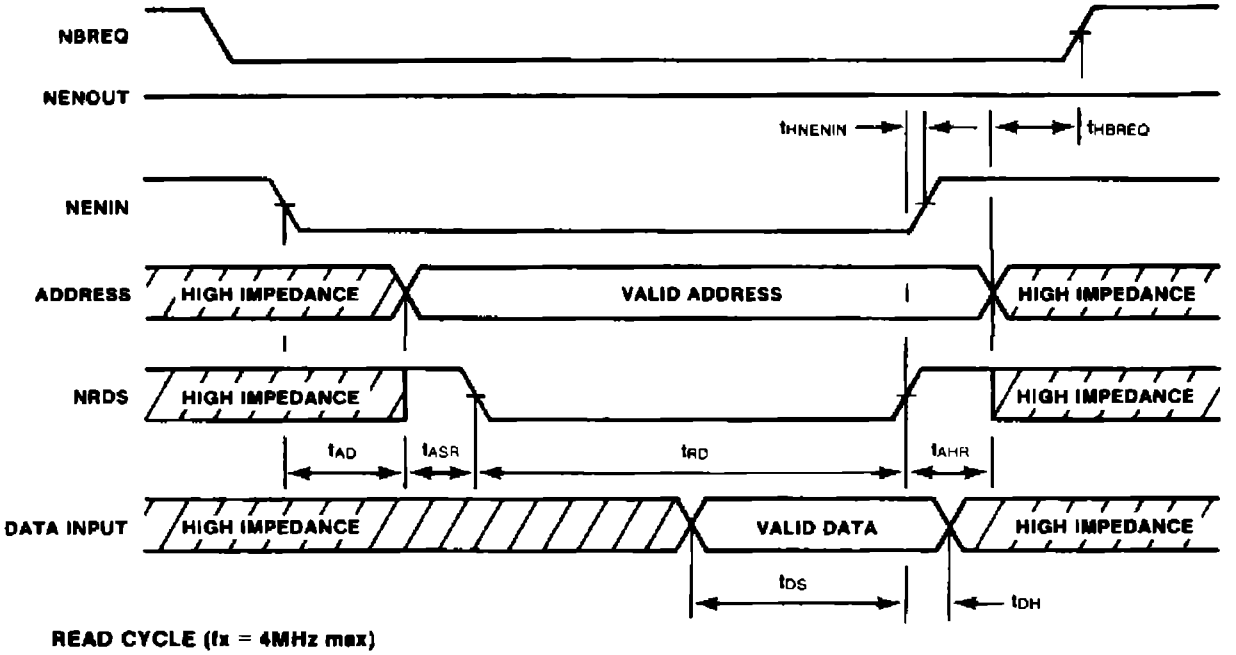


WRITE CYCLE (fx = 4MHz max)

t_{WR}	= 3tx ns typ	t_{AH}	= tx ns typ	t_{AD}	= 2tx ns typ
t_{AS}	= 2tx ns typ	t_{WS}	= 2tx ns typ	t_{HBREQ}	= 50 typ
		t_{WH}	= tx ns typ	t_{HNENIN}	= 0

18-8

Read Cycle



READ CYCLE (fx = 4MHz max)

$t_x = \frac{1}{f_x}$

t_{RD}	= 4tx ns typ	t_{AS}	= tx ns typ	t_{DH}	= 0
		t_{AH}	= tx ns typ	t_{AD}	= 2tx ns typ
		t_{HNENIN}	= 0	t_{HBREQ}	= 50 typ

18-7

INS807X Functional Pin Description

The following describes the function of all INS807X input/output pins. For brevity, the INS807X designator is used to refer to all members of the family.

INPUT SIGNALS

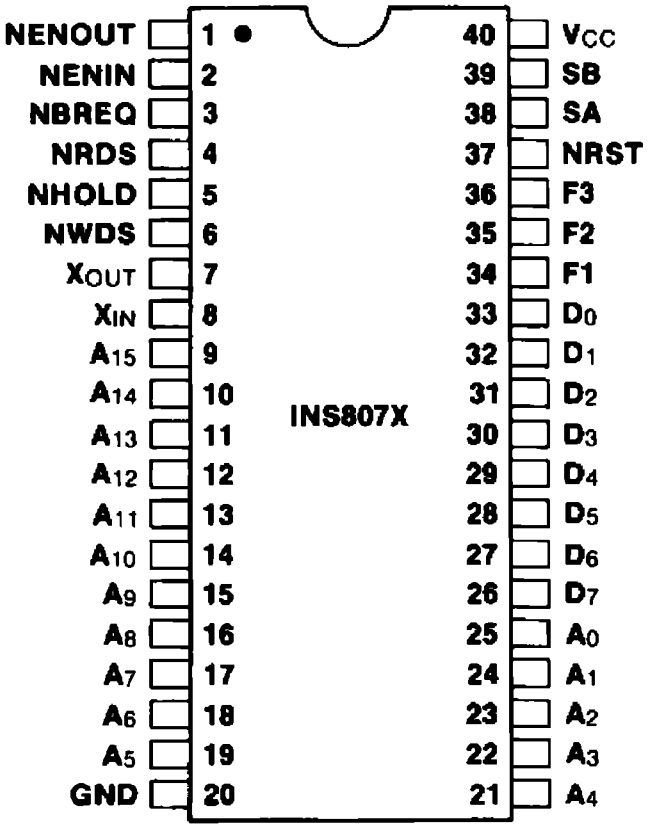
Reset	(NRST)	Active low RESET input. This pin initializes the INS807X by resetting the program counter (PC) and the stack pointer (SP) to zero (low). All bits in the Status Register (except SA and SB) are also reset to zero. This input is buffered by a T^2L compatible Schmitt Trigger, allowing slow rise and fall times.
Hold	(NHOLD)	Active low external memory cycle extend input for slow memories and slow peripherals. Also used for single memory cycle execution. NHOLD affects READ/WRITE cycles to external memory only.
Sense A/Interrupt A	(SA/INTA)	Provides a T^2L compatible Schmitt Trigger buffered sense input which sets/resets the Sense A bit (SA) in the status register. When interrupts are enabled (IE = 1 in the STATUS register) this pin also acts as a trailing edge triggered interrupt input.
Sense B/Interrupt B	(SB/INTB)	Similar operation as SA/INTA. Since SA/INTA has priority over SA/INTB, if both arrive simultaneously, Interrupt A will be serviced first.
Enable Input	(NENIN)	Active low bus enable input to the on-chip bus allocation logic. For DMA and multiprocessing applications this logic allows bus access to be granted or denied to the INS807X. Devices which can share the bus include other INS807X processors, INS8060 processors, DP8350 CRT Controller and DMA logic. NENIN affects INS807X operation as follows: 1) If NENIN is high, INS807X sets NENOUT high and is denied access to the bus. 2) If NENIN is low and the INS807X is holding NBREQ low, INS807X sets NENOUT high and is granted access to the bus. 3) If NENIN and NBREQ are both low and the INS807X is not holding NBREQ low, INS807X sets NENOUT low, and is denied access to the bus.
Power	-	V_{CC} = +5V, Pin 40 V_{SS} = GND, Pin 20

OUTPUT SIGNALS

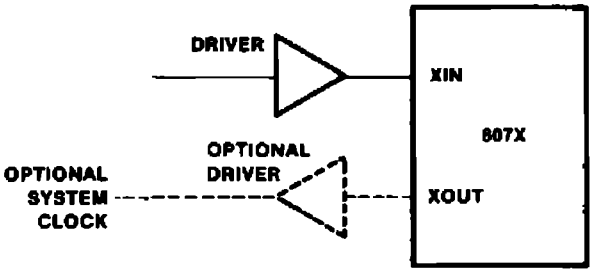
Address Bus	(A_{15} - A_0)	TRI-STATE™ address outputs for all external memory READ and WRITE operations. These outputs are normally in high-impedance state except during external memory READ and WRITE operations (Refer to Timing Waveforms.)
Write Data Strobe	(NWDS)	TRI-STATE active low WRITE strobe output. NWDS goes low during an external memory WRITE operation. NWDS is in the high-impedance state when an external memory WRITE operation is not in progress. (Refer to Write Cycle Timing.)
Read Data Strobe	(NRDS)	TRI-STATE active low READ strobe output. NRDS goes low during an external memory READ operation. NRDS is in the high-impedance state when an external memory READ operation is not in progress. (Refer to Read Cycle Timing.)
Enable Output	(NENOUT)	Active low bus enable output from the on-chip bus allocation logic. INS807X controls NENOUT as follows: 1) If NBREQ is low but INS807X is not holding it low, NENOUT = NENIN.

		2) If NEBREQ is low and INS807X is holding it low, NENOUT = high.
		3) If NBREQ = high, NENOUT = high.
Flags 1, 2, 3	(F1, F2, F3)	Flag outputs which can be set high/low by writing into the corresponding flag bits of the status register.
	BIDIRECTIONAL SIGNALS	
Bus Request	(NBREQ)	Bidirectional Bus Request input/output. NBREQ serves as an input/output to/from the on-chip bus allocation logic. When NBREQ is high (NBREQ acts as an input) and the processor requires use of the bus, the processor issues a bus request by pulling NBREQ low (NBREQ acts as an output.)
Data Bus	(D_0 - D_7)	Bidirectional TRI-STATE data bus input/output. These lines are normally in the high-impedance state except during external memory READ and WRITE operations. They provide input data during external memory READ; they provide output data during external memory WRITE.
Clock Inputs	(X_{IN}, X_{OUT})	On-chip clock generator input/outputs (see figure 1). The on-chip clock generator will operate with external crystal or RC inputs connected between X_{IN} and X_{OUT}. The on-chip clock generator may be disabled by supplying a T^2L level clock input on X_{IN}. In all modes a buffered T^2L level clock output is always available at X_{OUT}.

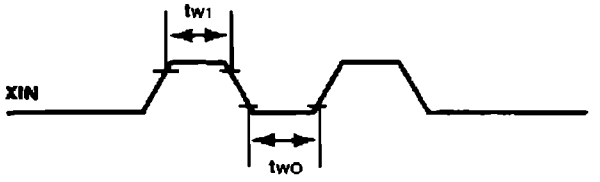
PIN CONFIGURATION



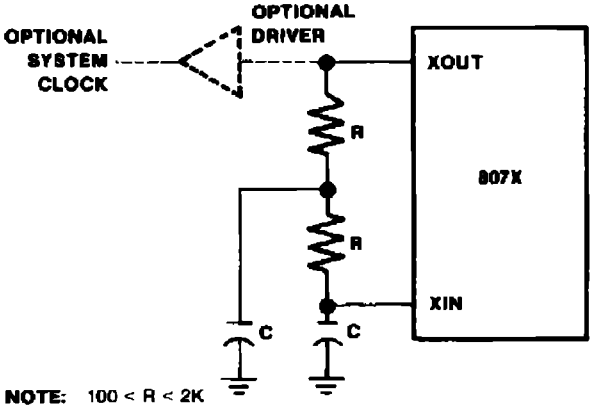
A. External Clock Input



EXTERNAL CLOCK PARAMETERS



B. Resistor Capacitor Feedback Network



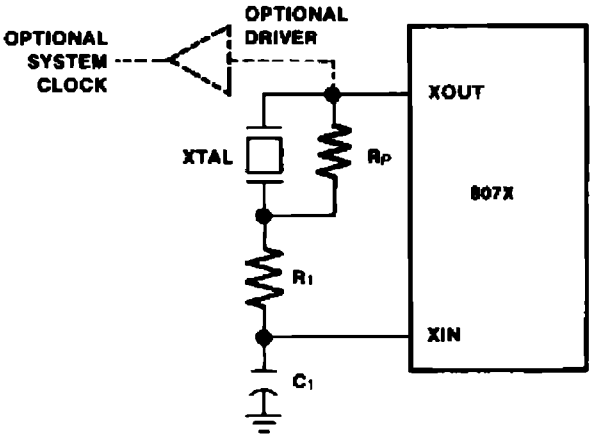
NOTE: 100 < R < 2K

Approximate Oscillator Frequency vs RC Time Constant

$$t = \frac{1}{f} = 0.725\ RC + 353$$

where: R is in KΩ
C is in pf
t is in ns

C. Crystal with Low Pass Filter (Above 1 MHz)

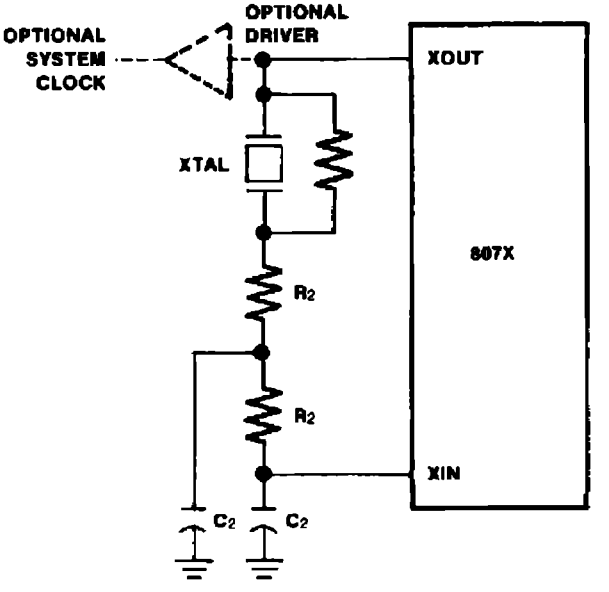


Typical values for Crystal with Low Pass Filter Network

Crystal	Rp	C1	R1
2 MHz	100kΩ	56pF	1kΩ
3.58 MHz	100kΩ	27pF	1kΩ
4 MHz	100kΩ	27pF	1kΩ

XTAL is parallel resonant with maximum series resonance equal to 1kΩ

D. Crystal with Low Pass Filter (1 MHz or Below)



NOTE: The values of R & C are determined by the operating frequency and crystal type.

FIGURE 1. Frequency Control Networks for On-Chip Oscillator

18-4

FUNCTIONAL DESCRIPTION

The 8072, 8070 are members of a single chip, general-purpose microprocessor family containing 2.5K and 0K bytes, respectively, of on-board ROM and 64 bytes of on-board RAM. They may be used in stand-alone, DMA (Direct Memory Access), and multiprocessor applications.

Communication between the 70-Series family and external memory/peripheral devices is effected via a 16-bit address bus and an 8-bit bidirectional data bus.

READ and WRITE strobe outputs from INS807X - (NRDS, NWDS) indicate when valid input/output data are present on the 8-bit data bus.

The remaining input/output signals shown in the block diagram are dedicated to initialization, clock generation, bus management, interrupt request, input/output cycle extension, and user-definable interface functions.

Drivers and Receivers

Equivalent circuits for INS807X drivers and receivers are shown in *Figure* 2. All inputs have static charge protection circuits consisting of an RC filter and voltage clamp. These devices should still be handled with care, as the protection circuits can be destroyed by excessive static charge.

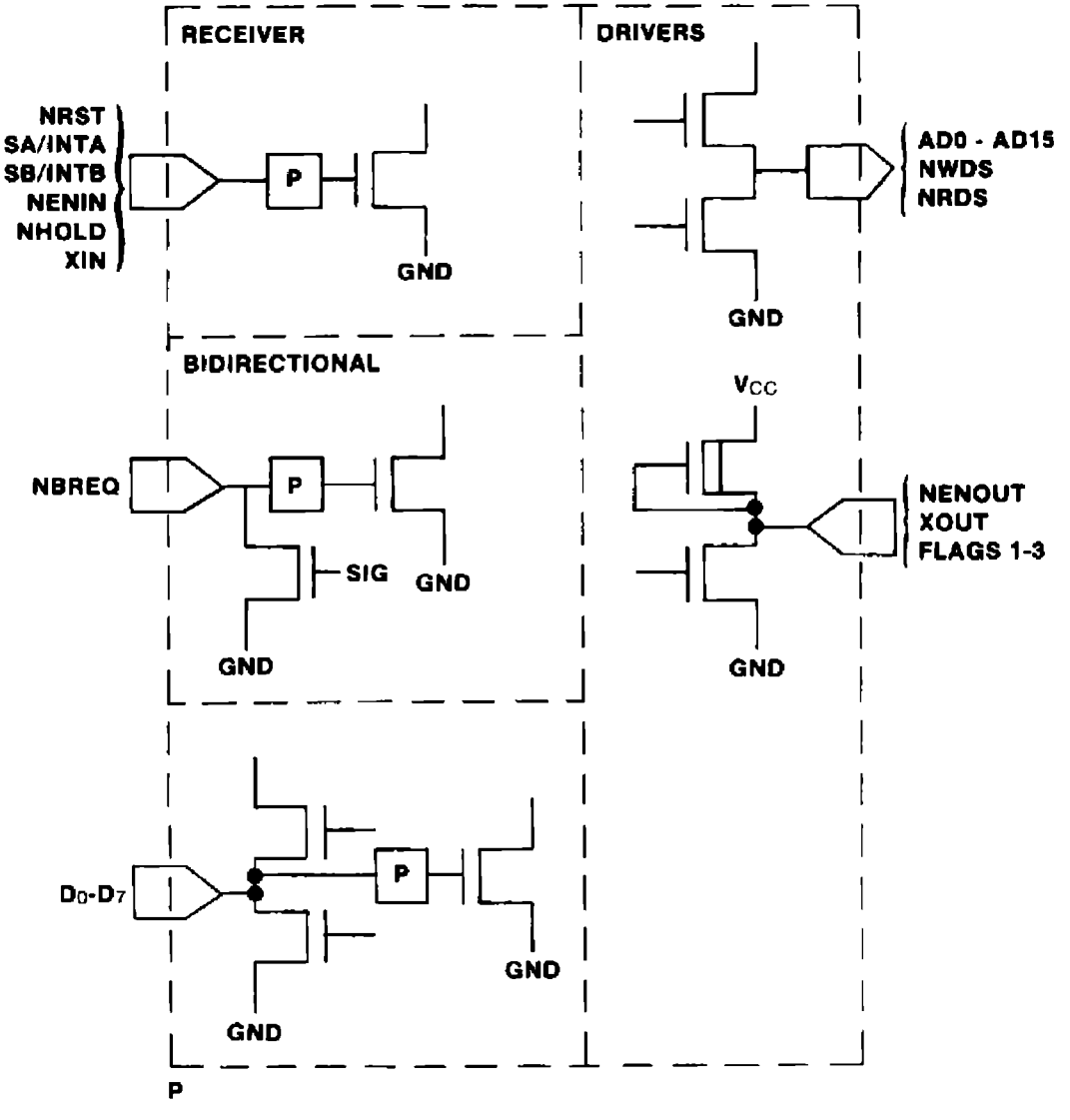


FIGURE 2. INS807X Drive and Receiver Equivalent Circuits

I/O Control Logic

The I/O Control Logic generates the basic timing for internal/external memory READ/WRITE operations. This includes extension of external READ/WRITE cycles in order to accomodate slow memory/peripherals

Instruction Decode and Control PLA:

During normal operation, the INS807X repeatedly fetches and executes instructions stored in internal ROM/RAM or in external memory. Instruction access and execution are carried out under the control of an internal microprogram stored in a Programmable Logic Array (PLA).

Bus Access Control

Before the INS807X can initiate data transfers with external peripherals or external memory, it must have access to the external address and data buses. Three of the INS807X I/O pins are associated with bus access control: NBREQ, NENIN and NENOUT. For simple stand-alone applications, the NENOUT signal can be ignored; the NENIN pin should be grounded or connected to NBREQ. The NBREQ pin must be connected to V_{CC} through an external pullup resistor.

Extended Bus Access Timing Considerations

There are four possible permutations of data and instruction accesses:

a. Instruction and data both internal
b. Instruction and data both external
c. Instruction internal, data external
d. Instruction external, data internal

For an external instruction, a number of microcycles equal to the number of bytes fetched by the instruction must be added to the total internal execution time.

For external data, a number of microcycles equal to the number of data bytes fetched or written must be added to the total instruction execution time.

The appropriate combination should be selected and the correct number of microcycles added to the execution time.

Exceptions to this general rule are:

a. Immediate data - treated as part of the instruction.

b. Increment, Decrement and Load - one data byte is accessed twice therefore, two microcycles must be added.

c. One byte call - three separate areas of memory are addressed.

   1) Single byte call
   1) Address
   3) Stack

Therefore, each area must be defined as internal or external.

NOTE: *Stack accesses must also be factored, as they are normally treated as data.*

External READ/WRITE Cycle Extension

As shown in *Figure 3*, the NHOLD signal may be set low following the leading edge of the READ strobe (NRDS) or the WRITE strobe (NWDS). Setting NHOLD low causes the INS807X to extend an external I/O (READ or WRITE) cycle. When an external I/O cycle is extended, the INS807X address, data and strobe lines are held active until the NHOLD signal is returned high. There is no restriction for the maximum duration of NHOLD in the low state. Thus, NHOLD can be utilized in a variety of ways, ranging from accommodation of slow memories/peripherals to single READ/WRITE cycle execution for software debug purposes.

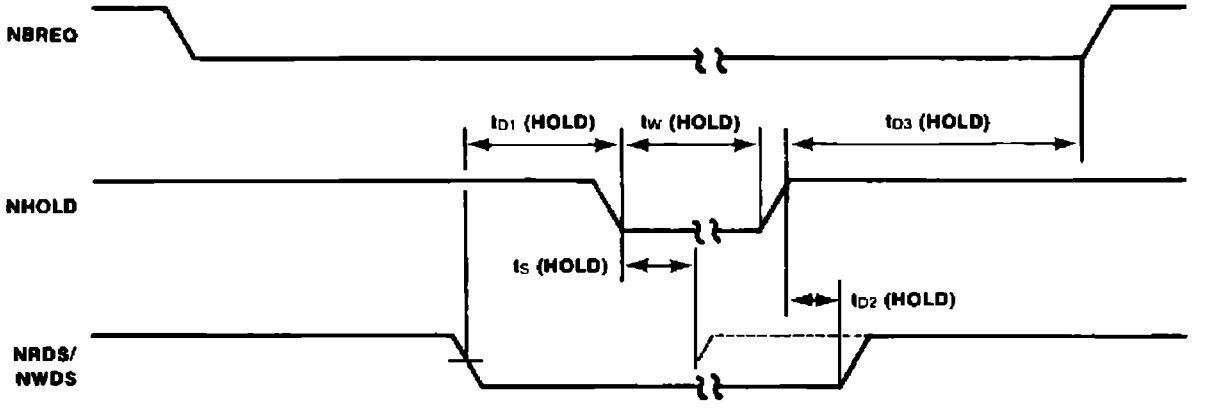


FIGURE 3. Extended Input/Output Timing Using NHOLD

Interrupt Servicing

When interrupts are enabled (IE = 1 in the STATUS register), pins SA/INTA and SB/INTA act as vectored interrupts. Both inputs are edge triggered, and are activated by the high to low logic transition. (Edge triggered interrupts offer the advantage of not requiring an interrupt acknowledge signal to remove the interrupt service request, as required by level triggered interrupts.) Interrupt service requests are latched when they occur, and are recognized prior to each instruction fetch. When an interrupt is recognized, the contents of the program counter are pushed onto the stack, saving the return address. The interrupt enable bit (IE) in the STATUS register is automatically reset, disabling further interrupts from being recognized, but interrupts occurring during this time will still be latched. The processor then fetches the next instruction from a predetermined memory location. The location for Interrupt A is 0004_{16} and location for interrupt B is 0007_{16}.(Refer to *Figure 4*.) The instructions stored in these vector locations are normally BRANCH or JUMP instructions, causing the processor to begin execution of the appropriate service subroutine. The interrupt service subroutines may be of arbitrary length and may be located anywhere in memory.

Since the Interrupt Enable bit (IE) is automatically cleared at the start of interrupt service, Interrupt A cannot normally interrupt itself or Interrupt B, and vice versa, unless interrupts are re-enabled in the service subroutine. Since interrupt requests are latched, an interrupt request will never be lost even though it occurs during servicing of the other interrupt.

Interrupt service subroutines should always be concluded with an interrupt enable instruction (ORS = 01_{16}) immediately followed by a return instruction (RET). This sequence causes the processor to resume main program execution at the point where it was interrupted.

Interrupt Timing and Interrupt Priority

Interrupts are latched when they occur and are sampled prior to each instruction fetch. Thus, when an interrupt occurs during execution of an instruction, the instruction in process must be completed before the interrupt will be recognized. This results in an Interrupt Recognize Latency Time (IRLT), which is a function of the time at which the asynchronous interrupt occurs. The minimum value for the IRLT is 9 microcycles, and the maximum value is equal to the duration of the longest program instruction (in microcycles) plus 9 microcycles.

Interrupt A Always has Priority Over Interrupt B

If both interrupts occur during the IRLT, Interrupt A will always be recognized first, even though Interrupt B may have occured slightly earlier. In order for Interrupt B to be recognized before Interrupt A, Interrupt B must occur prior to Interrupt A by an amount greater than the maximum value of the IRLT.

If several interrupt requests occur on the same pin during the IRLT, they will be treated as a single interrupt request. If two interrupt requests occur on the same pin spaced apart by a time duration greater than the IRLT, they will be treated as two separate requests. For this case, if the second interrupt occurs during interrupt service, service will not be interrupted unless interrupts have been re-enabled by the interrupt service routine.

The Interrupt Service Latency Time (ISLT) is the time duration measured from the active edge of the interrupt request to the start of execution of the first instruction in the interrupt service routine. When interrupts are enabled, the ISLT is equal to the IRLT plus 8 microcycles or 5 microcycles, depending upon whether a JUMP or BRANCH instruction is stored at the vector memory locations 0004_{16} and 0007_{16}.

MEMORY LOCATION	MEMORY CONTENTS	
0		
1	JMP	RESTART
2	XX	
3	XX	
4	JMP	INTERRUPT A
5	XX	
6	XX	
7	JMP	INTERRUPT B
8	XX	
9	XX	
A		

FIGURE 4. Interrupt Transfer Locations

70-SERIES FAMILY INSTRUCTION SET

INSTRUCTION	MNE-MONIC	FIRST OPERAND	SECOND OPERAND	ADDRESSING												FLAGS AFFECTED								OPERATION PERFORMED
				PC RELATIVE	POINTER RELATIVE			IMMEDIATE	DIRECT	AUTO-INDEXED		IMPLIED	INDEXED		ABSOLUTE or INDIRECT									
					SP	P2	P3			P2	P3		P2	P3										
				OP ~ #	OP ~ #	OP ~ #	OP ~ #	OP ~ #	OP ~ #	OP ~ #	OP ~ #	OP ~ #	OP ~ #	OP ~ #	OP ~ #	CY	OV	SB	SA	F3	F2	F1	IE	
LOAD	LD	A		C0 7 2	C1 7 2	C2 7 2	C3 7 2	C4 5 2	C5 7 2	C6 8 2	C7 8 2													(A)←(addr)
		EA		80 10 2	81 10 2	82 10 2	83 10 2	84 8 3	85 10 2	86 11 2	87 11 2													(EA)←(addr + 1, addr)
		T		A0 10 2	A1 10 2	A2 10 2	A3 10 2	A4 8 3	A5 10 2	A6 11 2	A7 11 2													(T)←(addr + 1, addr)
		SP						25 8 3																(SP)←(PC + 2, PC + 1)
		P2						26 8 3																(P2)←(PC + 2, PC + 1)
		P3						27 8 3																(P3)←(PC + 2, PC + 1)
		A	E									40 4 1												(A)←(E)
		E	A									48 4 1												(E)←(A)
		A	S									06 3 1												(A)←(S)
		S	A									07 3 1												(S)←(A)
		EA	PC									30 4 1												(EA)←(PC)
		EA	SP									31 4 1												(EA)←(SP)
		EA	P2									32 4 1												(EA)←(P2)
		EA	P3									33 4 1												(EA)←(P3)
		SP	EA									45 5 1												(SP)←(EA)
		P2	EA									46 5 1												(P2)←(EA)
		P3	EA									47 5 1												(P3)←(EA)
		T	EA									09 4 1												(T)←(EA)
		EA	T									0B 4 1												(EA)←(T)
STORE	ST	A		C8 7 2	C9 7 2	CA 7 2	CB 7 2		CD 7 2	CE 8 2	CF 8 2													(A)→(addr)
		EA		88 10 2	89 10 2	8A 10 2	8B 10 2		8D 10 2	8E 11 2	8F 11 2													(AE)→(addr + 1, addr)
ADD	ADD	A		F0 7 2	F1 7 2	F2 7 2	F3 7 2	F4 7 2	F5 7 2	F6 8 2	F7 8 2					•	•							(A)←(A) + (addr)
		A	E									70 4 1				•	•							(A)←(A) + (E)
		EA		B0 10 2	B1 10 2	B2 10 2	B3 10 2	B4 10 2	B5 10 2	B6 11 2	B7 11 2					•	•							(EA)←(EA) + (addr + 1, addr)
SUBTRACT	SUB	A		F8 7 2	F9 7 2	FA 7 2	FB 7 2	FC 7 2	FD 7 2	FE 8 2	FF 8 2					•	•							(A)←(A) - (addr)
		A	E									78 4 1				•	•							(A)←(A) - (E)
		EA		B8 10 2	B9 10 2	BA 10 2	BB 10 2	BC 10 2	BD 10 2	BE 11 2	BF 11 2					•	•							(EA)←(EA) - (addr + 1, addr)
MULTIPLY	MPY	EA	T									2C 37 1				•	•							(EA)←[(EA) * (T)] 31:16; (T)←[(EA) * (T)] 15:0
DIVIDE	DIV	EA	T									0D 41 1				•	•							Quotient: (EA)←[(EA) / (T)] 15:0 Remainder: (T)←[(EA) / (T)]
AND	AND	A		D0 7 2	D1 7 2	D2 7 2	D3 7 2	D4 7 2	D5 7 2	D6 8 2	D7 8 2													(A)←(A) /\ (addr)
		A	E									50 4 1												(A)←(A) /\ (E)
		S						39 5 2								•	•			•	•	•	•	(S)←(S) /\ data
OR	OR	A		D8 7 2	D9 7 2	DA 7 2	DB 7 2	DC 7 2	DD 7 2	DE 8 2	DF 8 2													(A)←(A) \/ (addr)
		A	E									58 4 1												(A)←(A) \/ (E)
		S						3B 5 2								•	•			•	•	•	•	(S)←(S) \/ data
EXCLUSIVE-OR	XOR	A		E0 7 2	E1 7 2	E2 7 2	E3 7 2	E4 7 2	E5 7 2	E6 8 2	E7 9 2													(A)←(A) ⊻ (addr)
		A	E									60 4 1												(A)←(A) ⊻ (E)

INSTRUCTION	MNE-MONIC	FIRST OPERAND	SECOND OPERAND	ADDRESSING												FLAGS AFFECTED								OPERATION PERFORMED
				PC RELATIVE	POINTER RELATIVE			IMMEDIATE	DIRECT	AUTO-INDEXED		IMPLIED	INDEXED		ABSOLUTE or INDIRECT									
					SP	P2	P3			P2	P3		P2	P3										
			•	OP ~ #	OP ~ #	OP ~ #	OP ~ #	OP ~ #	OP ~ #	OP ~ #	OP ~ #	OP ~ #	OP ~ #	OP ~ #	OP ~ #	CY	OV	SB	SA	F3	F2	F1	IE	
EXCHANGE REGISTERS	XCH	A	E									01 5 1												(A)← →(E)
		EA	SP									4D 7 1												(EA)← →(SP)
		EA	P2									4E 7 1												(EA)← →(P2)
		EA	P3									4F 7 1												(EA)← →(P3)
SHIFT RIGHT	SR	A										3C 3 1												(Ai)→(Ai-1); i = 7, 1; 0→A7
		EA										0C 4 1												(EAi)→(EAi-1); i = 15, 1; 0→EA15
SHIFT RIGHT WITH LINK	SRL	A										3D 3 1				•								(Ai)→(Ai-1); i = 7, 1; CY→A7
ROTATE RIGHT	RR	A										3E 3 1												(Ai)→(Ai-1); i = 7, 1; A0→A7
ROTATE RIGHT WITH LINK	RRL	A										3F 3 1				•								(Ai)→(Ai-1); i = 7, 1; A0→CY→A7
SHIFT LEFT	SL	A										0E 3 1												(Ai+1)←(Ai); i = 6, 0; 0→A0
		EA										0F 4 1												(EAi+1)←(EAi); i = 14, 0; 0→EA0
SEARCH AND SKIP IF CHARACTER MATCHED	SSM												2E 1	2F 1										See Text
BRANCH IF NOT DIGIT	BND			2D 2																				If (A) ≠ ASCII: (PC)← addr If not digit: 7 or 9, If digit: 7
PUSH	PUSH	A										0A 5 1												(SP)←(SP) - 1; ((SP))←(A)
		EA										08 8 1												((SP)) - 1←(E); ((SP)) - 2←(A); (SP)←(SP) - 2
		PC										54 8 1												((SP)) - 1←(PCH); ((SP)) - 2←(PCL); (SP)←(SP) -2
		P2										56 8 1												((SP)) - 1←(P2H); ((SP)) - 2←(P2L); (SP)←(SP) - 2
		P3										57 8 1												((SP)) - 1←(P3H); ((SP)) - 2←(P3L); (SP)←(SP) - 2
PUSH AND LOAD IMMEDIATE	PLI	P2						22 15 3																((SP)) - 1←(P2H); ((SP)) - 2←(P2L); (SP)←(SP) - 2; (P2H)←byte 3; (P2L)←byte 2
		P3						23 15 3																((SP)) - 1←(P3H); ((SP)) - 2←(P3L); (SP)←(SP) - 2; (P3H)←byte 3; (P3L)←byte 2
POP	POP	A										38 6 1												(A)←((SP)); (SP)←(SP) + 1
		EA										3A 9 1												(A)←((SP)); (E)←((SP)) + 1; (SP)←(SP) + 2
		P2										5E 10 1												(P2L)←((SP)); (P2H)←((SP)) + 1; (SP)←(SP) + 2
		P3										5F 10 1												(P3L)←((SP)); (P3H)←((SP)) + 1; (SP)←(SP) + 2
BRANCH UNCONDITIONAL	BRA			74 5 2		76 5 2	77 5 2																	(PC)←(PC) + disp
BRANCH POSITIVE	BP			64 5 2		66 5 2	67 5 2																	If (A) > 0: (PC)←(PC) + disp
BRANCH ZERO	BZ			6C 5 2		6E 5 2	6F 5 2																	If (A) = 0: (PC) ←(PC) + disp
BRANCH NOT ZERO	BNZ			7C 5 2		7E 5 2	7F 5 2																	If (A) ≠ 0: (PC)←(PC) + disp
JUMP UNCONDITIONAL	JMP														24 6 3									(PCH)←byte 3; (PCL)←byte 2
JUMP TO SUBROUTINE	JSR														20 15 3									((SP)) - 1←(PCH)←byte 3 ((SP)) - 2←(PCL)←byte 2, (SP)←(SP) - 2
CALL	CALL	0 - 15													10-1F 17 1									((SP)) - 1←(PCH)←(021 + 2'N); ((SP)) - 2←(PCL)←(020 + 2'N); (SP)←(SP) - 2
RETURN	RET											5C 10 1												(PCL)←((SP)); (PCH)←((SP)) + 1; (SP)←(SP) + 2
LOAD PC	LD	PC	EA									44 5 1												(PC)←(EA)
EXCHANGE PC	XCH	PC	EA									4C 7 1												(PC)← →(EA)
INCREMENT AND LOAD	ILD	A		90 8 2	91 8 2	92 8 2	93 8 2		95 8 2	96 9 2	97 9 2													(A), (addr)←(addr) + 1
DECREMENT AND LOAD	DLD	A		98 8 2	99 8 2	9A 8 2	9B 8 2		9D 8 2	9E 9 2	9F 9 2													(A), (addr)←(addr) - 1
NO OPERATION	NOP											00 3 1												(PC)←(PC) + 1

INS807X Architecture and Internal Registers

As shown in the block diagram (*Figure 5*), INS807X contains the equivalent of 18, 8-bit registers; their functions are described below:

Program Counter: PC The program counter is a 16-bit register that contains the address of the instruction to be executed. The program counter is incremented before each instruction fetch. The program counter contents may also be replaced by the contents of a specified memory location if the last instruction was a branch, jump, or jump to subroutine instruction. This causes execution to continue from the program specified location rather than from the next sequential location. The PC may also be exchanged with the contents of the Extended Accumulator (EA).

Pointer Registers: (SP, P2, P3) The three pointers, SP, P2, and P3 are 16-bit registers that may be modified under program control. A pointer which has been loaded with a given address can reference up to 256 memory or I/O bytes without being modified. This is accomplished using pointer relative addressing.

The stack pointer, SP, may be used as a general purpose pointer and/or to point to the top of a RAM-resident stack that is implicitly referenced by the CALL, JSR, RET, PUSH, POP and PLI instructions. Under program control, this stack may be located anywhere in memory and may be of arbitrary length.

In addition to the stack referenced by the stack pointer (SP), pointers P2 and P3 may also be used to reference other stacks. This is accomplished using auto-indexed addressing.

Accumulator: AC The 8-bit Accumulator Register (AC) is the primary working register of the INS807X and contains the results of all single byte (8-bit) arithmetic and logic operations. Its contents may also be shifted, rotated, exchanged with the Extension Register (E) and moved on and off the stack using the PUSH and POP instructions.

Extension Register: E The 8-bit Extension Register (E) is used for temporary storage, to perform arithmetic and logic operations in conjunction with the Accumulator (AC) and combined with the accumulator to form the Extended Accumulator (EA). (For 16-bit operations.)

Double Precision Accumulator Register: EA The 16-bit Double Precision Accumulator Register (EA) is the primary double byte working register of the INS807X. It is formed by the concatenation of the 8-bit Accumulator Register (AC) and the 8-bit Extension Register (E). All double byte words are transferred between (EA) and memory in two single byte transfers.

Temporary Register: T The 16-bit Temporary Register (T) is used for temporary storage of double byte words. It is also used to hold the 16-bit multiplier and the 16-bit divisor in multiplication/division. The (T) register may be loaded directly from memory and its contents may also be transferred to/from the (EA) register.

Status Register: S The 8-bit Status Register (S) is used to store the following status bits: Carry/Link (CY/L), 2's Complement Overflow (OV), Sense B (SB), Sense A (SA), flags (F1, F2, F3) and the Interrupt Enable bit (IE).

The bit positions of the status register are shown below:

Bit positions

7	6	5	4	3	2	1	0	
CY/L	OV	SB	SA	F3	F2	F1	IE	Flags

(CY/L) goes high to indicate a carry output following execution of an addition/subtraction instruction.

Status register bits 1, 2 and 3 are buffered and connected to the flag output pins F1, F2, and F3. These pins are particularly useful for direct I/O control and for execution of serial I/O operations in conjunction with the SA and SB sense inputs.

SA and SB are Schmitt triggered buffered inputs which may be tested under software control. They form bits 4 and 5 of the Status Register and are read-only bits. All other Status Register bits may be both written to or read from using the AND S and the OR S instructions, respectively.

The Interrupt Enable bit (IE), in the Status Register may be set/reset under program control by the OR S and AND S instructions. When set high, IE enables interrupts A and B. When IE is low, both interrupts are disabled. Following a hardware reset via the NRST pin, IE is automatically set low, disabling both interrupts.

NOTE: *All Status Register bits except for SA and SB are latched. When set to a specified state under program control, they will remain in that state until again modified under program control.*

Instruction Register: This instruction register is not accessible to the programmer. During the fetch cycle of each instruction, this register is loaded with the 8-bit instruction op code retrieved from memory.

Auxiliary Register and Data I/O Register: These 8-bit registers are not accessible to the programmer. They are used for temporary storage of input/output bytes transmitted over the 8-bit bidirectional data bus.

Address Register: The 16-bit address register is not accessible to the programmer. It is used for temporary storage of the 16-bit address transmitted during a memory READ/WRITE cycle.

Arithmetic Logic Unit (ALU): The Boolean operations provided by the ALU include AND, OR, and XOR. The ALU also performs increment, decrement, add, subtract, multiply, divide, shift and rotate operations.

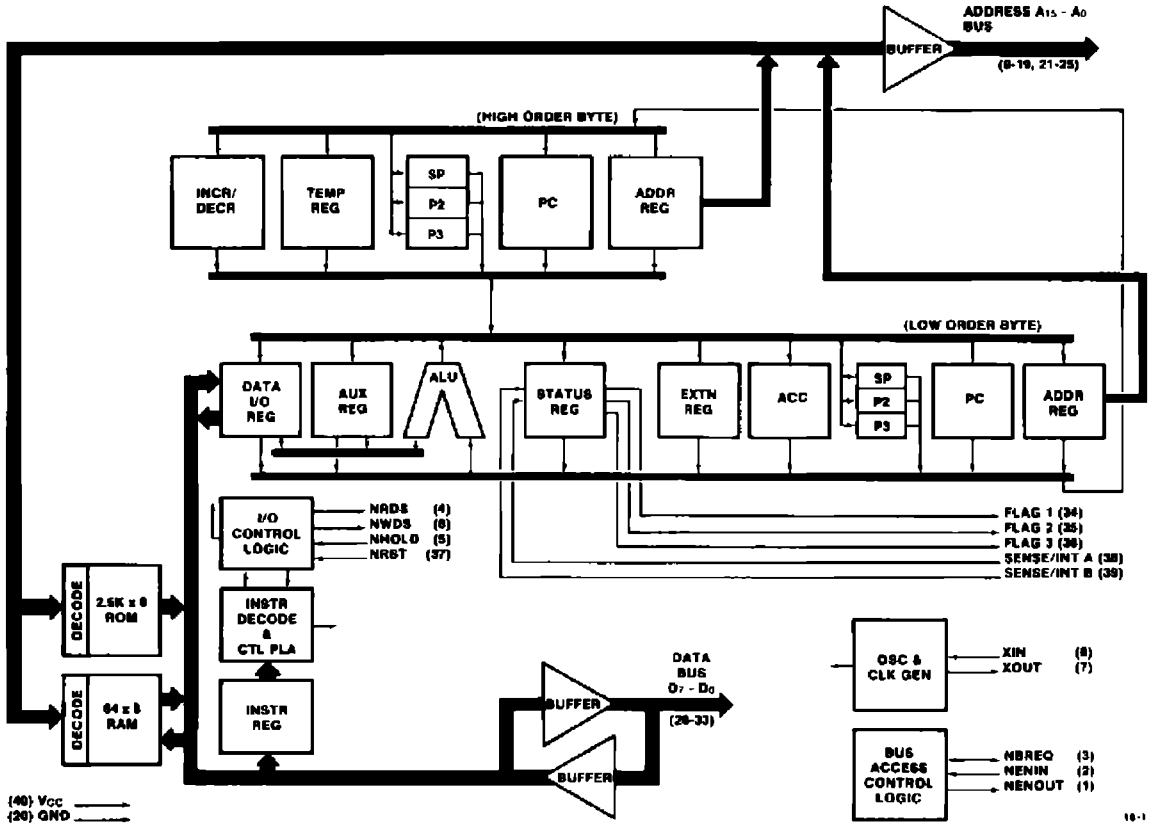


FIGURE 5. INS8070-Series Block Diagram

INSTRUCTION SET

The INS870X instruction set consists of 192 instructions which can be grouped into nine general catagories;

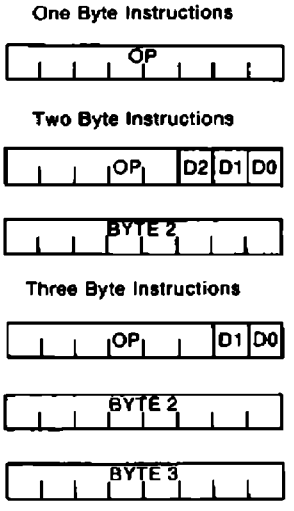
1) Register Load/Store Instructions.
2) Register to Register Exchange Instructions.
3) Arithmetic Operations: Add, Subtract, Multiply and Divide.
4) Boolean Operations: AND, OR, Exclusive OR.
5) Shift and Rotate Instructions.
6) Stack PUSH/POP Instructions.
7) Program Counter Modification Instructions.
8) Text Processing Instructions.
9) Miscellaneous Instructions.

A listing of the complete instruction set is provided in *Figure 6* (see pages 11, 12). In this figure, the symbol OP refers to the hexadecimal OP code for each instruction. The symbol BY refers to the number of memory bytes required for each instruction. The symbol RW refers to the number of READ plus WRITE operations required to execute each instruction. Note that OP, BY and RW depend upon the addressing mode specified.

The symbol MC in *Figure 6*, refers to the number of microcycles required to execute each instruction. (One microcycle equals 1 μsec when the processor is operating at an oscillator frequency of 4 MHz.) The numbers appearing under the symbol MC apply to the case where the op code and all operands are fetched/stored from internal ROM or RAM. When external memory is accessed additional microcycles must be added, because external memory I/O requires an additional microcycle per READ or WRITE operation. In addition to data READ and WRITE, external READ operations to fetch the op code and displacement (if required) must be taken into consideration.

INSTRUCTION FORMAT

All INS807X instructions fall into three catagories: single, double, and triple byte. *Figure 6* indicates the basic format for all three instruction catagories.



OP completely specifies the operation to be performed. No further memory references are required.

OP specifies the operation to be performed. The addressing mode is determined by bits D0, D1 and D2. For all addressing modes except immediate, BYTE 2 represents a displacement value between $+127_{10}$ and -128_{10} which is used to calculate the effective memory address. When the immediate mode is specified. BYTE 2 contains data.

If OP specifies an operation in which D0 and D1 specify a pointer (PC, SP, P2 or P3), bytes 2 and 3 represent a 16-bit address.

If OP specifies an operation which does not involve a pointer, bytes 2 and 3 represent 16 bits of immediate data. Byte 2 is the least significant byte, and Byte 3 is the most significant byte.

FIGURE 6. Instruction Byte Formats

SPECIALIZED INSTRUCTIONS

NOTE: *The number of microcycles (MC) and read/write (RW) operations for the SSM and BND instructions are not listed in the instruction set because the values of these numbers are data dependent.*

Search and Skip If Character Matched (SSM) Instruction

The SSM instruction searches memory for a character that matches the one present in the accumulator, using indexed addressing. The address where the search begins may be specified by pointers P2 or P3. The number of memory locations searched is 256. The search is conducted by using the specified pointer to indicate the address of the character to be tested. The pointer is automatically incremented by 1 after each test. If a match is found within the allowable 256 searches, the next two program bytes will be skipped. In this case, the instruction is concluded with the specified pointer pointing to an address one greater than that for which the match occurred. If a match is not found within the allowable 256 searches, the next instruction in sequence will be executed. In this case, the specified pointer will have been incremented by one less than the number of searches allowed (255).

Branch If Not Digit (BND) Instruction

The two byte BND instruction tests the ASCII character present in the accumulator to determine whether or not it represents the ASCII equivalent of a decimal digit (30_{16} - 39_{16}). If it does not, a PC relative branch is executed using the second byte of the instruction as displacement. In this case, the contents of the accumulator are not altered. If the accumulator does contain the ASCII equivalent of a decimal digit, the accumulator contents are converted to the 8-bit binary equivalent of the decimal digit (00_{16} - 09_{16}).

One Byte Call Instructions

The CALL and JSR instructions are similar in that both allow the user to invoke a subroutine. A major difference, however, is that the CALL instruction occupies only one byte, whereas the JSR instruction occupies three bytes. The CALL instruction utilizes indirect addressing and allows the user to invoke up to 16 subroutines. Each of the 16 subroutines may be of arbitrary length and located anywhere in memory. The two byte pointer address for each subroutine is stored in two consecutive memory locations, starting at 32_{10} (20_{16}). The values of the subroutine starting addresses stored in these indirect locations should be one less than the memory location where the subroutine actually begins. Subroutines invoked via a one byte CALL are exited in the normal manner using a RET instruction.

Multiply Instruction

The INS807X multiplication instruction (MULT) multiplies a signed 16-bit multiplicand by a signed 16-bit multiplier, generating a signed 32-bit product. The signed 16-bit multiplicand may be positive or negative and is stored in the double byte accumulator EA. (EA = the concatenation of register E and A.) The signed 16-bit multiplier must be positive and is stored in the 16-bit register T. The signed 32-bit product is stored in the EA and T registers, replacing the original multiplier and multiplicand. The most significant 16 bits of the

product are stored in the EA register, and the least significant 16 bits of the product are stored in the T register. In shorthand notation, the multiplication instruction may be written as follows:

$$(EA) \leftarrow (EA) * (T)_{31:16}$$

$$(T) \leftarrow (EA) * (T)_{15:0}$$

NOTE: *The subscripts represent bit positions of the product.*

The multiplication instruction is executed in only 37 micro-seconds at an oscillator frequency of 4 MHz.

Divide Instruction

The INS807X divide instruction divides a 16-bit dividend by a 16-bit divisor, generating a 16-bit quotient. Since the divide instruction is an integer divide, the magnitude of the dividend must be greater than the magnitude of the divisor, which must be non-zero. The 16-bit dividend can be an unsigned number in the range 0_{10} - $65,535_{10}$ or a signed positive number in range 0_{10} - $32,767_{10}$. The divisor can be an unsigned number in the range 0_{10} - $32,767_{10}$ or a signed, positive number in the range 0_{10} - $32,767_{10}$. Thus, the divide instruction will divide a signed dividend by a signed divisor, when both quantities are positive. It will also divide an unsigned dividend in the range 0_{10} - $65,535_{10}$ by an unsigned divisor in the range 0_{10} -$32,767_{10}$.

Before division can be performed, a 16-bit dividend must be stored in the EA register and a 16-bit divisor must be stored in the T register. After division has been executed, a 16-bit quotient is stored in the EA register, replacing the original dividend; the divisor which has been stored in the T register is lost.

The division instruction is executed in only 42 microseconds at an oscillator frequency of 4 MHz.

Memory Allocation

Figure 7 shows INS807X memory allocation, (decimal and hexadecimal address) for the National Semiconductor 70-Series family.

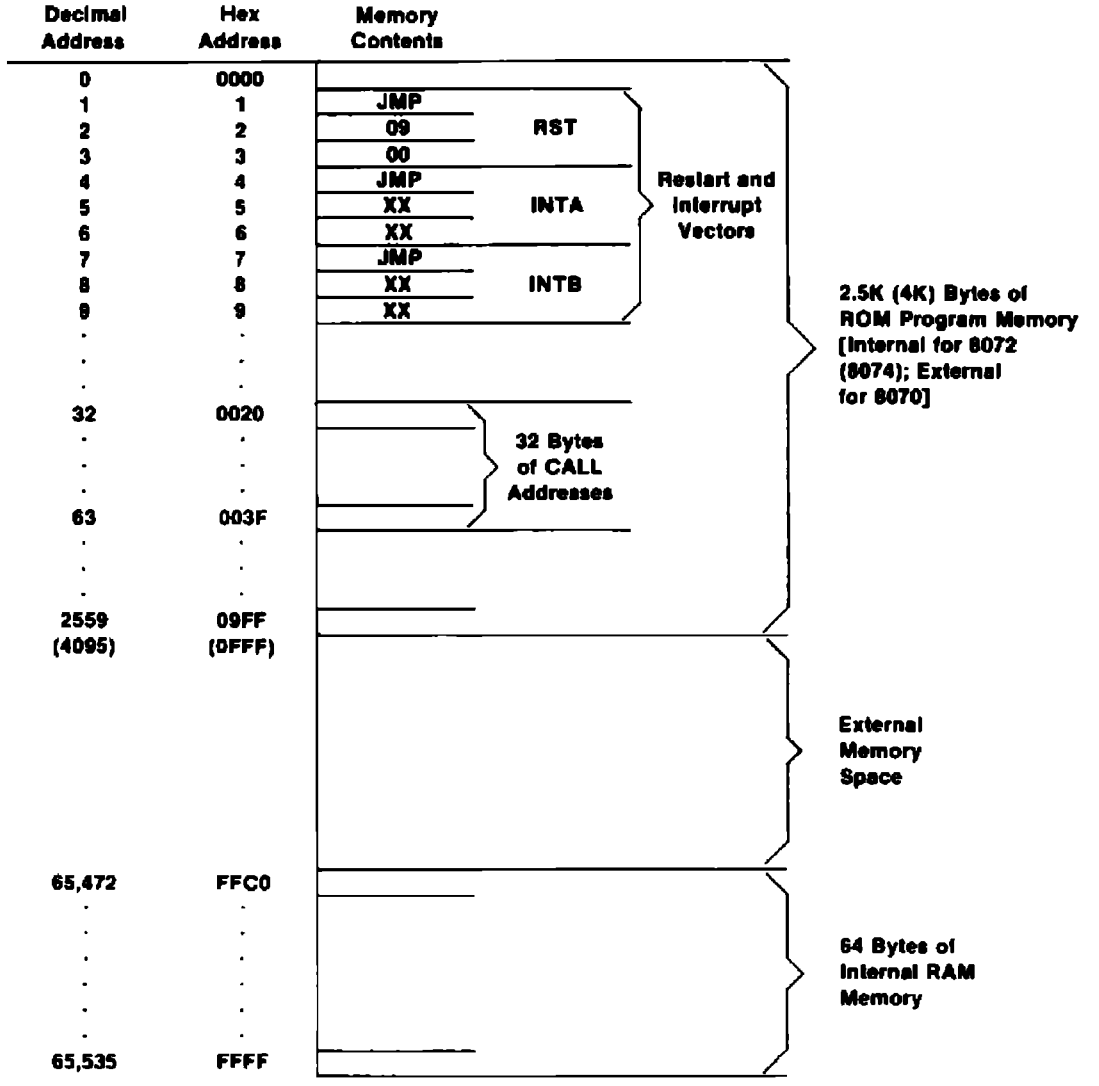


FIGURE 7. INS807X Memory Allocation

INITIALIZATION

The recommended power-up initialization sequence for the INS807X is as follows:

1. Apply power (GND and V_{CC}) and set NRST low. When NRST goes low the following occurs:

   a) Any in-process operations are automatically aborted.

   b) The READ/WRITE (NRDS/NWDS) strobes, address bus and data bus are set to the TRI-STATE (high Z) condition.

   c) The Program Counter (PC), Stack Pointer (SP), and Status Register (S) are automatically cleared.

2. Ample time should be allowed for the oscillator and internal clocks to stabilize. The minimum time that NRST must remain low is 4 T_C, where T_C is the period of one microcycle. (NOTE: *For a crystal controlled oscillator, 250 mS are typically required for crystal warmup and stabilization.*)

3. To commence program execution, NRST must be set high. The first instruction will be fetched from location 0001_{16}. This will occur within 13 T_C after NRST has gone high.

Normal program execution will continue as long as NRST remains high. In order to allow this signal to have slow rise/fall times without any adverse affects, NRST is buffered by a T^2L compatible Schmitt trigger. This allows the inexpensive RC power-on reset circuit of *Figure 8* to be used to initialize the INS807X.

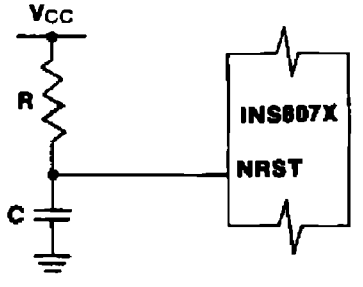


FIGURE 8. Reset Implementation

Interfacing With External Peripherals

Data are transferred on the INS807X external data bus during execution of external memory reference instructions. This enables the INS807X to use its entire memory reference class of instructions, including arithmetic and logic, to implement input/output (I/O) to external peripherals. Since all memory reference instructions are available for peripheral I/O, efficient service routines can be written that execute faster and require fewer memory bytes.

The INS807X can implement serial I/O efficiently via its two sense inputs (SA/INTA, SB/INTB) and three flag outputs (F1, F2, F3). Using these, two independent serial I/O channels can be implemented under software control using one sense input and one flag output per channel. Devices which can use this feature include teletypewriters, terminals, and modems.

APPLICATIONS

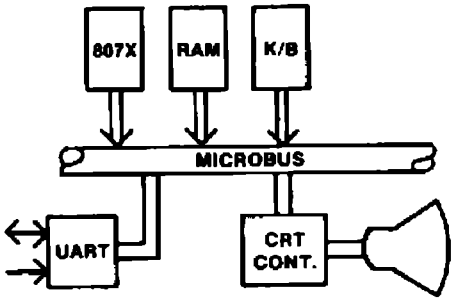
The 70-Series family of microprocessors is especially useful for the following applications.

Terminals, personal computers and Viewdata sets. (These three applications contain identical requirements for data display via a CRT.) System cost is drastically reduced by simplified buffering and synchronization requirements between the display controller, system RAM and CPU. Typically a dumb terminal may be implemented with less than 18 IC's.

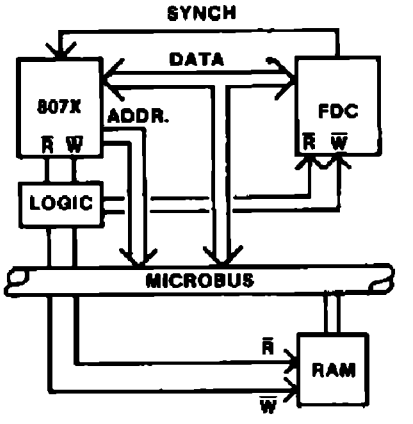
Intelligent ROMs - The INS807X system shown fulfills two functions: (1) generate addresses for the floppy disk controller (2) control floppy disk controller. This provides a system that does not require a DMA controller or a processor-specific firmware package for the file processing. This system therefore releases more memory for user programs and allows increased flexibility.

Multiprocessing Systems - Through the use of the on-board executable RAM, and by taking advantage of the high-speed 16-bit arithmetic operations, real-time solutions of polynomials become extremely simple and economical.

- TERMINAL
- PERSONAL COMPUTERS
- VIEWDATA

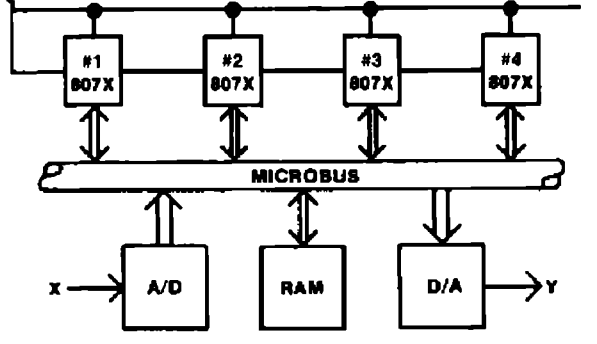


- INTELLIGENT ROMS



- MULTIPROCESSING SYSTEMS
- HIGH SPEED LOW COST REAL TIME

SOLUTION OF POLYNOMIALS



$Y = AX_n + BX_{n-1} = CX_{n-2} + DX_{n-3}$ etc.

INPUT MEDIUM:

EPROM

PAPER TAPE

EPROM:

2716 Total No. of EPROMS

2708

IMPORTANT - EPROM LABELLING

Only one customer program may be included in a single order. The following method must be used to identify the EPROMs comprising a program.

a) The EPROMs used for storing a custom program are designated as shown:

2716:	BLOCK A	0-2047
	BLOCK B	2048 - 4095
2708:	BLOCK A	0 - 1023
	BLOCK B	1024 - 2047
	BLOCK C	2048 - 3071
	BLOCK D	3072 - 4095

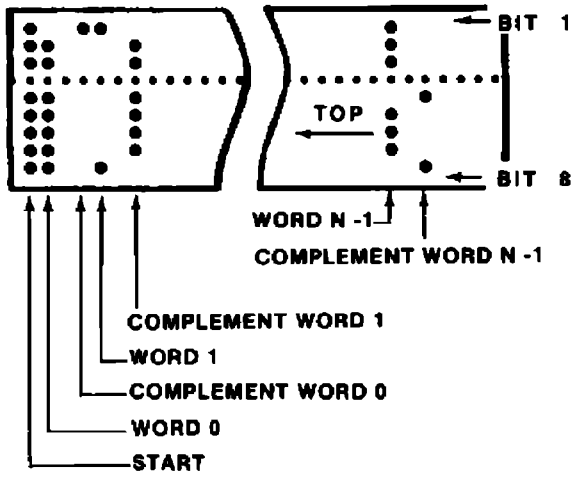
b) All EPROMs must be labelled (stickers, paint, etc) with this block designation plus a customer identification number. For example a 4K custom program contained in two 2716's would be labelled:

Customer I.D. A	Customer I.D. B
0-2047	2048-4095

Paper Tape

Tapes may only be submitted in binary complement format. The following information should be written on the paper tape.

Company Name
Company I.D. No.
NSC Part No.
A Punch = ("1" or "0")
This is logic (POS or NEG)



NOTE 1: Tape must be blank except for the data words.
NOTE 2: Tape must start with a ruboul character.
NOTE 3: Data is comprised of two words, the first being the actual Data and the second being the complement of the data.

Binary Complement Format

Verification

The customer can use software (the listing) or hardware (EPROMs) to verify the program. He will receive a verification listing and a set of EPROMs programmed and tested with tapes generated by the NSC Mask Programming System (MPS). He will be asked for a GO/NO GO response within a week after receipt of the listing and EPROMs.

Verification Listing

The verification listing has five sections:

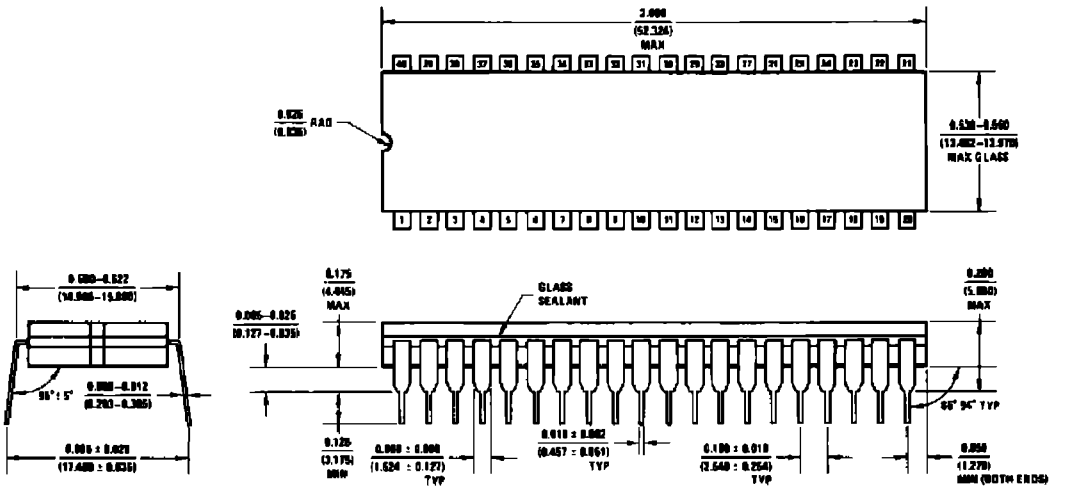
1. A cover sheet with provision for "STOP, DO NOT PROCEED" or "VERIFICATION CERTIFIED" signatures.
2. A description of the logic designations and assumptions used to process the data.
3. A listing of the data submitted by the customer.
4. An error summary.
5. A definition of the standard logic definitions for the ROM and the reduced form of the data. This list shows the output word corresponding to each address coded three ways - binary, octal, and hexadecimal.

Ordering Information for Custom Programmed Parts

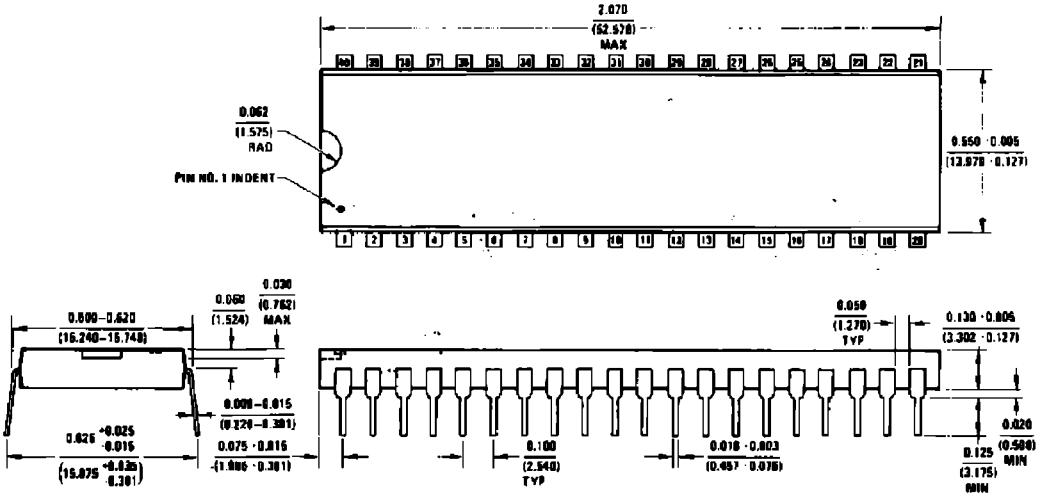
The following information must be submitted with each customer microcomputer program. An order w ll not be processed unless it is accompanied by this information. This form acts as a Traveler from Customer through Customer Service to ROM programming. Please retain a copy of this form to compare against the verification listing. The form will be sent back to the customer by Customer Service.

	National Microcomputer Part Number
	ROM Letter Code (National Use Only)
Name	Date
Address	Customer Print or I.D. No.
City State Zip	Purchase Order No.
Telephone ( )	Name of person National can contact (Print)
Authorized Signature	Date

Physical Dimensions



Ceramic Dual-In-Line Package [Cerdip (J)]
Order Number INS8070J or INS8072XXX/J



Plastic Dual In-Line Package (N)
Order Number INS8070 or INS8072XXX/N

National Semiconductor Corporation
2900 Semiconductor Drive
Santa Clara, California 95051
Tel (408) 737-5000
TWX (910) 339-9240

National Semiconductor GmbH
Eisenheimerstrasse 61/II
8000 Munchen 21
West Germany
Tel (089) 576091
Telex 05-22772

NS International Inc., Japan
Miyake Building
1-9 Yotsuya, Shinjuku-ku 160
Tokyo, Japan
Tel (03) 355-3711
TWX 232-2015 NSCJ J

National Semiconductor (Hong Kong) Ltd
8th Floor
Cheung Kong Electronic Bldg
4 Hing Yip Street
Kwun Tong
Kowloon, Hong Kong
Tel 3-899235
Telex 73866 NSEHK HX
Cable NATSEMI

NS Electronics Do Brasil
Avda Brigadeiro Faria Lima 844
11 Andar Conjunto 1104
Jardim Paulistano
Sao Paulo, Brasil
Telex
1121008 CABINE SAO PAULO

NS Electronics Pty. Ltd.
Cnr Stud Rd & Mtn Highway
Bayswater, Victoria 3153
Australia
Tel 03-729-6333
Telex 32096