OKI semiconductor

MSM80C88RS/GS

8-BIT CMOS MICROPROCESSOR

GENERAL DESCRIPTION

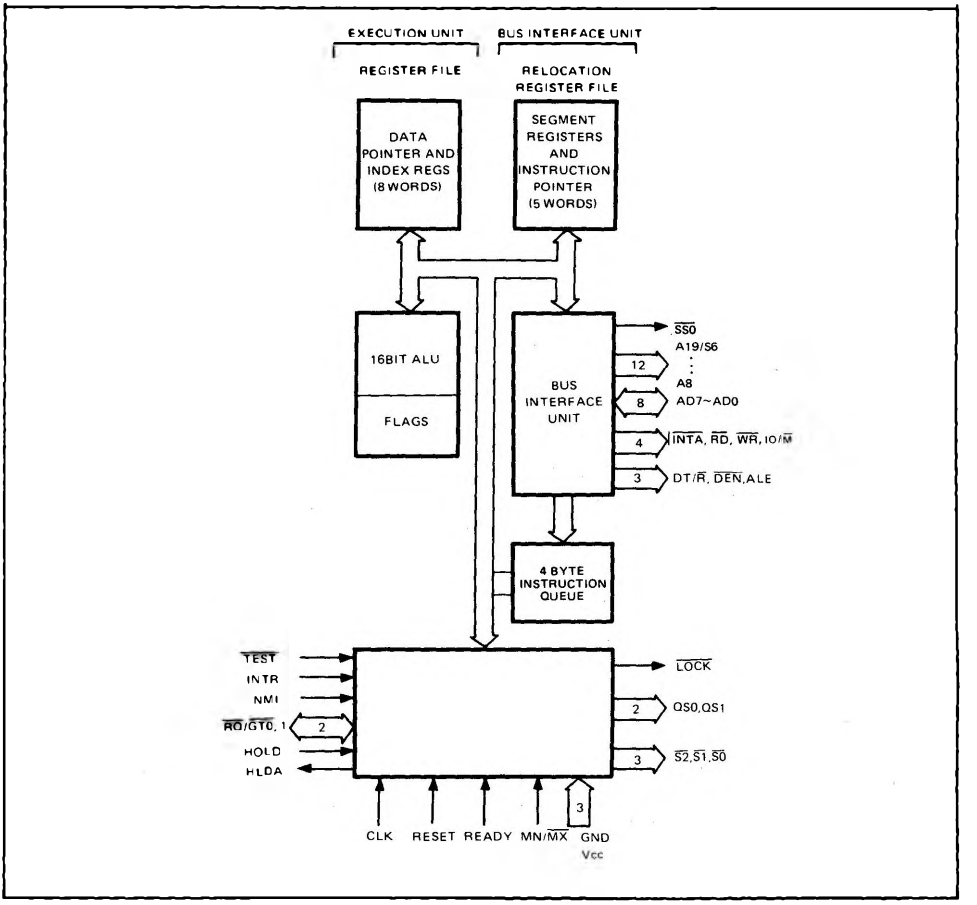
The MSM80C88 is a internal 16-bit CPU with 8-bit interface implemented in Sillicon Gate CMOS technology. It is designed with same processing speed as the NMOS8088 but with considerably less power consumption.

The processor has attributes of both 8- and 16-bit microprocessor. It is directly compatible with MSM80C86 software and MSM80C85A hardware and peripherals.

FEATURES

- 8-Bit Data Bus Interface
- 16-Bit Internal Architecture
- 1 Mbyte Direct Addressable Memory Space
- Software Compatible with MSM80C86
- Internal 14 Word by 16-bit Register Set
- 24 Operand Addressing Modes
- Bit, Byte, Word and String Operations
- 8 and 16-bit Signed and Unsigned Arithmetic Operation
- 5 MHz Clock Rate
- Low Power Dissipation (MAX 55 mA)

FUNCTIONAL BLOCK DIAGRAM



PIN CONFIGURATION

MSM80C88RS (Top View)
40 Lead Plastic DIP

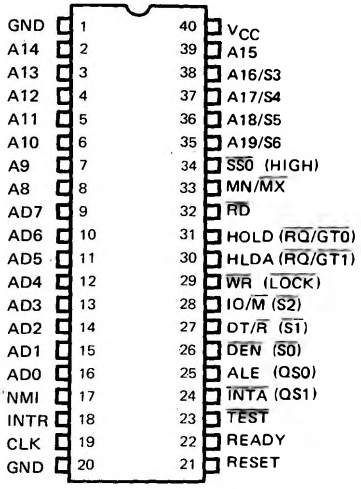


Fig. 2a MSM80C88RS

MSM80C88GS (Top View)
56 Lead Plastic Flat Package

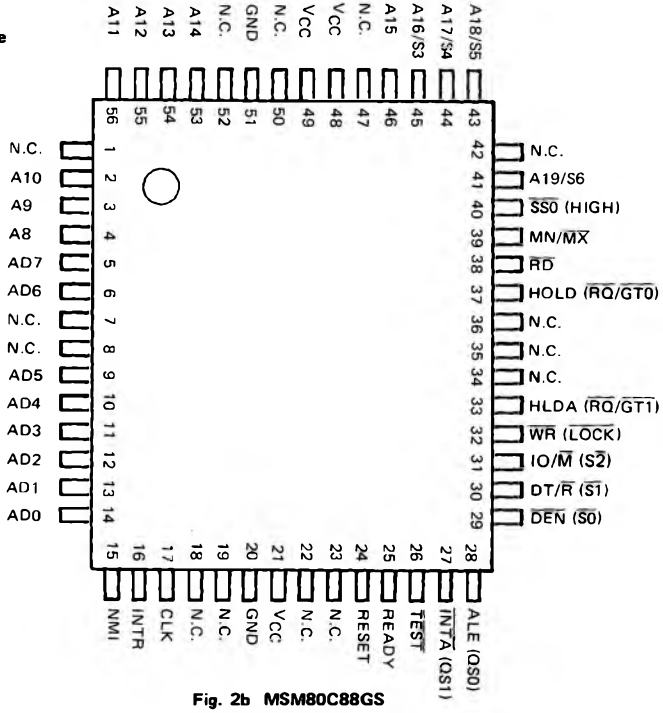


Fig. 2b MSM80C88GS

ABSOLUTE MAXIMUM RATINGS

Parameter	Symbol	Limits		Unit	Conditions
		MSM80C88RS	MSM80C88GS		
Power Supply Voltage	V_{CC}	−0.5 ~ +7		V	With respect to GND
Input Voltage	V_{IN}	−0.5 ~ V_{CC} +0.5		V	
Output Voltage	V_{OUT}	−0.5 ~ V_{CC} +0.5		V	
Storage Temperature	Tstg	−65 ~ 150		°C	−
Power Dissipation	P_D	1.0	0.7	W	Ta = 25°C

OPERATING RANGE

Parameter	Symbol	Limits	Unit
Power Supply Voltage	V_{CC}	3 ~ 6	V
Operating Temperature	T_{OP}	−40 ~ 85	°C

RECOMMENDED OPERATING CONDITIONS

Parameter	Symbol		MIN	TYP	MAX	Unit
Power Supply Voltage	V_{CC}		4.5	5.0	5.5	V
Operating Temperature	T_{OP}		−40	+25	+85	°C
"L" Input Voltage	V_{IL}		−0.5		+0.8	V
"H" Input Voltage	V_{IH}	(*1)	V_{CC}−0.8		V_{CC} +0.5	V
		(*2)	3.0		V_{CC} +0.5	V
		(*3)	2.2		V_{CC} +0.5	V

*1 Only CLK, *2 Reset & Ready, *3 Except CLK, Reset and Ready

DC CHARACTERISTICS

(V_{CC} = 4.5V to 5.5V, Ta = −40°C to +85°C)

Parameter	Symbol	MIN	TYP	MAX	Unit	Conditions
"L" Output Voltage	V_{OL}			0.4	V	I_{OL} = 2.5mA
"H" Output Voltage	V_{OH}	3.0 V_{CC}−0.4			V	I_{OH} = −2.5 mA I_{OH} = −100 μA
Input Leak Current	I_{LI}	−1.0		+1.0	μA	$0 \leq V_I \leq V_{CC}$
Output Leak Current	I_{LO}	−10		+10	μA	$0 \leq V_O \leq V_{CC}$
Operating Supply Current	I_{CC}			55	mA	TCLCL=200NS, Ta=25°C C_L = 0pF, at Reset
Input Capacitance	C_{in}			5	pF	*4
Output Capacitance	C_{out}			15	pF	*4
I/O Capacitance	$C_{I/O}$			20	pF	*4

*4. Test Conditions: a) Freq = 1 MHz.
b) Unmeasured Pins at GND.
c) V_{in} at 5.0V or GND.

A.C. CHARACTERISTICS

(V_{CC} = 4.5V to 5.5V, Ta = −40°C to +85°C)

Minimum Mode System
Timing Requirements

Parameter	Symbol	MIN	MAX	Unit
CLK Cycle Period	TCLCL	200	500	NS
CLK Low Time	TCLCH	118		NS
CLK High Time	TCHCL	65		NS
CLK Rise Time (From 1.0V to 3.5V)	TCH1CH2		10	NS
CLK Fall Time (From 3.5V to 1.0V)	TCL2CL1		10	NS
Data in Setup Time	TDVCL	30		NS
Data in Hold Time	TCLDX	10		NS
RDY Setup Time into MSM 82C84A (See Notes 1, 2)	TR1VCL	35		NS
RDY Hold Time into MSM 82C84A (See Notes 1, 2)	TCLR1X	0		NS
READY Setup Time into MSM 80C88	TRYHCH	110		NS
READY Hold Time into MSM 80C88	TCHRYX	30		NS
READY Inactive to CLK (See Note 3)	TRYLCL	−8		NS
HOLD Setup Time	THVCH	35		NS
INTR, NMI, $\overline{\text{TEST}}$ Setup Time (See Note 2)	TINVCE	30		NS
Input Rise Time (Except CLK) (From 0.8V to 2.2V)	TILIH		15	NS
Input Fall Time (Except CLK) (From 2.2V to 0.8V)	TIHIL		15	NS

Timing Responses

Parameter	Symbol	MIN	MAX	Unit
Address Valid Delay	TCLAV	10	110	NS
Address Hold Time	TCLAX	10		NS
Address Float Delay	TCLAZ	TCLAX	80	NS
ALE Width	TLHLL	TCLCH-20		NS
ALE Active Delay	TCLLH		80	NS
ALE Inactive Delay	TCHLL		85	NS
Address Hold Time to ALE Inactive	TLLAX	TCHCL-10		NS
Data Valid Delay	TCLDV	10	110	NS
Data Hold Time	TCHDX	10		NS
Data Hold Time after $\overline{\text{WR}}$	TWHDX	TCLCH-30		NS
Control Active Delay 1	TCVCTV	10	110	NS
Control Active Delay 2	TCHCTV	10	110	NS
Control Inactive Delay	TCVCTX	10	110	NS
Address Float to $\overline{\text{RD}}$ Active	TAZRL	0		NS
$\overline{\text{RD}}$ Active Delay	TCLRL	10	165	NS
$\overline{\text{RD}}$ Inactive Delay	TCLRH	10	150	NS
$\overline{\text{RD}}$ Inactive to Next Address Active	TRHAV	TCLCL-45		NS
HLDA Valid Delay	TCLHAV	10	160	NS
$\overline{\text{RD}}$ Width	TRLRH	2TCLCL-75		NS
$\overline{\text{WR}}$ Width	TWLWH	2TCLCL-60		NS
Address Valid to ALE Low	TAVAL	TCLCH-60		NS
Output Rise Time (From 0.8V to 2.2V)	TOLOH		20	NS
Output Fall Time (From 2.2V to 0.8V)	TOHOL		15	NS

NOTES: 1. Signals at MSM 82C84A shown for reference only.
2. Setup requirement for asynchronous signal only to guarantee recognition at next CLK.
3. Applies only to T2 state. (8ns into T3)

Max. mode system (Using MSM 82C88 Bus Controller)
Timing Requirement

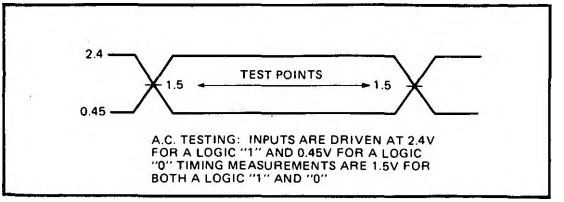
Parameter	Symbol	MIN	MAX	Unit
CLK Cycle Period	TCLCL	200	500	NS
CLK Low Time	TCLCH	118		NS
CLK High Time	TCHCL	65		NS
CLK Rise Time (From 1.0V to 3.5V)	TCH1CH2		10	NS
CLK Fall Time (From 3.5V to 1.0V)	TCL2CL1		10	NS
Data in Setup Time	TDVCL	30		NS
Data in Hold Time	TCLDX	10		NS
RDY Setup Time into MSM 82C84A (See Notes 1, 2)	TR1VCL	35		NS
RDY Hold Time into MSM 82C84A (See Notes 1, 2)	TCLR1X	0		NS
READY Setup Time into MSM 80C88	TRYHCH	110		NS
READY Hold Time into MSM 80C88	TCHRYX	30		NS
READY Inactive to CLK (See Note 3)	TRYLCL	−8		NS
Setup Time for Recognition (NM1, INTR, $\overline{\text{TEST}}$) (See Note 2)	TINVCH	30		NS
$\overline{\text{RQ}}/\overline{\text{GT}}$ Setup Time	TGVCH	30		NS
$\overline{\text{RQ}}$ Hold Time into MSM 80C88	TCHGX	40		NS
Input Rise Time (Except CLK) (From 0.8V to 2.2V)	TILIH		15	NS
Input Fall Time (Except CLK) (From 2.2V to 0.8V)	TIHIL		15	NS

Timing Responses

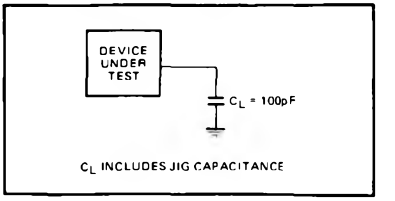
Parameter	Symbol	MIN	MAX	Unit
Command Active Delay (See Note 1)	TCLML	5	45	NS
Command Inactive Delay (See Note 1)	TCLMH	5	35	NS
READY Active to Status Passive (See Note 4)	TRYHSH		110	NS
Status Active Delay	TCHSV	10	110	NS
Status Inactive Delay	TCLSH	10	130	NS
Address Valid Delay	TCLAV	10	110	NS
Address Hold Time	TCLAX	10		NS
Address Float Delay	TCLAZ	TCLAX	80	NS
Status Valid to ALE High (See Note 1)	TSVLH		35	NS
Status Valid to MCE High (See Note 1)	TSVMCH		35	NS
CLK Low to ALE Valid (See Note 1)	TCLLH		35	NS
CLK Low to MCE High (See Note 1)	TCLMCH		35	NS
ALE Inactive Delay (See Note 1)	TCHLL	4	35	NS
MCE Inactive Delay (See Note 1)	TCLMCL		35	NS
Data Valid Delay	TCLDV	10	110	NS
Data Hold Time	TCHDX	10		NS
Control Active Delay (See Note 1)	TCVNV	5	45	NS
Control Inactive Delay (See Note 1)	TCVNX	5	45	NS
Address Float to $\overline{RD}$ Active	TAZRL	0		NS
$\overline{RD}$ Active Delay	TCLRL	10	165	NS
$\overline{RD}$ Inactive Delay	TCLRH	10	150	NS
$\overline{RD}$ Inactive to Next Address Active	TRHAV	TCLCL-45		NS
Direction Control Active Delay (See Note 1)	TCHDTL		50	NS
Direction Control Inactive Delay (See Note 1)	TCHDTH		35	NS
$\overline{GT}$ Active Delay	TCLGL	0	85	NS
$\overline{GT}$ Inactive Delay	TCLGH	0	85	NS
$\overline{RD}$ Width	TRLRH	2TCLCL-75		NS
Output Rise Time (From 0.8V to 2.2V)	TOLOH		20	NS
Output Fall Time (From 2.2V to 0.8V)	TOHOL		15	NS

NOTES: 1. Signal at MSM 82C84A and MSM82C88 shown for reference only.
2. Setup requirement for asynchronous signal only to guarantee recognition at next CLK.
3. Applies only to T2 state. (8ns into T3)
4. Applies only to T3 and wait states.

A.C. TESTING INPUT, OUTPUT WAVEFORM

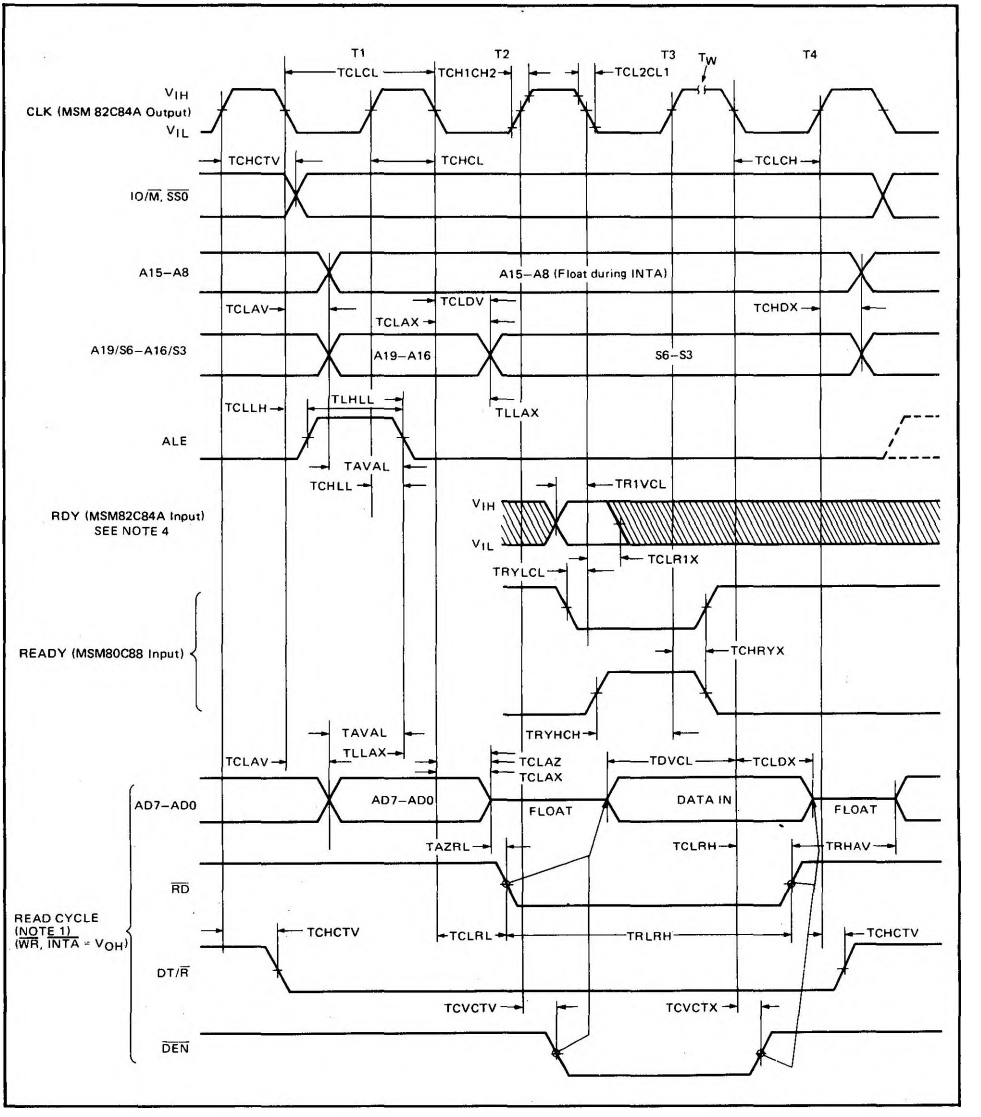


A.C. TESTING LOAD CIRCUIT

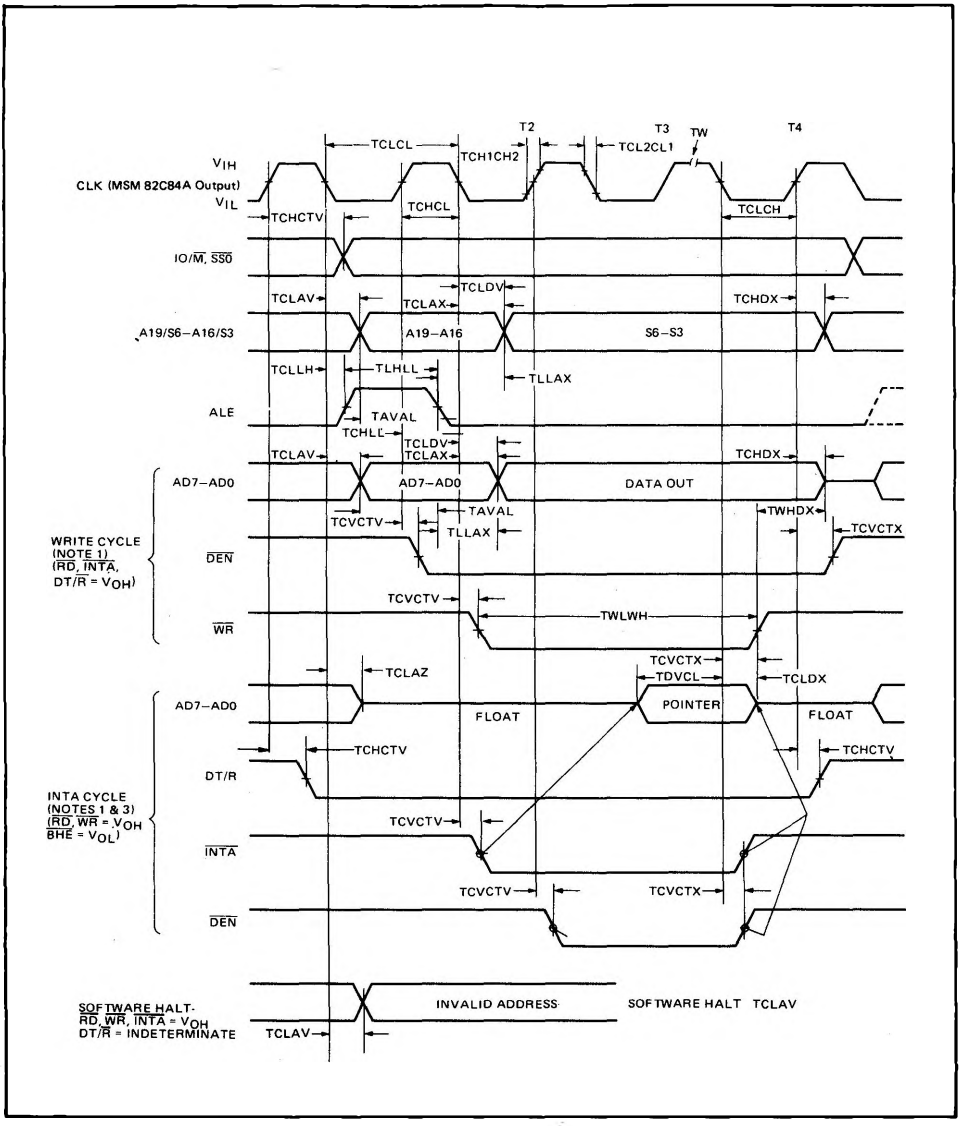


TIMING CHART

Minimum Mode



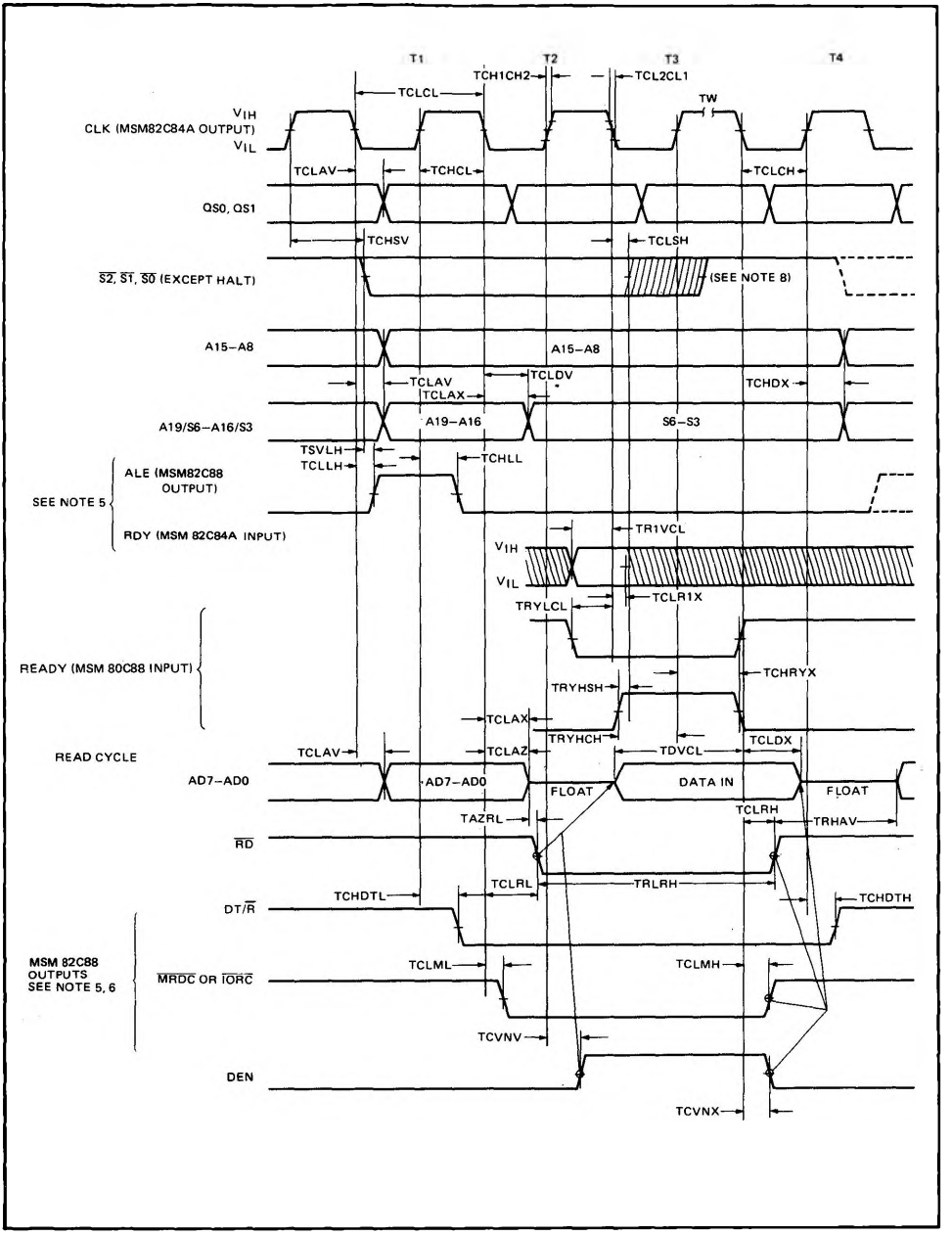
Minimum Mode (Continued)



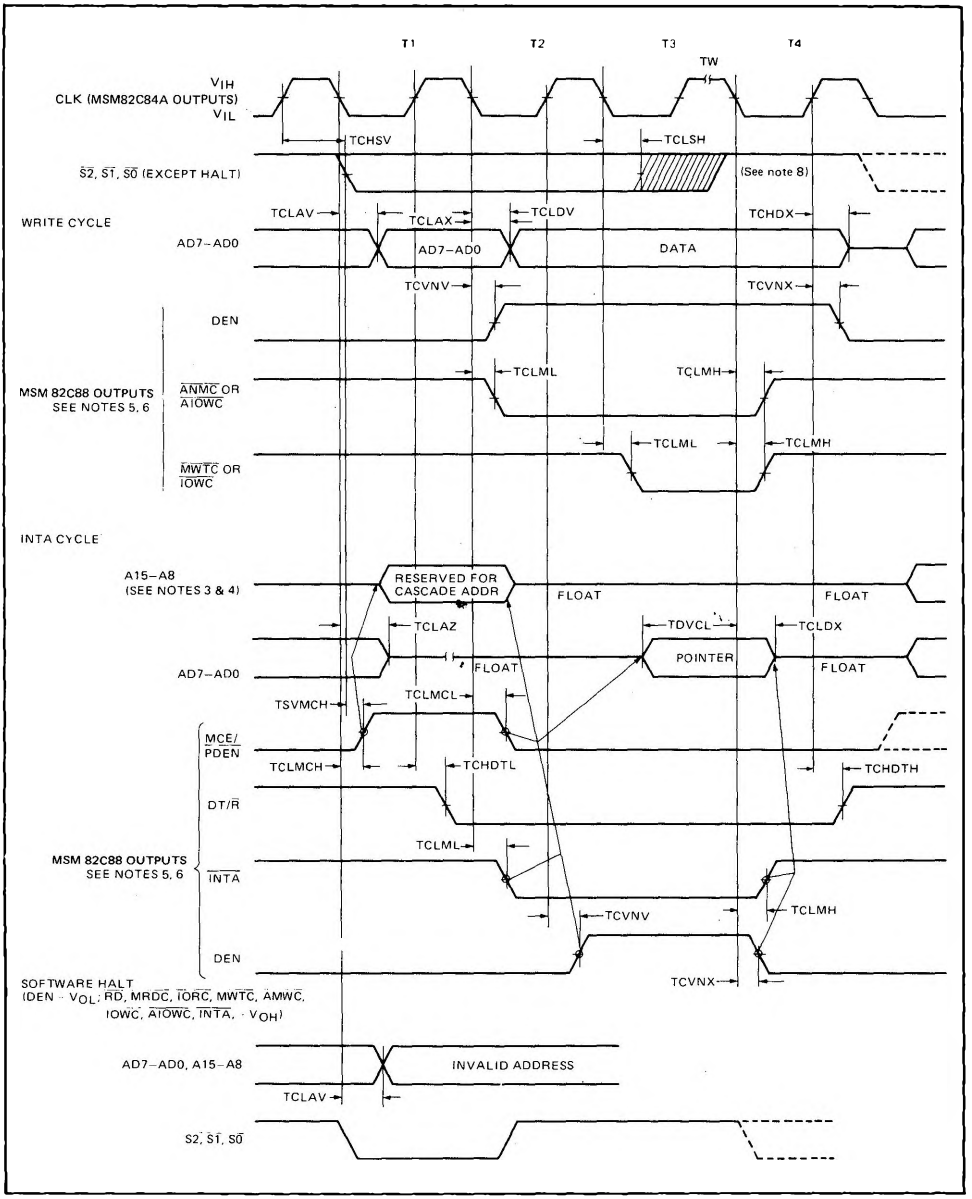
NOTES:

1. All signals switch between V_{OH} and V_{OL} unless otherwise specified.
2. RDY is sampled near the end of T2, T3, TW to determine if TW machines states are to be inserted.
3. Two INTA cycles run back-to-back. The MSM 80C88 LOCAL ADDR/DATA BUS is floating during both INTA cycles. Control signals shown for second INTA cycle.
4. Signals at MSM 82C84A shown for reference only.
5. All timing measurements are made at 1.5V unless otherwise noted.

Maximum Mode



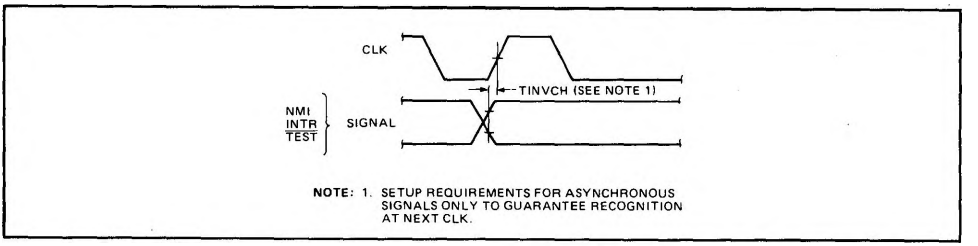
Maximum Mode (Continued)



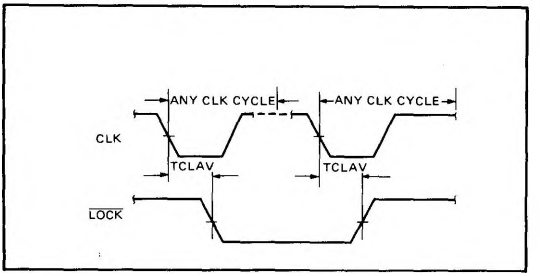
NOTES:

1. All signals switch between V_{OH} and V_{OL} unless otherwise specified
2. RDY is sampled near the end of T2, T3, TW to determine if TW machines states are to be inserted.
3. Cascade address is valid between first and second INTA cycle
4. Two INTA cycles run back-to-back. The MSM 80C88 LOCAL ADDR/DATA BUS is floating during both INTA cycles Control for pointer address is shown for second INTA cycle
5. Signal at MSM 82C84A and MSM82C88 shown for reference only.
6. The issuance of the MSM 82C88 command and control signals ($\overline{MRDC}$, $\overline{MWTC}$, $\overline{AMWC}$, $\overline{IORC}$, $\overline{IOWC}$, $\overline{AIOWC}$, $\overline{INTA}$ and DEN) lags the active high MSM 82C88 CEN.
7. All timing measurements are made at 1.5V unless otherwise noted.
8. Status inactive in state just prior to T4

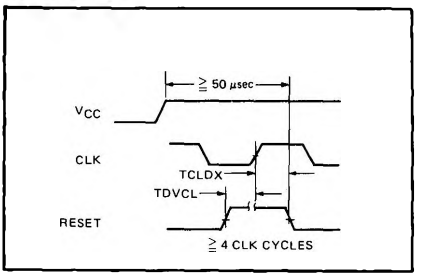
Asynchronous Signal Recognition



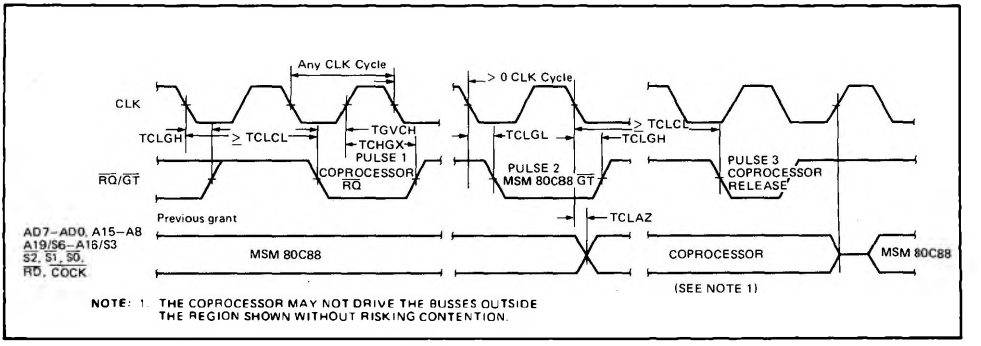
Bus Lock Signal Timing (Maximum Mode Only)



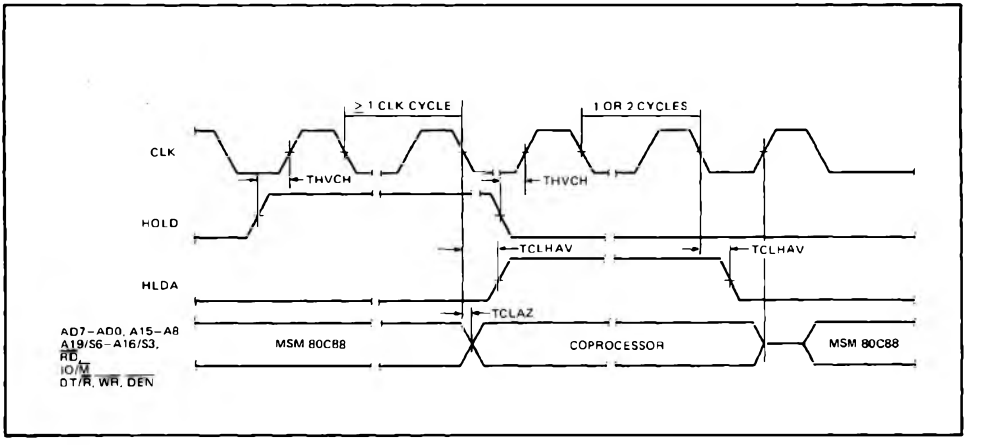
Reset Timing



Request/Grant Sequence Timing (Maximum Mode Only)



Hold/Hold Acknowledge Timing (Minimum Mode Only)



PIN DESCRIPTION

AD0–AD7
ADDRESS DATA BUS: Input/Output
These lines are multiplexed address and data bus.
These are address bus at T1 cycle and data bus at T2, T3, TW and T4 cycle.
These lines are high impedance during interrupt acknowledge and hold acknowledge.

A8–A15
ADDRESS BUS: Output
These lines are address bus of bit 8 thru 15 at all cycles.
These lines do not have to be latch by ALE signal.
These lines are high impedance during interrupt acknowledge and hold acknowledge.

A16/S3, A17/S4, A18/S5, A19/S6
ADDRESS/STATUS: Output
These are four most significant address at T1 cycle.
Accessing I/O port address, these are low at T1 Cycle.
These lines are Status lines at T2, T3, TW and T4 Cycle.
S5 indicate interrupt enable Flag.
S3 and S4 are encoded as shown.

S3	S4	Characteristics
0	0	Alternate Data
1	0	Stack
0	1	Code or None
1	1	Data

These lines are high impedance during hold acknowledge.

$\overline{RD}$
READ: Output
This line indicates that CPU is memory or I/O read cycle.
This line is read strobe signal when CPU read data from memory or I/O device.
This line is active low.
This line is high impedance during hold acknowledge.

READY
READY: Input
This line indicates to CPU that addressed memory or I/O device is ready to read or write.
This line is active high.
If the setup and hold time is out of specification illegal operation will occur.

INTR
INTERRUPT REQUEST: Input
This line is level triggered interrupt request signal which is sampled during the last clock cycle of instruction and string manipulation.
It can be internally masked by software.
This signal is active high and internally synchronized.

$\overline{TEST}$
TEST: Input
This line is examined by "WAIT" instruction.
When $\overline{TEST}$ is high, CPU enter idle cycle.
When $\overline{TEST}$ is low, CPU exit idle cycle.

NMI
NON MASKABLE INTERRUPT: Input
This line causes type 2 interrupt.
NMI is not maskable.
This signal is internally synchronized and needs 2 clock cycles pulse width.

RESET
RESET: Input
This signal causes CPU to initialize immediately.
This signal is active high and must be at least four clock cycles.

CLK
CLOCK: Input
This signal provide the basic timing for internal circuit.

MN/$\overline{MX}$
MINIMUM/MAXIMUM: Input
This signal selects CPU's operate mode.
When V_{CC} is connected. CPU operates minimum mode.
When GND is connected. CPU operates maximum mode.

V_{CC}
V_{CC}:
+3 – +6 V supplied.

GND
GROUND:

The following pin function descriptions are maximum mode only.
Other pin functions are already described.

$\overline{S0}$, $\overline{S1}$, $\overline{S2}$
STATUS: Output

These lines indicate bus status and they are used by the MSM82C88 Bus Controller to generate all memory and I/O access control signals.
These lines are high impedance during hold acknowledge.
These status lines are encoded as shown.

$\overline{S2}$	$\overline{S1}$	$\overline{S0}$	Characteristics
0 (LOW)	0	0	Interrupt acknowledge
0	0	1	Read I/O Port
0	1	0	Write I/O Port
0	1	1	Halt
1 (HIGH)	0	0	Code Access
1	0	1	Read Memory
1	1	0	Write Memory
1	1	1	Passive

$\overline{RQ}$/GT0
$\overline{RQ}$/GT1
REQUEST/GRANT: Input/Output

These lines are used for Bus Request from other device and Bus GRANT to other device.
These lines are bidirectional and active low.

$\overline{LOCK}$
LOCK: Output

This line is active low.
When this line is low, other device could not gain control of the bus.
This line is high impedance during hold acknowledge.

QS0/QS1
QUEUE STATUS: Output

These lines are Queue Status that indicate internal instruction queue status.

QS1	QS0	Characteristics
0 (LOW)	0	No Operation
0	1	First Byte of Op Code from Queue
1 (HIGH)	0	Empty the Queue
1	1	Subsequent Byte from Queue

The following pin function descriptions are minimum mode only. Other pin functions are already described.

IO/$\overline{M}$
STATUS: Output

This line selects memory address space or I/O address space.
When this line is low. CPU select memory address space and when it is high. CPU select I/O address space.
This line is high impedance during hold acknowledge.

$\overline{WR}$
WRITE: Output

This line indicates that CPU is memory or I/O write cycle.
This line is write strobe signal when CPU write data to memory or I/O device.
This line is active low.
This line is high impedance during hold acknowledge.

$\overline{INTA}$
INTERRUPT ACKNOWLEDGE: Output

This line is read strobe signal for interrupt acknowledge cycle.
This line is active low.

ALE
ADDRESS LATCH ENABLE: Output

This line is used for latching address into MSM82C12 address latch. It is positive pulse and trailing edge is used to strobe the address. This line is never floated.

DT/$\overline{R}$
DATA TRANSMIT/RECEIVE: Output

This line is used for control a direction of bus transceiver.
When this line is high. CPU transmit data, and when it is low. CPU receive data.
This line is high impedance during hold acknowledge.

$\overline{DEN}$
DATA ENABLE: Output

This line is used for control an output enable of bus transceiver.
This line is active low. This line is high impedance during hold acknowledge.

HOLD
HOLD REQUEST: Input

This line is used for Bus Request from other device.
This line is active high.

HLDA
HOLD ACKNOWLEDGE: Output

This line is used for Bus Grant to other device.
This line is active high.

$\overline{SS0}$
STATUS: Output

This line is logically equivalent to $\overline{S0}$ in the maximum mode.

FUNCTIONAL DESCRIPTION

GENERAL OPERATION

The internal function of MSM80C88 consist of Bus Interface Unit (BIU) and Execution Unit (EU). These units operate mutually but perform as separate processor.

BIU performs instruction fetch and queueing, operand fetch, DATA read and write address relocation and basic bus control. By instruction pre-fetch while waiting for decoding and execution of instruction. CPU's performance is increased. Up to 4-bytes of instruction stream can be queued.

EU receives pre-fetched instructions from BIU queue, decodes and executes instruction and provided un-relocated operand address to BIU.

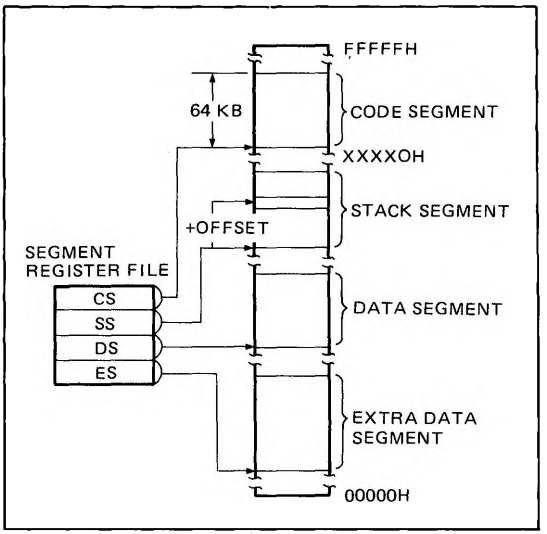
MEMORY ORGANIZATION

MSM80C88 has 20-bit address to memory. Each address has 8-bit data width. Memory is organized 00000H to FFFFFH and is logically divided four segment, code, data, extra data and stack segment. Each segment contain up to 64 Kbytes and locate on 16-byte boundary. (Fig. 3a)

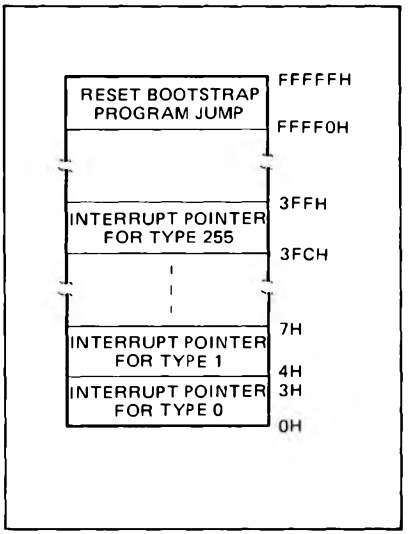
All memory references are made relative to segment register which is according to select rule. Memory location FFFF0H is start address after reset and 00000H through 003FFH are reserved for interrupt pointer, where 256 types interrupt pointer are there.

Each interrupt type has 4-byte pointer element consist of 16-bit segment address and 16-bit offset address.

Memory Organization



Reserved Memory Locations



Memory Reference Need	Segment Register Used	Segment Selection Rule
Instructions	CODE (CS)	Automatic with all instruction prefetch.
Stack	STACK (SS)	All stack pushes and pops. Memory references relative to BP base register except data references.
Local Data	DATA (DS)	Data references when relative to stack, destination of string operation, or explicitly overridden.
External (Global) Data	EXTRA (ES)	Destination of string operations: Explicitly selected using a segment override.

MINIMUM AND MAXIMUM MODES

MSM80C88 has two system mode: minimum and maximum mode. As using maximum ode, it is easy to organize multi-CPU system with MSM82C88 Bus Controller which general bus control signal generate.

As using minimum mode, it is easy to organize simple system by generating bus control signal itself. MN/$\overline{MX}$ is mode select pin. Definition of 24–31, 34 pin changes depend on the MN/$\overline{MX}$ pin.

BUS OPERATION

MSM80C88 has a time multiplexed address and data bus. If non-multiplexed bus is desired for system, it is only to add the address latch.

CPU bus cycle consists of at least four clock cycles. T1, T2, T3 and T4. (Fig. 4)

The address output occurs during T1 and data transfer occurs during T3 and T4. T2 is used for changing the direction of the bus at read operation. When the device which is accessed by CPU is not ready to data transfer and send to CPU "NOT READY". TW cycles are inserted between T3 and T4.

When bus cycle is not needed. T1 cycles are inserted between nus cycles for internal execution. At T1 cycle ALE signal is output from CPU or MSM82C88 depending in MN/MX. At the trailing edge of ALE, a valid address may be latched. Status bits S0, S1 and S2 are used in maximum mode, by the bus controller to recognize the type of bus operation according to the following table.

$\overline{S2}$	$\overline{S1}$	$\overline{S0}$	Characteristics
0 (LOW)	0	0	Interrupt acknowledge
0	0	1	Read I/O
0	1	0	Write I/O
0	1	1	Halt
1 (HIGH)	0	0	Instruction Fetch
1	0	1	Read Data from Memory
1	1	0	Write Data to Memory
1	1	1	Passive (no bus cycle)

Status bits S3 through S6 are multiplexed with A16–A19, and therefore there are valid during T2 through T4. S3 and S4 indicate which segment register was selected on the bus cycle, according to the following table.

S4	S3	Characteristics
0 (LOW)	0	Alternate Data (Extra Segment)
0	1	Stack
1 (HIGH)	0	Code or None
1	1	Data

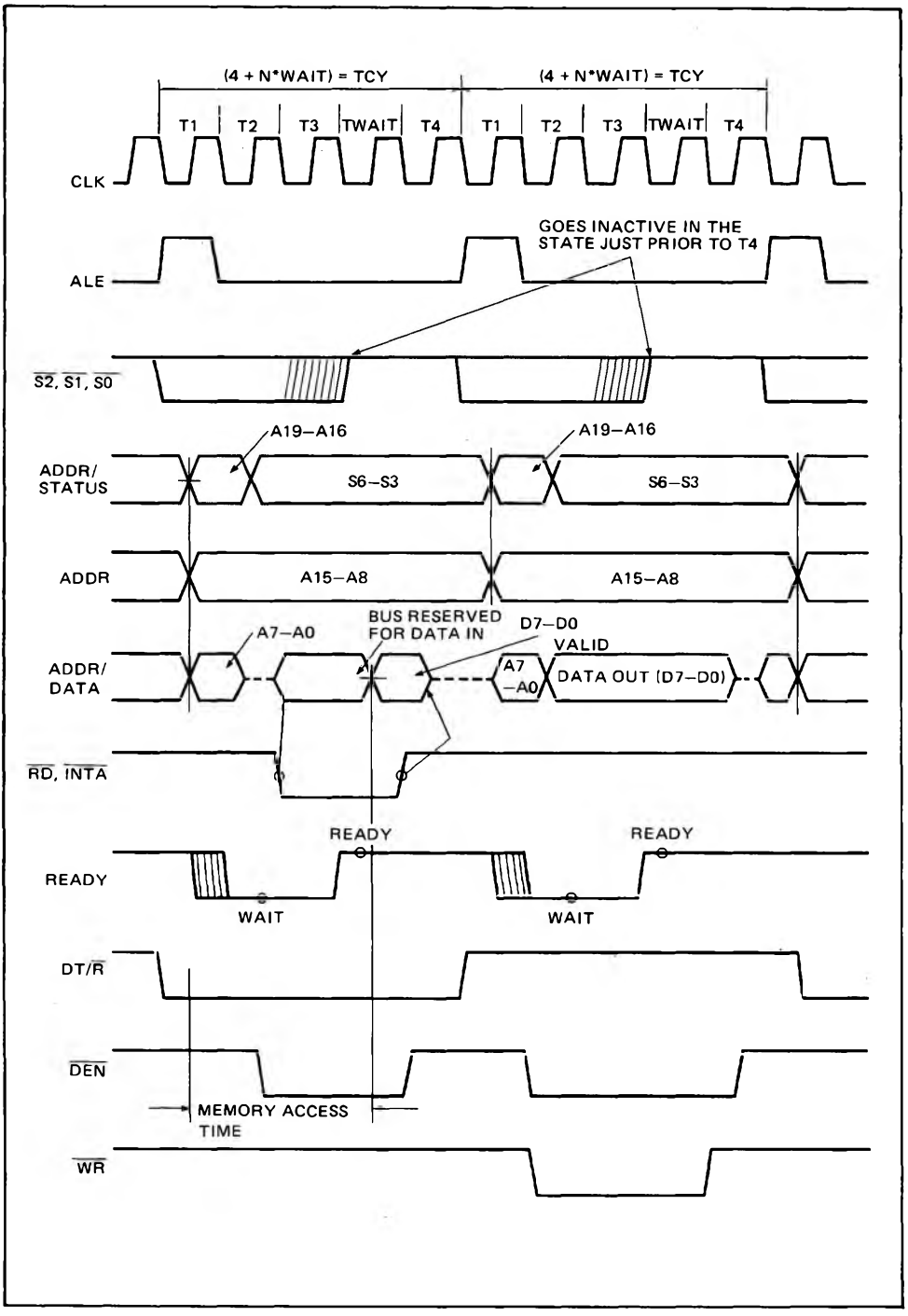
S5 indicates interrupt enable Flag.

I/O ADDRESSING

MSM80C88 has 64 Kbyte I/O. When CPU accesses I/O device, address A0–A15 are same format as a memory access, and A16–A19 are low.

I/O ports address are same as memory.

Basic System Timing



EXTERNAL INTERFACE

RESET

CPU initialization is executed by RESET pin. MSM80C88's RESET High signal is required for greater than 4 clock cycles.

Rising edge of RESET terminate present operation immediately. Falling edge of RESET triggered internal reset sequence for approximately 10 clock cycles. After internal reset sequence is done, normal operation begin from absolute location FFFF0H.

INTERRUPT OPERATIONS

Interrupt operation is classified as software or hardware and hardware interrupt is classified as non-maskable or maskable.

Interrupt causes to a new program location which is defined interrupt pointer table, according to interrupt type. Absolute location 00000H through 003FFH is reserved for interrupt pointer table. Interrupt pointer table consist of 256-element, and each element is 4 bytes in size and corresponds to an 8 bit type number which is sent from interrupt request device during interrupt acknowledge cycle.

NON-MASKABLE INTERRUPT (NMI)

MSM80C88 has Non-maskable Interrupt (NM1) which is higher priority than maskable interrupt request (INTR).

NMI request pulse width need minimum 2 clock cycles. NMI will be serviced at the end of the current instruction or between string manipulation.

MASKABLE INTERRUPT (INTR)

MSM80C88 provides an another interrupt request (INTR) which can be masked by software. INTR is level triggered, so it must be hold until interrupt request is acknowledged.

INTR will be serviced at the end of the current instruction or between string manipulation.

INTERRUPT ACKNOWLEDGE

During the interrupt acknowledge sequence, further interrupts are disabled. Interrupt enable bit is reset by any interrupt, after Flag register is automatically pushed onto the stack. During acknowledge sequence. CPU emits the lock signal from T2 of first bus cycle to T2 of second bus cycle. At second bus cycle, byte is fetched from external device as a vector which identified the type of interrupt, and this vector is multiplied by four and used as a interrupt pointer address (INTR only).

The Interrupt Return (IRET) instruction includes a Flag pop operation which returns the original interrupt enable bit when it restores the Flag.

HALT

When Halt instruction is executed, CPU enter Halt state. Interrupt request or RESET will force the MSM80C88 out of the Halt state.

SYSTEM TIMING–MINIMUM MODE

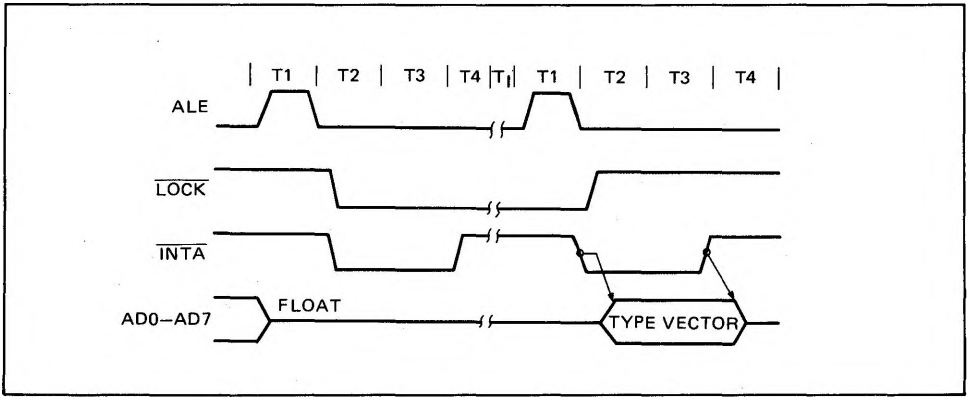
Bus cycle begins T1 with ALE signal. The trailing edge of ALE is used to latch the address. From T1 to T4 the IO/$\overline{M}$ signal indicates a memory or I/O operation. From T2 to T4, the address data bus change address bus to data bus.

The read ($\overline{RD}$), write ($\overline{WR}$), interrupt acknowledge ($\overline{INTA}$) signal causes the addressed device to enable data bus. These signal becomes active at beginning of T2 and inactive at beginning of T4.

SYSTEM TIMING–MAXIMUM MODE

At maximum mode, MSM82C88 Bus Controller is added to system. CPU sends status information to Bus Controller. Bus timing signals are generated by Bus Controller. Bus timing is almost same as minimum mode.

Interrupt Acknowledge Sequence



DATA TRANSFER

	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
MOV = Move:																																
Register/memory to/from register	1	0	0	0	1	0	d	w	mod		reg			r/m																		
Immediate to register/memory	1	1	0	0	0	1	1	w	mod	0	0	0		r/m			data								data if w = 1							
Immediate to register	1	0	1	1	w		reg				data						addr-low								addr-high							
Memory to accumulator	1	0	1	0	0	0	0	w			addr-low						addr-high															
Accumulator to memory	1	0	1	0	0	0	1	w			addr-low						addr-high															
Register/memory to segment register	1	0	0	0	1	1	1	0	mod	0	reg			r/m																		
Segment register to register/memory	1	0	0	0	1	1	0	0	mod	0	reg			r/m																		
PUSH = Push:																																
Register/memory	1	1	1	1	1	1	1	1	mod	1	1	0		r/m																		
Register	0	1	0	1	0		reg																									
Segment register	0	0	0	reg		1	1	0																								
POP = Pop:																																
Register/memory	1	0	0	0	1	1	1	1	mod	0	0	0		r/m																		
Register	0	1	0	1	1		reg																									
Segment register	0	0	0	reg		1	1	1																								
XCHG = Exchange:																																
Register/memory with register	1	0	0	0	0	1	1	w	mod		reg			r/m																		
Register with accumulator	1	0	0	1	0		reg																									
IN = Input from:																																
Fixed port	1	1	1	0	0	1	0	w			port																					
Variable port	1	1	1	0	1	1	0	w																								
OUT = Output to:																																
Fixed port	1	1	1	0	0	1	1	w			port																					
Variable port	1	1	1	0	1	1	1	w																								
XLAT = Translate byte to AL	1	1	0	1	0	1	1	1																								
LEA = Load EA to register	1	0	0	0	1	1	0	1	mod		reg			r/m																		
LDS = Load pointer to DS	1	1	0	0	0	1	0	1	mod		reg			r/m																		
LES = Load pointer to ES	1	1	0	0	0	1	0	0	mod		reg			r/m																		
LAHF = Load AH with flags	1	0	0	1	1	1	1	1																								
SAHF = Store AH into flags	1	0	0	1	1	1	1	0																								
PUSHF = Push flags	1	0	0	1	1	1	0	0																								
POPF = Pop flags	1	0	0	1	1	1	0	1																								

ARITHMETIC

ADD = Add:				
Reg./memory with register to either	0 0 0 0 0 0 d w | mod reg r/m			
Immediate to register/memory	1 0 0 0 0 0 s w | mod 0 0 0 r/m	data	data if s:w = 01	
Immediate to accumulator	0 0 0 0 0 1 0 w | data	data if w = 1		
ADC = Add with carry:				
Reg./memory with register to either	0 0 0 1 0 0 d w | mod reg r/m			
Immediate to register/memory	1 0 0 0 0 0 s w | mod 0 1 0 r/m	data	data if s:w = 01	
Immediate to accumulator	0 0 0 1 0 1 0 w | data	data if w = 1		
INC = Increment:				
Register/memory	1 1 1 1 1 1 1 w | mod 0 0 0 r/m			
Register	0 1 0 0 0 reg			
AAA = ASCII adjust for add	0 0 1 1 0 1 1 1			
DAA = Decimal adjust for add	0 0 1 0 0 1 1 1			
SUB = subtract:				
Reg./memory and register to either	0 0 1 0 1 0 d w | mod reg r/m			
Immediate from register/memory	1 0 0 0 0 0 s w | mod 1 0 1 r/m	data	data if s:w = 01	
Immediate from accumulator	0 0 1 0 1 1 0 w | data	data if w = 1		
SBB = Subtract with borrow:				
Reg./memory and register to either	0 0 0 1 1 0 d w | mod reg r/m			
Immediate from register/memory	1 0 0 0 0 0 s w | mod 0 1 1 r/m	data	data if s:w = 01	
Immediate from accumulator	0 0 0 1 1 1 0 w | data	data if w = 1		
DEC = Decrement:				
Register/memory	1 1 1 1 1 1 1 w | mod 0 0 1 r/m			
Register	0 1 0 0 1 reg			
NEG = Change sign	1 1 1 1 0 1 1 w | mod 0 1 1 r/m			
CMP = Compare:				
Register/memory and register	0 0 1 1 1 0 d w | mod reg r/m			
Immediate with register/memory	1 0 0 0 0 0 s w | mod 1 1 1 r/m	data	data if s:w = 01	
Immediate with accumulator	0 0 1 1 1 1 0 w | data	data if w = 1		
AAS = ASCII adjust for subtract	0 0 1 1 1 1 1 1			

DAS = Decimal adjust for subtract	0 0 1 0 1 1 1 1			
MUL = Multiply (unsigned)	1 1 1 1 0 1 1 w	mod 1 0 0 r/m		
IMUL = Integer multiply (signed)	1 1 1 1 0 1 1 w	mod 1 0 1 r/m		
AAM = ASCII adjust for multiply	1 1 0 1 0 1 0 0	0 0 0 0 1 0 1 0		
DIV = Divide (unsigned)	1 1 1 1 0 1 1 w	mod 1 1 0 r/m		
IDIV = Integer divide (signed)	1 1 1 1 0 1 1 w	mod 1 1 1 r/m		
AAD = ASCII adjust for divide	1 1 0 1 0 1 0 1	0 0 0 0 1 0 1 0		
CBW = Convert byte to word	1 0 0 1 1 0 0 0			
CWD = Convert word to double word	1 0 0 1 1 0 0 1			

LOGIC

NOT = Invert	1	1	1	1	0	1	1	w	mod	0	1	0	r/m					
SHL/SAL = Shift logical/arithmetic left	1	1	0	1	0	0	v	w	mod	1	0	0	r/m					
SHR = Shift logical right	1	1	0	1	0	0	v	w	mod	1	0	1	r/m					
SAR = Shift arithmetic right	1	1	0	1	0	0	v	w	mod	1	1	1	r/m					
ROL = Rotate left	1	1	0	1	0	0	v	w	mod	0	0	0	r/m					
ROR = Rotate right	1	1	0	1	0	0	v	w	mod	0	0	1	r/m					
RCL = Rotate left through carry	1	1	0	1	0	0	v	w	mod	0	1	0	r/m					
RCR = Rotate right through carry	1	1	0	1	0	0	v	w	mod	0	1	1	r/m					
AND = And:																		
Reg./memory and register to either	0	0	1	0	0	0	d	w	mod			reg	r/m					
Immediate to register/memory	1	0	0	0	0	0	0	w	mod	1	0	0	r/m	data		data if w = 1		
Immediate to accumulator	0	0	1	0	0	1	0	w				data		data if w = 1				
TEST = And function to flags, no result:																		
Register/memory and register	1	0	0	0	0	1	0	w	mod			reg	r/m					
Immediate data and register/memory	1	1	1	1	0	1	1	w	mod	0	0	0	r/m	data		data if w = 1		
Immediate data and accumulator	1	0	1	0	1	0	0	w				data		data if w = 1				
OR = Or:																		
Reg./memory and register to either	0	0	0	0	1	0	d	w	mod			reg	r/m					
Immediate to register/memory	1	0	0	0	0	0	0	w	mod	0	0	1	r/m	data		data if w = 1		
Immediate to accumulator	0	0	0	0	1	1	0	w				data		data if w = 1				
XOR = Exclusive or:																		
Reg./memory and register to either	0	0	1	1	0	0	d	w	mod			reg	r/m					
Immediate to register/memory	1	0	0	0	0	0	0	w	mod	1	1	0	r/m	data		data if w = 1		
Immediate to accumulator	0	0	1	1	0	1	0	w				data		data if w = 1				

STRING MANIPULATION

REP = Repeat	1	1	1	1	0	0	1	z										
MOVS = Move byte/word	1	0	1	0	0	1	0	w										
CMPS = Compare byte/word	1	0	1	0	0	1	1	w										
SCAS = Scan byte/word	1	0	1	0	1	1	1	w										
LODS = Load byte/word to AL/AX	1	0	1	0	1	1	0	w										
STOS = Store byte/word from AL/AX	1	0	1	0	1	0	1	w										

CJMP = Conditional JMP											
JE/JZ = Jump on equal/zero	0	1	1	1	0	1	0	0	disp		
JZ/JNGE = Jump on less/not greater or equal	0	1	1	1	1	1	0	0	disp		
JLE/JNG = Jump on less or equal/not greater	0	1	1	1	1	1	1	0	disp		
JB/JNAE = Jump on below/not above or equal	0	1	1	1	0	0	1	0	disp		
JBE/JNA = Jump on below or equal/not above	0	1	1	1	0	1	1	0	disp		
JP/JPE = Jump on parity/parity even	0	1	1	1	1	0	1	0	disp		
JO = Jump on over flow	0	1	1	1	0	0	0	0	disp		
JS = Jump on sign	0	1	1	1	1	0	0	0	disp		
JNE/JNZ = Jump on not equal/not zero	0	1	1	1	0	1	0	1	disp		
JNL/JGE = Jump on not less/greater or equal	0	1	1	1	1	1	0	1	disp		
JNLE/JG = Jump on not less or equal/greater	0	1	1	1	1	1	1	1	disp		
JNB/JAE = Jump on not below/above or equal	0	1	1	1	0	0	1	1	disp		
JNBE/JA = Jump on not below or equal/above	0	1	1	1	0	1	1	1	disp		
JNP/JPO = Jump on not parity/parity odd	0	1	1	1	1	0	1	1	disp		
JNO = Jump on not overflow	0	1	1	1	0	0	0	1	disp		
JNS = Jump on not sign	0	1	1	1	1	0	0	1	disp		
LOOP = Loop CX times	1	1	1	0	0	0	1	0	disp		
LOOPZ/LOOPE = Loop while zero/equal	1	1	1	0	0	0	0	1	disp		
LOOPNZ/LOOPNE = Loop while not zero/equal	1	1	1	0	0	0	0	0	disp		
JCXZ = Jump on CX zero	1	1	1	0	0	0	1	1	disp		
INT = Interrupt:											
Type specified	1	1	0	0	1	1	0	1	type		
Type 3	1	1	0	0	1	1	0	0			
INTO = Interrupt on overflow	1	1	0	0	1	1	1	0			
IRET = Interrupt return	1	1	0	0	1	1	1	1			

PROCESSOR CONTROL

CLC = Clear carry	1	1	1	1	1	0	0	0			
CMC = Complement carry	1	1	1	1	0	1	0	1			
STC = Set carry	1	1	1	1	1	0	0	1			
CLD = Clear direction	1	1	1	1	1	1	0	0			
STD = Set direction	1	1	1	1	1	1	0	1			
CLI = Clear interrupt	1	1	1	1	1	0	1	0			
STI = Set interrupt	1	1	1	1	1	0	1	1			
HLT = Halt	1	1	1	1	0	1	0	0			
WAIT = Wait	1	0	0	1	1	0	1	1			
ESC = Escape (to external device)	1	1	0	1	1	x	x	x	mod x x x r/m		
LOCK = Bus lock prefix	1	1	1	1	0	0	0	0			

CONTROL TRANSFER

	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
CALL = Call:																																
Direct within segment	1	1	1	0	1	0	0	0				disp-low								disp-high												
Indirect within segment	1	1	1	1	1	1	1	1	mod		0	1	0		r/m																	
Direct intersegment	1	0	0	1	1	0	1	0				offset-low								offset-high												
												seg-low								seg-high												
Indirect intersegment	1	1	1	1	1	1	1	1	mod		0	1	1		r/m																	
JMP = Unconditional Jump:																																
Direct within segment	1	1	1	0	1	0	0	1				disp-low								disp-high												
Direct within segment-short	1	1	1	0	1	0	1	1				disp																				
Indirect within segment	1	1	1	1	1	1	1	1	mod		1	0	0		r/m																	
Direct intersegment	1	1	1	0	1	0	1	0				offset-low								offset-high												
												seg-low								seg-high												
Indirect intersegment	1	1	1	1	1	1	1	1	mod		1	0	1		r/m																	
RET = Return from CALL:																																
Within segment	1	1	0	0	0	0	1	1																								
Within seg. adding immediate to SP	1	1	0	0	0	0	1	0				data-low								data-high												
Intersegment	1	1	0	0	1	0	1	1																								
Intersegment adding immediate to SP	1	1	0	0	1	0	1	0				data-low								data-high												

Footnotes:

AL = 8-bit accumulator
AX = 18-bit accumulator
CX = Count register
DS = Data segment
ES = Extra segment
Above/below refers to unsigned value
Greater = more positive
Less = less positive (more negative) signed value
If d = 1 then "to" reg: If d = 0 then "from" reg.
If w = 1 then word instruction: If w = 0 then byte instruction

If mod = 11 then r/m is treated as a REG field
If mod = 00 then DISP = 0*, disp-low and disp-high are absent
If mod = 01 then DISP = disp-low sign-extended to 16-bits, disp-high is absent
If mod = 10 then DISP = disp-high: disp-low

If r/m = 000 then EA = (BX) + (SI) + DISP
If r/m = 001 then EA = (BX) + (DI) + DISP
If r/m = 010 then EA = (BP) + (SI) + DISP
If r/m = 011 then EA = (BP) + (DI) + DISP
If r/m = 100 then EA = (SI) + DISP
If r/m = 101 then EA = (DI) + DISP
If r/m = 110 then EA = (BP) + DISP*
If r/m = 111 then EA = (BX) + DISP
DISP follows 2nd byte of instruction (before data if required)
* except if mod = 00 and r/m = 110 then EA-disp-high: disp-low

If s:w = 01 then 16 bits of immediate data form the operand
If s:w = 11 then an immediate data byte is sign extended to form the 16-bit operand
If v = 0 then "count" = 1: If v = 1 then "count" in (CL)
x = don't care
z is used for string primitives for comparison with ZF FLAG

SEGMENT OVERRIDE PREFIX

001 reg 110

REG is assigned according to the following table:

16-Bit	(w = 1)	8-Bit	(w = 0)	Segment
000	AX	000	AL	00 ES
001	CX	001	CL	01 CS
010	DX	010	DL	10 SS
011	BX	011	BL	11 DS
100	SP	100	AH	
101	BP	101	CH	
110	SI	110	DH	
111	DI	111	BH	

Instructions which reference the flag register file as a 16-bit object use the symbol FLAGS to represent the file:

FLAGS = x:x:x:x:(OF):(DF):(IF):(TF):(SF):(ZF):X:(AF):X:(PF):X:(CF)