

---

# **MSM80C88A-10RS/GS/JS**

---

## **8-Bit CMOS MICROPROCESSOR**

---

### **GENERAL DESCRIPTION**

The MSM80C88A-10 is internal 16-bit CPUs with 8-bit interface implemented in Silicon Gate CMOS technology. It is designed with the same processing speed as the NMOS8088-1, but with considerably less power consumption.

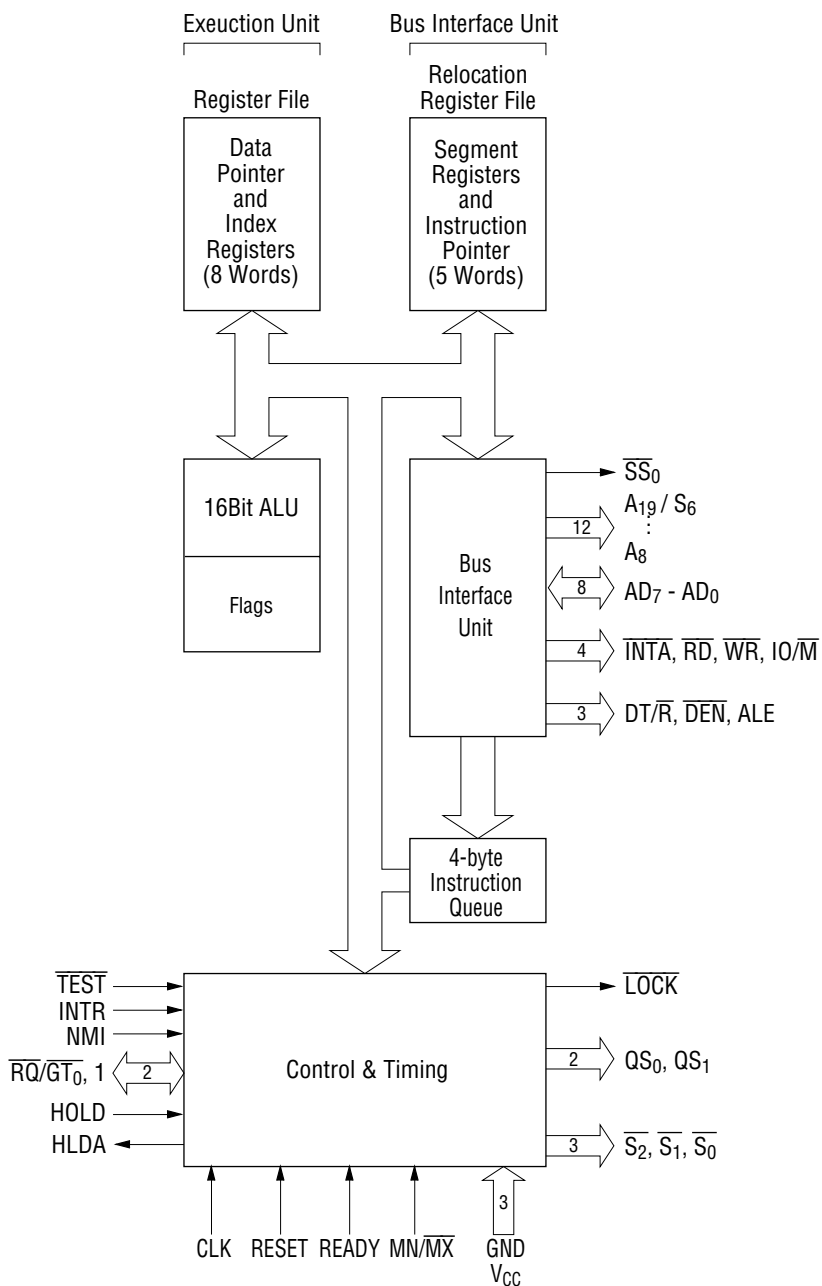
The processor has attributes of both 8 and 16-bit microprocessor. It is directly compatible with MSM80C86A-10 software and MSM80C85AH hardware and peripherals.

### **FEATURES**

- 8-Bit Data Bus interface
- 16-Bit Internal Architecture
- 1 Mbyte Direct Addressable Memory Space
- Software Compatible with MSM80C86A-10
- 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
- From DC to 10 MHz Clock Rate (Note)
- Low Power Dissipation (10mA/MHz)
- Bus Hold Circuitry Eliminated Pull-Up Resistors
- 40-pin Plastic DIP (DIP 40-P-600-2.54): (Product name: MSM80C88A-10RS)
- 44-pin Plastic QFJ (QFJ44-P-S650-1.27): (Product name: MSM80C88A-10JS)
- 56-pin Plastic QFP (QFP56-P-1519-1.00-K): (Product name: MSM80C88A-10GS-K)

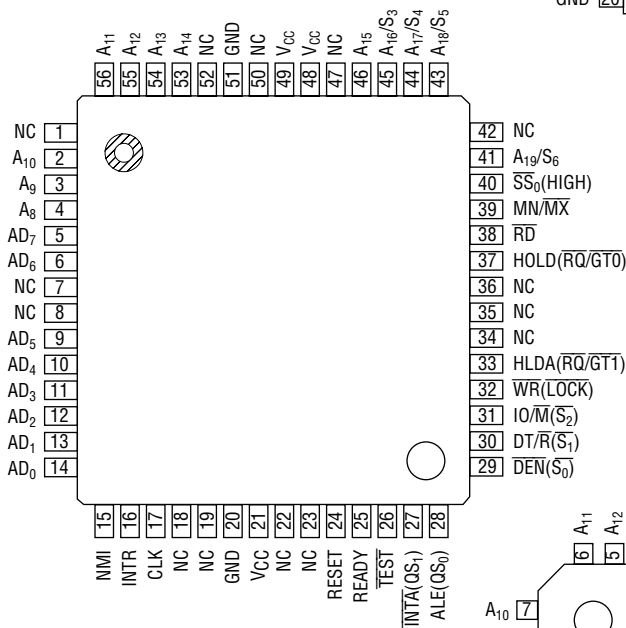
(Note) 10 MHz Spec. is not compatible with Intel 8088-1 spec.

## FUNCTIONAL BLOCK DIAGRAM

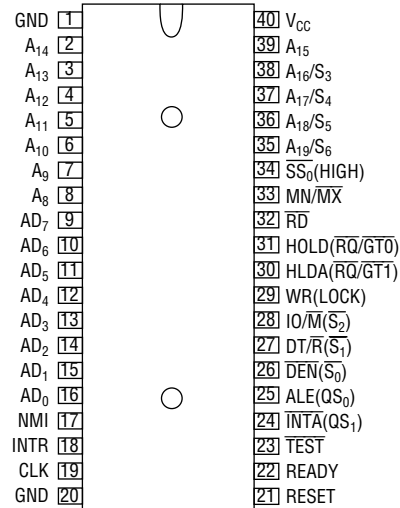


# PIN CONFIGURATION (TOP VIEW)

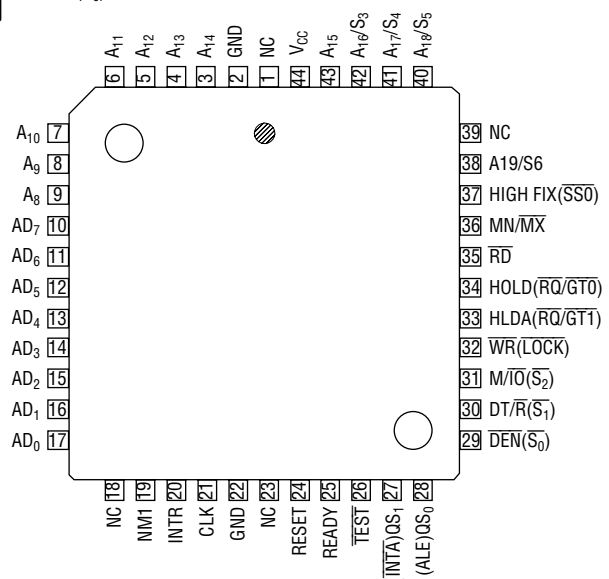
56 pin Plastic QFP



40 pin Plastic DIP



44 pin Plastic QFJ



**ABSOLUTE MAXIMUM RATING**

Parameter	Symbol	Rating			Units	Condition
		MSM80C88A-10RS	MSM80C88A-10GS	MSM80C88A-10JS		
Power Supply Voltage	$V_{CC}$	-0.5 to +7			V	With respect to GND
Input Voltage	$V_{IN}$	-0.5 to $V_{CC} + 0.5$			V	
Output Voltage	$V_{OUT}$	-0.5 to $V_{CC} + 0.5$			V	
Storage Temperature	$T_{STG}$	-65 to +150			°C	—
Power Dissipation	$P_D$	1.0	0.7		W	$T_a = 25^{\circ}\text{C}$

**OPERATING RANGE**

Parameter	Symbol	Range	Unit
Power Supply Voltage	$V_{CC}$	4.75 to 5.25	V
Operating Temperature	$T_{op}$	0 to +70	°C

**RECOMMENDED OPERATING CONDITIONS**

Parameter	Symbol	Min.	Typ.	Max.	Unit
Power Supply Voltage	$V_{CC}$	4.75	5.0	5.25	V
Operating Temperature	$T_{op}$	0	+25	+70	°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 2.0	—	$V_{CC} + 0.5$	V

\*1 Only CLK

\*2 Except CLK

## DC CHARACTERISTICS

(V<sub>CC</sub> = 4.5 to 5.5 V, T<sub>a</sub> = -40°C to +85°C)

Parameter	Symbol	Min.	Typ.	Max.	Unit	Conditions
"L" Output Voltage	V <sub>OL</sub>	—	—	0.4	V	I <sub>OL</sub> = 2.5 mA
"H" Output Voltage	V <sub>OH</sub>	3.0	—	—	V	I <sub>OH</sub> = -2.5 mA
		V <sub>CC</sub> - 0.4				I <sub>OH</sub> = -100 μA
Input Leak Current	I <sub>LI</sub>	-1.0	—	+1.0	μA	0 ≤ V <sub>IN</sub> ≤ V <sub>CC</sub>
Output Leak Current	I <sub>LO</sub>	-10	—	+10	μA	V <sub>O</sub> = V <sub>CC</sub> or GND
Input Leakage Current (Bus Hold Low)	I <sub>BHL</sub>	50	—	400	μA	V <sub>IN</sub> = 0.8 V *3
Input Leakage Current (Bus Hold High)	I <sub>BHH</sub>	-50	—	-400	μA	V <sub>IN</sub> = 3.0 V *4
Bus Hold Low Overdrive	I <sub>BHLO</sub>	—	—	600	μA	*5
Bus Hold High Overdrive	I <sub>BHHO</sub>	—	—	-600	μA	*6
Operating Power Supply Current	I <sub>CCS</sub>	—	—	10	mA/MHz	V <sub>IL</sub> = GND V <sub>IH</sub> = V <sub>CC</sub>
Standby Power Current	I <sub>CC</sub>	—	—	500	μA	V <sub>IN</sub> = V <sub>CC</sub> or GND Outputs Unloaded CLK = GND or V <sub>CC</sub>
Input Capacitance	C <sub>IN</sub>	—	—	10	pF	*7
Output Capacitance	C <sub>OUT</sub>	—	—	15	pF	*7
I/O Capacitance	C <sub>I/O</sub>	—	—	20	pF	*7

\*3 Test conditions are to lower V<sub>IN</sub> to GND and then raise V<sub>IN</sub> to 0.8 V on pins 2-16, and 35-39.

\*4 Test conditions are to raise V<sub>IN</sub> to V<sub>CC</sub> and then lower V<sub>IN</sub> to 3.0 V on pins 2-16, 26-32, and 34-39.

\*5 An external driver must source at least I<sub>BHLO</sub> to switch this node from LOW to HIGH.

\*6 An external driver must sink at least I<sub>BHHO</sub> to switch this node from HIGH to LOW.

\*7 Test Conditions: a) Freq = 1 MHz.

b) Unmeasured Pins at GND.

c) V<sub>IN</sub> at 5.0 V or GND.

## AC CHARACTERISTICS

Minimum Mode System  
Timing Requirements

Parameter	Symbol	5 MHz Spec. V <sub>CC</sub> = 4.5 V to 5.5 V Ta = -40 to +85°C		8 MHz Spec. V <sub>CC</sub> = 4.75 V to 5.25 V Ta = 0 to +70°C		10 MHz Spec. V <sub>CC</sub> = 4.75 V to 5.25 V Ta = 0 to +70°C		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
CLK Cycle Period	T <sub>CLCL</sub>	200	DC	125	DC	100	DC	ns
CLK Low Time	T <sub>CLCH</sub>	118	—	68	—	46	—	ns
CLK High Time	T <sub>CHCL</sub>	69	—	44	—	44	—	ns
CLK Rise Time (From 1.0 V to 3.5 V)	T <sub>CH1CH2</sub>	—	10	—	10	—	10	ns
CLK Fall Time (From 3.5 V to 1.0 V)	T <sub>CL2CL1</sub>	—	10	—	10	—	10	ns
Data in Setup Time	T <sub>DVCL</sub>	30	—	20	—	20	—	ns
Data in Hold Time	T <sub>CLDX</sub>	10	—	10	—	10	—	ns
RDY Setup Time into MSM 82C84A-2 (See Notes 1, 2)	T <sub>R1VCL</sub>	35	—	35	—	35	—	ns
RDY Hold Time into MSM 82C84A-2 (See Notes 1, 2)	T <sub>CLR1X</sub>	0	—	0	—	0	—	ns
READY Setup Time into MSM80C88A-10	T <sub>RYHCH</sub>	118	—	68	—	46	—	ns
READY Hold Time into MSM80C88A-10	T <sub>CHRYX</sub>	30	—	20	—	20	—	ns
READY inactive to CLK (See Note 3)	T <sub>RYLCL</sub>	-8	—	-8	—	-8	—	ns
HOLD Setup Time	T <sub>HVCH</sub>	35	—	20	—	20	—	ns
INTR, NMI, TEST Setup Time (See Note 2)	T <sub>INVCH</sub>	30	—	15	—	15	—	ns
Input Rise Time (Except CLK) (From 0.8 V to 2.0 V)	T <sub>ILIH</sub>	—	15	—	15	—	15	ns
Input Fall Time (Except CLK) (From 2.0 V to 0.8 V)	T <sub>IHIL</sub>	—	15	—	15	—	15	ns

## Timing Responses

Parameter	Symbol	5 MHz Spec. $V_{CC} = 4.5 \text{ V to } 5.5 \text{ V}$ $T_a = -40 \text{ to } +85^\circ\text{C}$		8 MHz Spec. $V_{CC} = 4.75 \text{ V to } 5.25 \text{ V}$ $T_a = 0 \text{ to } +70^\circ\text{C}$		10 MHz Spec. $V_{CC} = 4.75 \text{ V to } 5.25 \text{ V}$ $T_a = 0 \text{ to } +70^\circ\text{C}$		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
Address Valid Delay	$t_{CLAV}$	10	110	10	60	10	60	ns
Address Hold Time	$t_{CLAX}$	10	—	10	—	10	—	ns
Address Float Delay	$t_{CLAZ}$	$t_{CLAX}$	80	$t_{CLAX}$	50	$t_{CLAX}$	50	ns
ALE Width	$t_{LHLL}$	$t_{CLCH}-20$	—	$t_{CLCH}-10$	—	$t_{CLCH}-10$	—	ns
ALE Active Delay	$t_{CLLH}$	—	80	—	50	—	40	ns
ALE Inactive Delay	$t_{CHLL}$	—	85	—	55	—	45	ns
Address Hold Time to ALE Inactive	$t_{LLAX}$	$t_{CLCH}-10$	—	$t_{CLCH}-10$	—	$t_{CLCH}-10$	—	ns
Data Valid Delay	$t_{CLDV}$	10	110	—	60	10	60	ns
Data Hold Time	$t_{CHDX}$	10	—	—	—	10	—	ns
Data Hold Time after $\overline{WR}$	$t_{WHDX}$	$t_{CLCH}-30$	—	$t_{CLCH}-30$	—	$t_{CLCH}-25$	—	ns
Control Active Delay 1	$t_{CVCTV}$	10	110	10	70	10	55	ns
Control Active Delay 2	$t_{CHCTV}$	10	110	10	60	10	50	ns
Control Inactive Delay	$t_{CVCTX}$	10	110	10	70	10	55	ns
Address Float to $\overline{RD}$ Active	$t_{AZRL}$	0	—	0	—	0	—	ns
$\overline{RD}$ Active Delay	$t_{CLRL}$	10	165	10	100	10	70	ns
$\overline{RD}$ Inactive Delay	$t_{CLRH}$	10	150	10	80	10	60	ns
$\overline{RD}$ Inactive to Next Address Active	$t_{RHAV}$	$t_{CLCH}-45$	—	$t_{CLCH}-40$	—	$t_{CLCL}-35$	—	ns
HLDA Valid Delay	$t_{CLHAV}$	10	160	10	100	10	60	ns
$\overline{RD}$ Width	$t_{RLRH}$	$2t_{CLCL}-75$	—	$2t_{CLCL}-50$	—	$2t_{CLCL}-40$	—	ns
$\overline{WR}$ Width	$t_{WLWH}$	$2t_{CLCL}-60$	—	$2t_{CLCL}-40$	—	$2t_{CLCL}-35$	—	ns
Address Valid to ALE Low	$t_{AVAL}$	$t_{CLCH}-60$	—	$t_{CLCH}-40$	—	$t_{CLCH}-35$	—	ns
Output Rise Time (From 0.8 V to 2.0 V)	$t_{OLOH}$	—	15	—	15	—	15	ns
Output Fall Time (From 2.0 V to 0.8 V)	$t_{OHOL}$	—	15	—	15	—	15	ns

- Notes: 1. Signals at MSM82C84A-2 shown for reference only.  
2. Setup requirement for asynchronous signal only to guarantee recognition at next CLK.  
3. Applies only to  $T_2$  state. (8 ns into  $T_3$ )

### Maximum Mode System (Using MSM82C88-2 Bus Controller) Timing Requirements

Parameter	Symbol	5 MHz Spec. $V_{CC} = 4.5\text{ V to }5.5\text{ V}$ $T_a = -40\text{ to }+85^\circ\text{C}$		8 MHz Spec. $V_{CC} = 4.75\text{ V to }5.25\text{ V}$ $T_a = 0\text{ to }+70^\circ\text{C}$		10 MHz Spec. $V_{CC} = 4.75\text{ V to }5.25\text{ V}$ $T_a = 0\text{ to }+70^\circ\text{C}$		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
CLK Cycle Period	$t_{CLCL}$	200	DC	125	DC	100	DC	ns
CLK Low Time	$t_{CLCH}$	118	—	68	—	46	—	ns
CLK High Time	$t_{CHCL}$	69	—	44	—	44	—	ns
CLK Rise Time (From 1.0 V to 3.5 V)	$t_{CH1CH2}$	—	10	—	10	—	10	ns
CLK Fall Time (From 3.5 V to 1.0 V)	$t_{CL2CL1}$	—	10	—	10	—	10	ns
Data in Setup Time	$t_{DVCL}$	30	—	20	—	20	—	ns
Data in Hold Time	$t_{CLDX}$	10	—	10	—	10	—	ns
RDY Setup Time into MSM 82C84A-2 (See Notes 1, 2)	$t_{R1VCL}$	35	—	35	—	35	—	ns
RDY Hold Time into MSM82C84A-2 (See Notes 1, 2)	$t_{CLR1X}$	0	—	0	—	0	—	ns
READY Setup Time into MSM80C88A-10	$t_{RYHCH}$	118	—	68	—	46	—	ns
READY Hold Time into MSM80C88A-10	$t_{CHRYX}$	30	—	20	—	20	—	ns
READY inactive to CLK (See Note 3)	$t_{RYLCL}$	-8	—	-8	—	-8	—	ns
Setup Time for Recognition (NMI, INTR, TEST) (See Note 2)	$t_{INVCH}$	30	—	15	—	15	—	ns
$\overline{RQ}/\overline{GT}$ Setup Time	$t_{GVCH}$	30	—	15	—	15	—	ns
$\overline{RQ}$ Hold Time into MSM80C88A-10	$t_{CHGX}$	40	—	30	—	20	—	ns
Input Rise Time (Except CLK) (From 0.8 V to 2.0 V)	$t_{ILIH}$	—	15	—	15	—	15	ns
Input Fall Time (Except CLK) (From 2.0 V to 0.8 V)	$t_{IHIL}$	—	15	—	15	—	15	ns

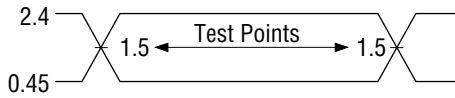


## Timing Responses

Parameter	Symbol	5 MHz Spec. $V_{CC} = 4.5 \text{ V to } 5.5 \text{ V}$ $T_a = -40 \text{ to } +85^\circ\text{C}$		8 MHz Spec. $V_{CC} = 4.75 \text{ V to } 5.25 \text{ V}$ $T_a = 0 \text{ to } +70^\circ\text{C}$		10 MHz Spec. $V_{CC} = 4.75 \text{ V to } 5.25 \text{ V}$ $T_a = 0 \text{ to } +70^\circ\text{C}$		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	
Command Active Delay (See Note 1)	$t_{CLML}$	5	45	5	35	5	35	ns
Command Inactive Delay (See Note 1)	$t_{CLMH}$	5	45	5	45	5	45	ns
READY Active to Status Passive (See Note 4)	$t_{RYHSH}$	—	110	—	65	—	45	ns
Status Active Delay	$t_{CHSV}$	10	110	10	60	10	45	ns
Status Inactive Delay	$t_{CLSH}$	10	130	10	70	10	60	ns
Address Valid Delay	$t_{CLAV}$	10	110	10	60	10	60	ns
Address Hold Time	$t_{CLAX}$	10	—	10	—	10	—	ns
Address Float Delay	$t_{CLAZ}$	$t_{CLAX}$	80	$t_{CLAX}$	50	$t_{CLAX}$	50	ns
Status Valid to ALE High (See Note 1)	$t_{SVLH}$	—	35	—	25	—	25	ns
Status Valid to MCE High (See Note 1)	$t_{SVMCH}$	—	35	—	30	—	30	ns
CLK Low to ALE Valid (See Note 1)	$t_{CLLH}$	—	35	—	25	—	25	ns
CLK Low to MCE High (See Note 1)	$t_{CLMCH}$	—	35	—	25	—	25	ns
ALE Inactive Delay (See Note 1)	$t_{CHLL}$	4	35	4	25	4	25	ns
Data Valid Delay	$t_{CLDV}$	10	110	10	60	10	60	ns
Data Hold Time	$t_{CHDX}$	10	—	10	—	10	—	ns
Control Active Delay (See Note 1)	$t_{CVNV}$	5	45	5	45	5	45	ns
Control Inactive Delay (See Note 1)	$t_{CVNX}$	5	45	5	45	5	45	ns
Address Float to RD Active	$t_{AZRL}$	0	—	0	—	0	—	ns
RD Active Delay	$t_{CLRL}$	10	165	10	100	10	70	ns
RD Inactive Delay	$t_{CLRH}$	10	150	10	80	10	60	ns
RD Inactive to Next Address Active	$t_{RHAV}$	$t_{CLCL}-45$	—	$t_{CLCL}-40$	—	$t_{CLCL}-35$	—	ns
Direction Control Active Delay (See Note 1)	$t_{CHDTL}$	—	50	—	50	—	50	ns
Direction Control Inactive Delay (See Note 1)	$t_{CHDTH}$	—	35	—	30	—	30	ns
GT Active Delay (See Note 5)	$t_{CLGL}$	0	85	0	50	0	45	ns
GT Inactive Delay	$t_{CLGH}$	0	85	0	50	0	45	ns
RD Width	$t_{RLRH}$	$2t_{CLCL}-75$	—	$2t_{CLCL}-50$	—	$2t_{CLCL}-40$	—	ns
Output Rise Time (From 0.8 V to 2.0 V)	$t_{OLOH}$	—	15	—	15	—	15	ns
Output Fall Time (From 2.0 V to 0.8 V)	$t_{OHOL}$	—	15	—	15	—	15	ns

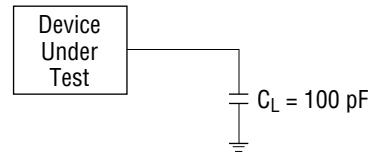
- Notes: 1. Signals at MSM82C84A-2 or MSM82C88-2 are shown for reference only.  
2. Setup requirement for asynchronous signal only to guarantee recognition at next CLK.  
3. Applies only to  $T_2$  state (8 ns into  $T_3$ )  
4. Applies only to  $T_3$  and wait states.  
5.  $C_L = 40 \text{ pF}$  ( $RQ/\overline{GT}_0$ ,  $RQ/\overline{GT}_1$ )

### A.C. Testing Input, Output Waveform



A.C. Testing: Inputs are driven at 2.4 V for a logic "1" and 0.45 V for a logic "0" timing measurements are 1.5 V for both a logic "1" and "0".

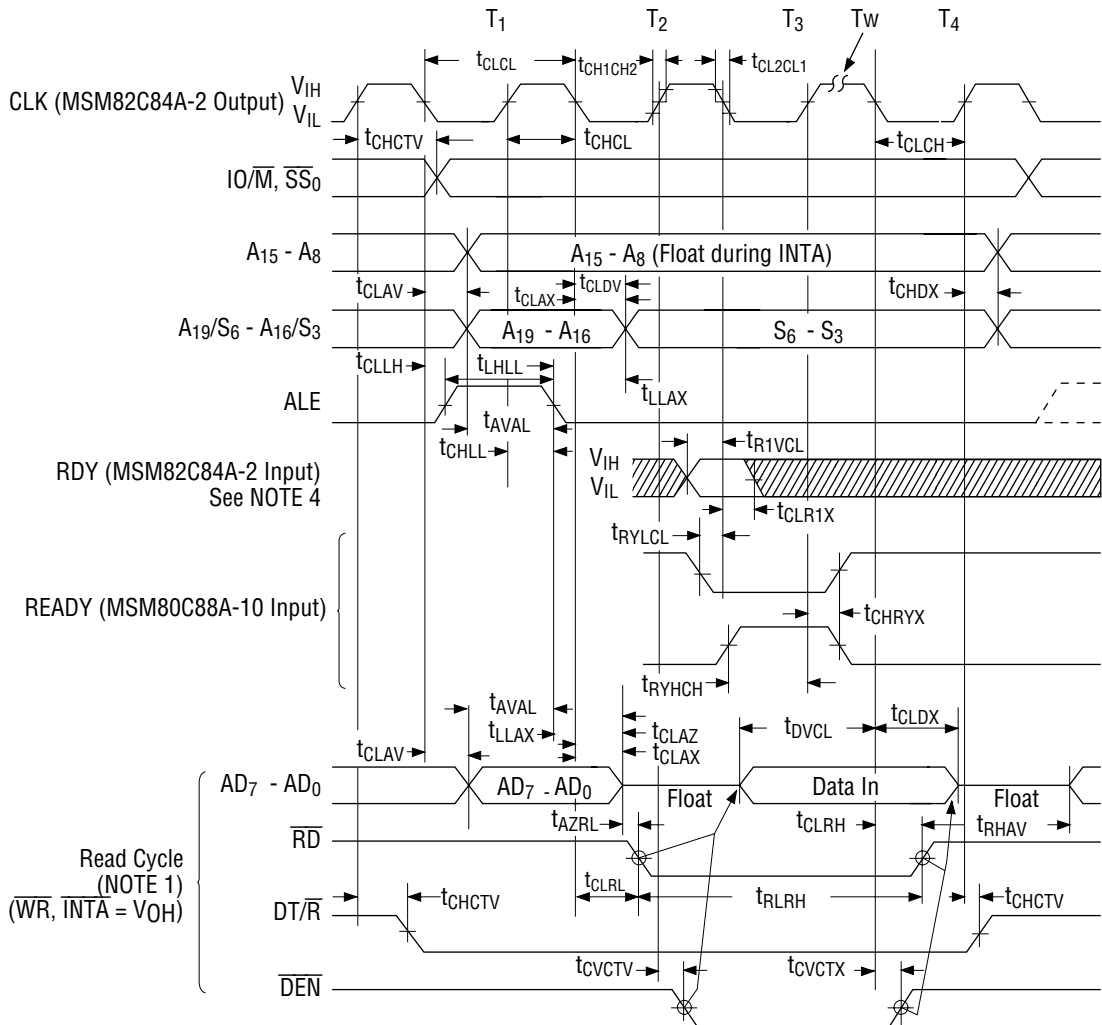
### A.C. Testing Load Circuit



$C_L$  includes jig capacitance.

## TIMING CHART

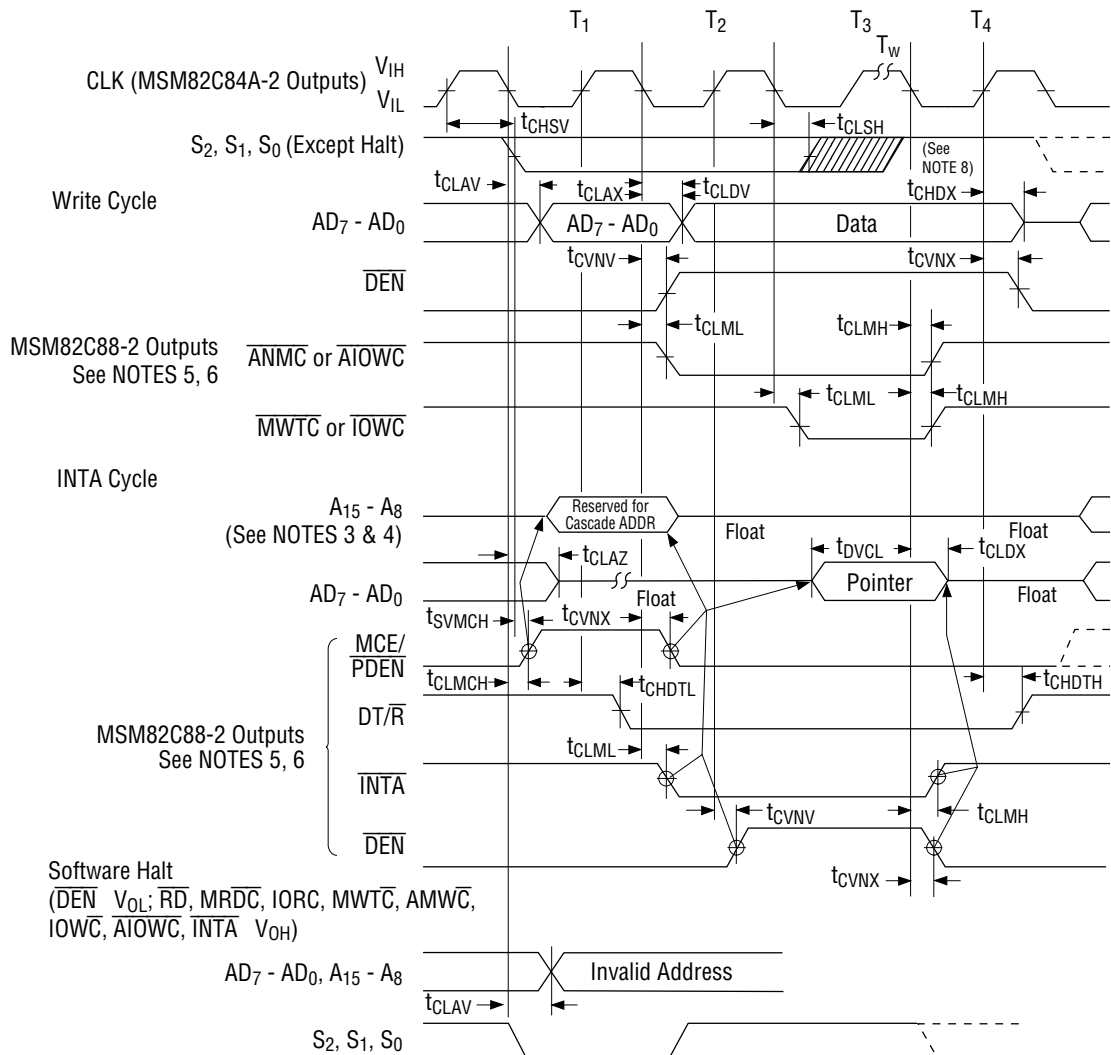
### Minimum Mode





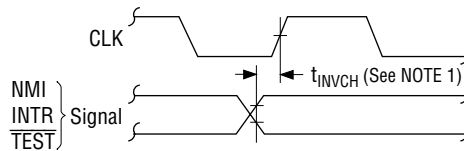


# Maximum Mode (continued)



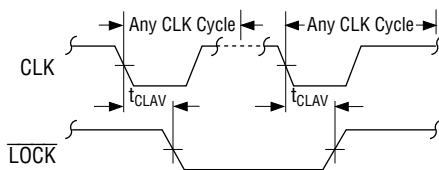
- Notes:
1. All signals switch between V<sub>OH</sub> and V<sub>OL</sub> unless otherwise specified.
  2. RDY is sampled near the end of T<sub>2</sub>, T<sub>3</sub>, T<sub>w</sub> to determine if T<sub>w</sub> machines states are to be inserted.
  3. Cascade address is valid between first and second  $\overline{\text{INTA}}$  cycle.
  4. Two  $\overline{\text{INTA}}$  cycles run back-to-back. The MSM80C86A-10 LOCAL ADDR/DATA BUS is floating during both  $\overline{\text{INTA}}$  cycles. Control for pointer address is shown for second  $\overline{\text{INTA}}$  cycle.
  5. Signal at MSM82C84A-2 or MSM82C88-2 shown for reference only.  
The issuance of the MSM82C88-2 command and control signals ( $\overline{\text{MRDC}}$ ,  $\overline{\text{MWTC}}$ ,  $\overline{\text{AMWC}}$ ,  $\overline{\text{IORC}}$ ,  $\overline{\text{IOWC}}$ ,  $\overline{\text{AIOWC}}$ ,  $\overline{\text{INTA}}$  and  $\overline{\text{DEN}}$ ) lags the active high MSM82C88-2 CEN.
  7. All timing measurements are made at 1.5 V unless otherwise noted.
  8. Status inactive in state just prior to T<sub>4</sub>.

## Asynchronous Signal Recognition

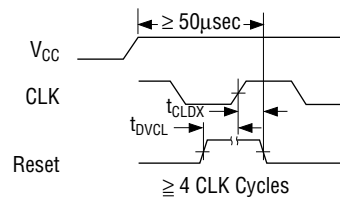


**NOTE: 1** Setup requirements for asynchronous signals only to guarantee recognition at next CLK

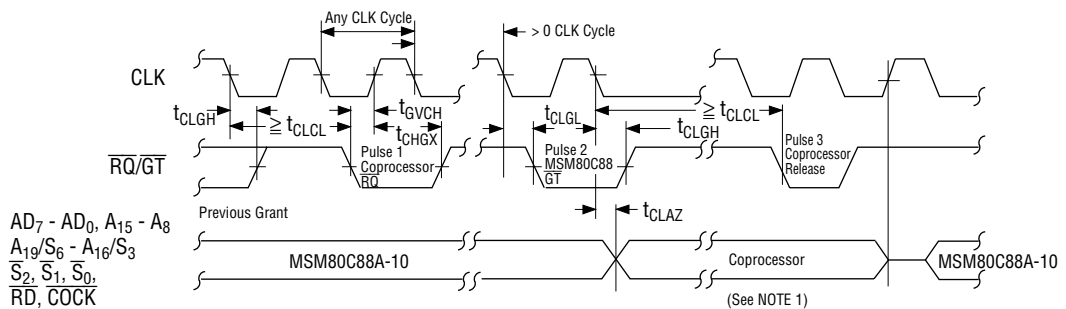
## Bus Lock Signal Timing (Maximum Mode Only)



## Reset Timing

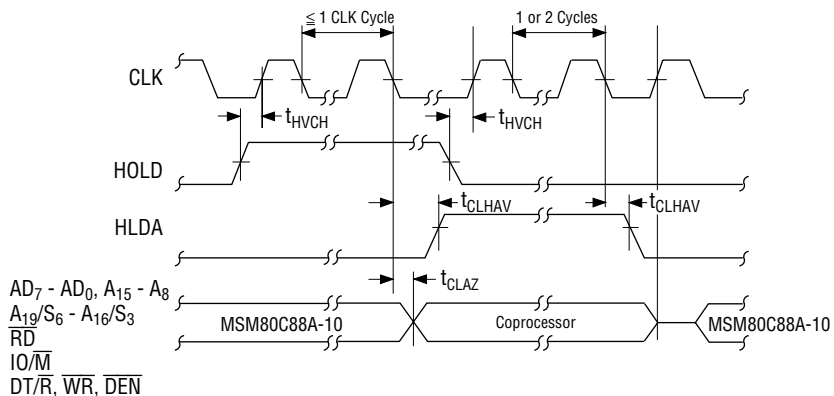


## Request/Grant Sequence Timing (Maximum Mode Only)



**NOTE: 1** The coprocessor may not drive the busses outside the region shown without risking contention

## Hold/Hold Acknowledge Timing (Minimum Mode Only)



## PIN DESCRIPTION

### AD<sub>0</sub> - AD<sub>7</sub>

ADDRESS DATA BUS: Input/Output

These lines are the multiplexed address and data bus.

These are the address bus at T<sub>1</sub> cycle and the data bus at T<sub>2</sub>, T<sub>3</sub>, T<sub>W</sub> and T<sub>4</sub> cycle.

T<sub>2</sub>, T<sub>3</sub>, T<sub>W</sub> and T<sub>4</sub> cycle.

These lines are high impedance during interrupt acknowledge and hold acknowledge.

### A<sub>8</sub> - A<sub>15</sub>

ADDRESS BUS: Output

These lines are the address bus bits 8 thru 15 at all cycles.

These lines do not have to be latched by an ALE signal.

These lines are high impedance during interrupt acknowledge and hold acknowledge.

### A<sub>16</sub>/S<sub>3</sub>, A<sub>17</sub>/S<sub>4</sub>, A<sub>18</sub>/S<sub>5</sub>, A<sub>19</sub>/S<sub>6</sub>

ADDRES/STATUS : Output

These are the four most significant address as at the T<sub>1</sub> cycle.

Accessing I/O port address, these are low at T<sub>1</sub> Cycle.

These lines are Status lines at the T<sub>2</sub>, T<sub>3</sub>, T<sub>W</sub> and T<sub>4</sub> Cycles.

S<sub>5</sub> indicates interrupt enable Flag.

S<sub>3</sub> and S<sub>4</sub> are encoded as shown below.

S <sub>3</sub>	S <sub>4</sub>	Characteristics
0	0	Alternate Data
1	0	Stack
0	1	Code or None
1	1	Data

These lines are high impedance during hold acknowledge.

### RD

READ: Output

This line indicates that CPU is in a memory or I/O read cycle.

This line is the read strobe signal when CPU reads data from a 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 the CPU that the addressed memory or I/O device is ready to read or write.

This line is active high. If the setup and hold time are out of specification, an illegal operation will occur.

### INTR

INTERRUPT REQUEST: Input

This line is the level triggered interrupt request signal which is sampled during the last clock cycle of instruction and string manipulations.

It can be internally masked by software.

This signal is active high and internally synchronized.

**TEST**

TEST: Input

This line is examined by a "WAIT" instruction.

When  $\overline{\text{TEST}}$  is high, the CPU enters an idle cycle.When  $\overline{\text{TEST}}$  is low, the CPU exits in an idle cycle.**NMI**

NON MASKABLE INTERRUPT: Input

This line causes a type 2 interrupt.

NMI is not maskable.

This signal is internally synchronized and needs 2-clock cycle pulse width.

**RESET**

RESET: Input

This signal causes the CPU to initialize immediately.

This signal is active high and must be at least four clock cycles.

**CLK**

CLOCK: Input

This signal provides the basic timing for the internal circuit.

**MN/ $\overline{\text{MX}}$** 

MINIMUM/MAXIMUM: Input

This signal selects the CPU's operating mode.

When  $V_{CC}$  is connected, the CPU operates in minimum mode.

When GND is connected, the CPU operates in maximum mode.

 **$V_{CC}$**  $V_{CC}$ : +5V supplied.**GND**

GROUND

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

 **$\overline{\text{S}}_0, \overline{\text{S}}_1, \overline{\text{S}}_2$** 

STATUS: Output

These lines indicate bus status and they are used by the MSM82C88-2 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 below.

$\overline{\text{S}}_2$	$\overline{\text{S}}_1$	$\overline{\text{S}}_0$	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



**$\overline{\text{RQ/GT}}_0$**  **$\overline{\text{RQ/GT}}_1$** 

REQUEST/GRANT:Input/Output

These lines are used for Bus Request from other devices and Bus GRANT to other devices.

These lines are bidirectional and active low.

 **$\overline{\text{LOCK}}$** 

LOCK:Output

This line is active low.

When this line is low, other devices cannot gain control of the bus.

This line is high impedance hold acknowledge.

 **$\text{QS}_0/\text{QS}_1$** 

QUEUE STATUS: Output

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

<b><math>\text{QS}_1</math></b>	<b><math>\text{QS}_0</math></b>	<b>Characteristics</b>
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.

 **$\text{IO}/\overline{\text{M}}$** 

STATUS: Output

This line selects memory address space or I/O address space.

When this line is low, the CPU selects memory address space and when it is high, the CPU selects I/O address space.

This line is high impedance during hold acknowledge.

 **$\overline{\text{WR}}$** 

WRITE: Output

This line indicates that the CPU is in a memory or I/O write cycle.

This line is a write strobe signal when the CPU writes data to memory or an I/O device.

This line is active low. This line is high impedance during hold acknowledge.

 **$\overline{\text{INTA}}$** 

INTERRUPT ACKNOWLEDGE: Output

This line is a read strobe signal for the interrupt acknowledge cycle.

This line is active low.

**ALE**

ADDRESS LATCH ENABLE: Output

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

**DT/ $\overline{R}$** 

DATA TRANSMIT/RECEIVE: Output

This line is used to control the direction of the bus transceiver.

When this line is high, the CPU transmits data, and when it is low. the CPU receives data.

This line is high impedance during hold acknowledge.

 **$\overline{DEN}$** 

DATA ENABLE: Output

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

**HOLD**

HOLD REQUEST: Input

This line is used for a Bus Request from an other device.

This line is active high.

**HLDA**

HOLD ACKNOWLEDGE: Output

This line is used for a Bus Grant to an other device.

This line is active high.

 **$\overline{SS_0}$** 

STATUS: Output

This line is logically equivalent to  $\overline{S_0}$  in the maximum mode.

## STATIC OPERATION

The MSM80C88A-10 circuitry is of static design. Internal registers, counters and latches are static and require no refresh as with dynamic circuit design. This eliminates the minimum operating frequency restriction placed on other microprocessors. The MSM80C88A-10 can operate from DC to the appropriate upper frequency limit. The processor clock may be stopped in either state (high/low) and held there indefinitely. This type of operation is especially useful for system debug or power critical applications.

The MSM80C88A-10 can be signal stepped using only the CPU clock. This state can be maintained as long as is necessary. Signal step clock operation allows simple interface circuitry to provide critical information for bringing up your system.

Static design also allows very low frequency operation (down to DC). In a power critical situation, this can provide extremely low power operation since 80C88A power dissipation is directly related to operating frequency. As the system frequency is reduced, so is the operating power until, ultimately, at a DC input frequency, the MSM80C88A-10 power requirement is the standby current (500  $\mu$ A maximum).

## FUNCTIONAL DESCRIPTION

### General Operation

The internal function of the MSM80C88A-10 consists of a Bus interface Unit (BIU) and an Execution Unit (EU). These units operate mutually but perform as separate processors.

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

EU receives pre-fetched instructions from the BIU queue, decodes and executes instructions and provides an un-relocated operand address to the BIU.

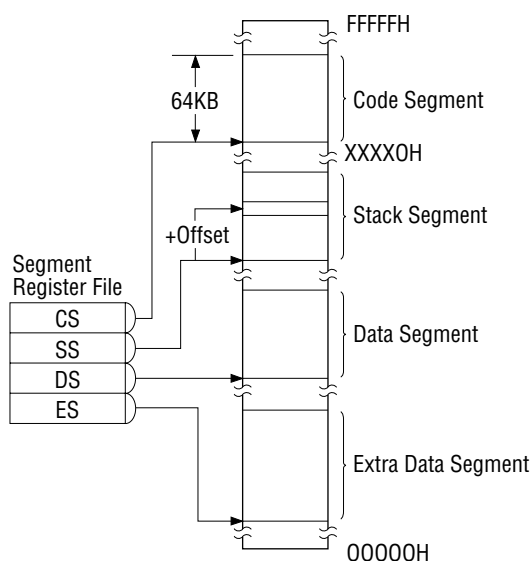
### Memory Organization

The MSM80C88A-10 has a 20-bit address to memory. Each address has 8-bit data width. Memory is organized 00000H to FFFFFH and is logically divided into four segments: code, data, extra data and stack segment. Each segment contains up to 64 Kbytes and locates on a 16-byte boundary. (Fig. 3a)

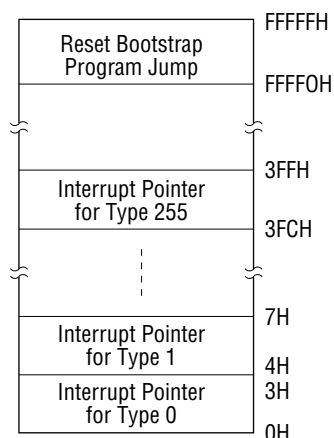
All memory references are made relative to a segment register according to a select rule. Memory location FFFF0H is the start address after reset, and 00000H through 003FFH are reserved as an interrupt pointer. There are 256 types of interrupt pointer:

Each interrupt type has a 4-byte pointer element consisting of a 16-bit segment address and a 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 (CS)	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

The MSM80C88A-10 has two system modes: minimum and maximum. When using the maximum mode, it is easy to organize a multiple-CPU system with the MSM82C88-2 Bus Controller which generates the bus control signal.

When using the minimum mode, it is easy to organize a simple system by generating the bus control signal itself. MN/MX is the mode select pin. Definition of 24-31, 34 pin changes depends on the MN/MX pin.

## Bus Operation

The MSM80C88A-10 has a time multiplexed address and data bus. If a non-multiplexed bus is desired for the system, it is only needed to add the address latch.

A CPU bus cycle consists of at least four clock cycles:  $T_1$ ,  $T_2$ ,  $T_3$  and  $T_4$ . (Fig. 4)

The address output occurs during  $T_1$ , and data transfer occurs during  $T_3$  and  $T_4$ .  $T_2$  is used for changing the direction of the bus during read operation. When the device which is accessed by the CPU is not ready to data transfer and send to the CPU "NOT READY" is indicated  $T_W$  cycles are inserted between  $T_3$  and  $T_4$ .

When a bus cycle is not needed,  $T_1$  cycles are inserted between the bus cycles for internal execution. At the  $T_1$  cycle an ALE signal is output from the CPU or the MSM82C88-2 depending in  $MN/\overline{MX}$ , at the trailing edge of an ALE, a valid address may be latched. Status bits  $S_0$ ,  $S_1$  and  $S_2$  are used, in maximum mode, by the bus controller to recognize the type of bus operation according to the following table.

$\overline{S_2}$	$\overline{S_1}$	$\overline{S_0}$	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  $S_3$  through  $S_6$  are multiplexed with  $A_{16}$ - $A_{19}$ , and therefore they are valid during  $T_2$  through  $T_4$ .  $S_3$  and  $S_4$  indicate which segment register was selected on the bus cycle, according to the following table.

$S_4$	$S_3$	Characteristics
0 (LOW)	0	Alternate Data (Extra Segment)
0	1	Stack
1 (HIGH)	0	Code or None
1	1	Data

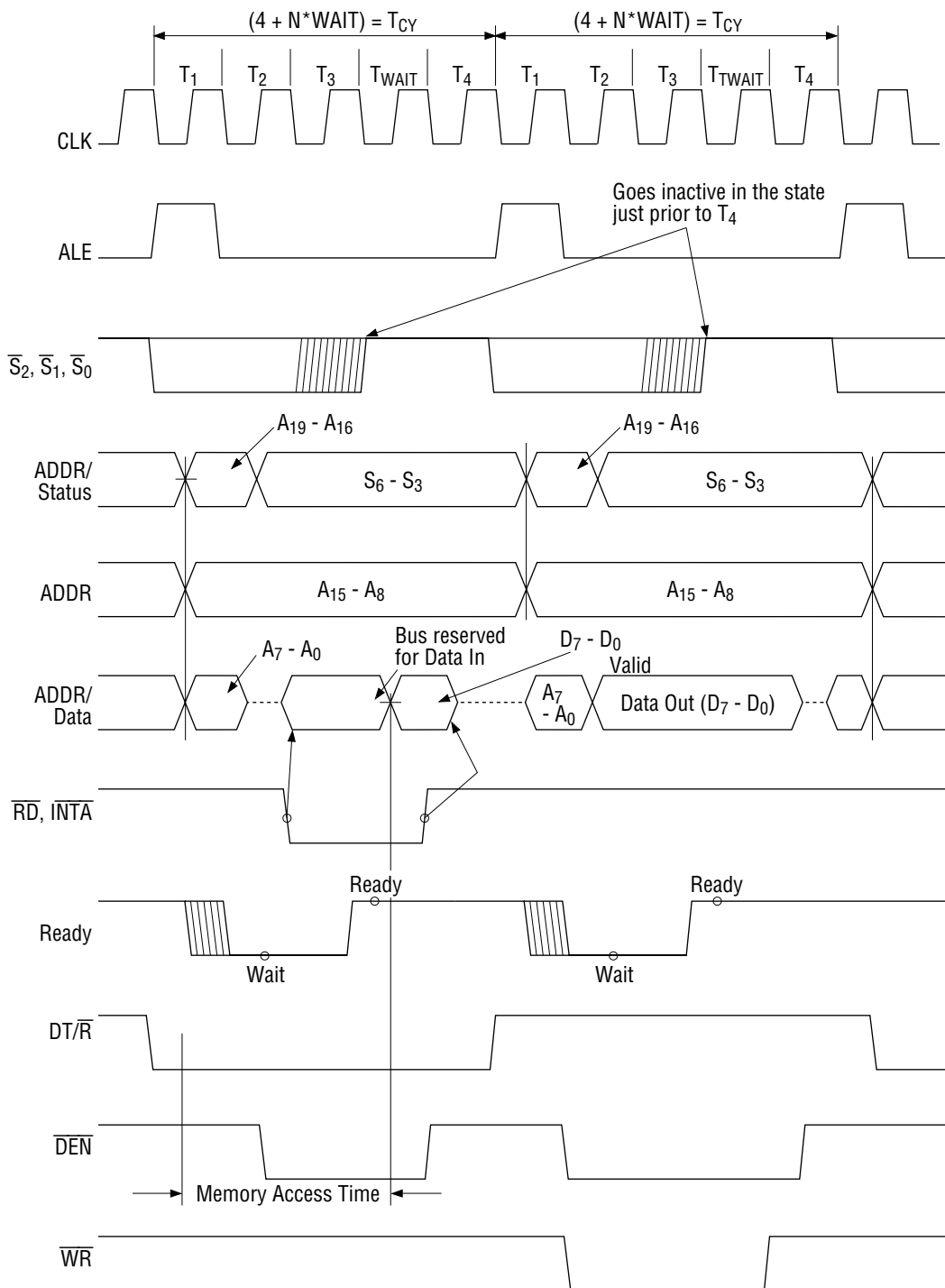
$S_5$  indicates interrupt enable Flag.

## I/O Addressing

The MSM80C88A-10 has a 64 Kbyte I/O. When the CPU accesses an I/O device, addresses  $A_0$ - $A_{15}$  are in same format as a memory access, and  $A_{16}$ - $A_{19}$  are low.

I/O ports addresses are same as four memory.

## Basic System Timing



## EXTERNAL INTERFACE

### Reset

CPU initialization is executed by the RESET pin. The MSM80C88A-10's RESET High signal is required for greater than 4 clock cycles.

The rising edge of RESET terminates the present operation immediately. The falling edge of RESET triggers an internal reset sequence for approximately 10 clock cycles. After internal reset sequence is finished, normal operation begins from absolute location FFFF0H.

### Interrupt Operations

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

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

### Non-maskable Interrupt (NMI)

The MSM80C88A-10 has a non-maskable interrupt (NMI) which is of higher priority than a maskable interrupt request (INTR).

An NMI request pulse width needs minimum of 2 clock cycles. The NMI will be serviced at the end of the current instruction or between string manipulations.

### Maskable Interrupt (INTR)

The MSM80C88A-10 provides another interrupt request (INTR) which can be masked by software. INTR is level triggered, so it must be held until interrupt request is acknowledged. The INTR will be serviced at the end of the current instruction or between string manipulations.

### Interrupt Acknowledge

During the interrupt acknowledge sequence, further interrupts are disabled. The interrupt enable bit is reset by any interrupt, after which the Flag register is automatically pushed onto the stack. During an acknowledge sequence, the CPU emits the lock signal from T<sub>2</sub> of first bus cycle to T<sub>2</sub> of second bus cycle. At the second bus cycle, a byte is fetched from the external device as a vector which identifies the type of interrupt. This vector is multiplied by four and used as an 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 a Halt instruction is executed, the CPU enters Halt state. An interrupt request or RESET will force the MSM80C88A-10 out of the Halt state.

### System Timing – Minimum Mode

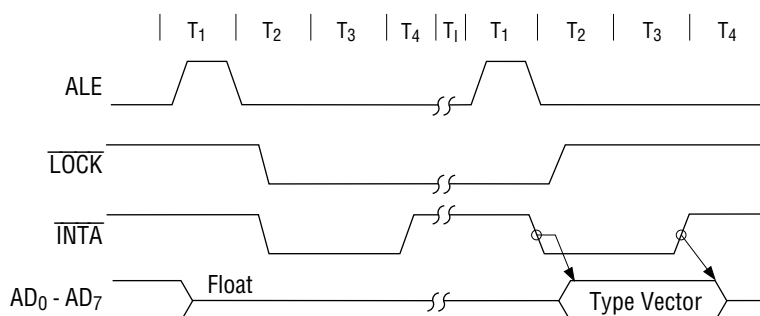
A bus cycle begins at  $T_1$  with an ALE signal. The trailing edge of ALE is used to latch the address. From  $T_1$  to  $T_4$  the  $\text{IO}/\overline{\text{M}}$  signal indicates a memory or I/O operation. From  $T_2$  to  $T_4$ , the address data bus changes the address but to the data bus.

The read ( $\overline{\text{RD}}$ ), write ( $\overline{\text{WR}}$ ), and interrupt acknowledge ( $\overline{\text{INTA}}$ ) signals caused the addressed device to enable the data bus. These signals become active at the beginning of  $T_2$  and inactive at the beginning of  $T_4$ .

### System Timing – Maximum Mode

In maximum mode, the MSM82C88-2 Bus Controller is added to system. The CPU sends status information to the Bus Controller. Bus timing signals are generated by the Bus Controller. Bus timing is almost the same as in minimum mode.

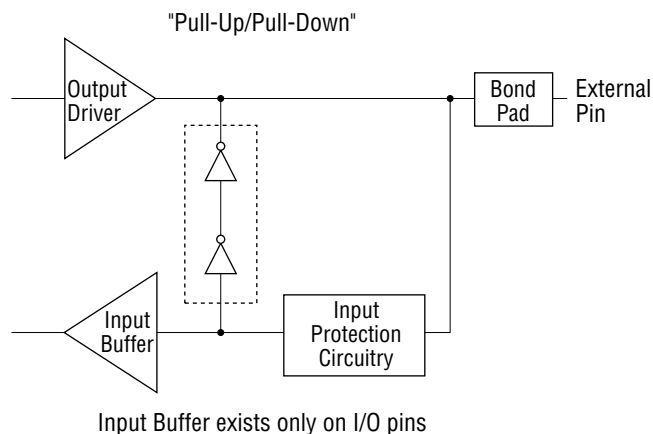
### Interrupt Acknowledge Sequence



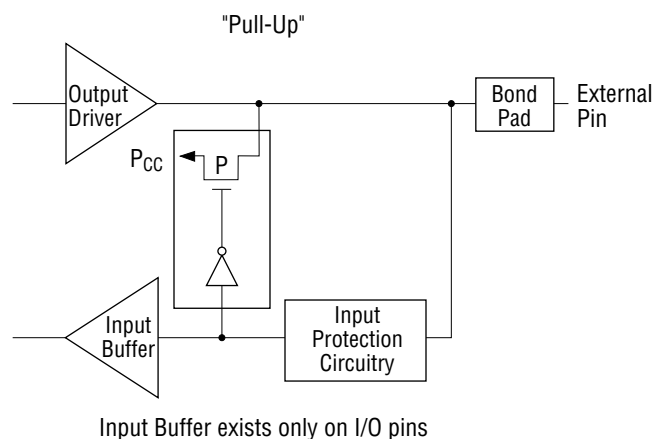


## BUS HOLD CIRCUITRY

To avoid high current conditions caused by floating inputs to CMOS devices, and to eliminate the need for pull-up/down resistors, "bus-hold" circuitry has been used on MSM80C88A-10 pins 2-16, 26-32, and 34-39 (Figures 6a, 6b). These circuits will maintain the last valid logic state if no driving source is present (i.e. an unconnected pin or a driving source which goes to a high impedance state). To overdrive the "bus hold" circuits, an external driver must be capable of supplying approximately 400  $\mu$ A minimum sink or source current at valid input voltage levels. Since this "bus hold" circuitry is active and not a "resistive" type element, the associated power supply current is negligible and power dissipation is significantly reduced when compared to the use of passive pull-up resistors.



**Figure 6a. Bus Hold Circuitry Pin 2-16, 35-39**



**Figure 6b. Bus Hold Circuit Pin 26-32, 34**

# DATA TRANSFER

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

# ARITHMETIC

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

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

## LOGIC

NOT = Invert	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 with register to either Immediate to register/memory Immediate to accumulator	0	0	1	0	0	0	d	w	mod		reg	r/m		
	1	0	0	0	0	0	0	w	mod	1	0	0	r/m	data if w = 1
	0	0	1	0	0	1	0	w			data			
TEST = And function to flags, no result: Register/memory and register Immediate data and register/memory Immediate data and accumulator	1	0	0	0	0	1	0	w	mod		reg	r/m		
	1	1	1	0	1	1	1	w	mod	0	0	0	r/m	data if w = 1
	1	0	1	0	1	0	0	w			data			
OR = Or: Reg./memory and register to either Immediate to register/memory Immediate to accumulator	0	0	0	0	1	0	d	w	mod		reg	r/m		
	1	0	0	0	0	0	0	w	mod	0	0	1	r/m	data if w = 1
	0	0	0	0	1	1	0	w			data			
XOR = Exclusive or: Reg./memory and register to either Immediate to register/memory Immediate to accumulator	0	0	1	1	0	0	d	w	mod		reg	r/m		
	1	0	0	0	0	0	0	w	mod	1	1	0	r/m	data if w = 1
	0	0	1	1	0	1	0	w			data			

# 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 sigh	0 1 1 1 1 0 0 1	disp		
LOOP = Loop CX times	1 1 0 1 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 = Complementary 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							
LOCK = Bus lock prefix	1	1	1	1	0	0	0	0											

# CONTROL TRANSFER

CALL = Call:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
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	0	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								dat-high			

Notes: AL = 8-bit accumulator

AX = 18-bit accumulator

CX = Count register

DS = Data segment

EX = 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)



**NOTICE ON REPLACING LOW-SPEED DEVICES WITH HIGH-SPEED DEVICES**

The conventional low speed devices are replaced by high-speed devices as shown below.  
When you want to replace your low speed devices with high-speed devices, read the replacement notice given on the next pages.

High-speed device (New)	Low-speed device (Old)	Remarks
M80C85AH	M80C85A/M80C85A-2	8bit MPU
M80C86A-10	M80C86A/M80C86A-2	16bit MPU
M80C88A-10	M80C88A/M80C88A-2	8bit MPU
M82C84A-2	M82C84A/M82C84A-5	Clock generator
M81C55-5	M81C55	RAM.I/O, timer
M82C37B-5	M82C37A/M82C37A-5	DMA controller
M82C51A-2	M82C51A	USART
M82C53-2	M82C53-5	Timer
M82C55A-2	M82C55A-5	PPI

**Differences between MSM80C88A-10 and MSM80C88A-2, MSM80C88A****1) Manufacturing Process**

All devices use a 1.5  $\mu$  Si-CMOS process technology.

**2) Design**

Although circuit timings of these devices are a little different, these devices have the same chip size and logics.

**3) Electrical Characteristics**

Oki's '96 Data Book for MICROCONTROLLER describes that the MSM80C88A-10 satisfies the electrical characteristics of the MSM80C88A-2 and MSM80C88A.

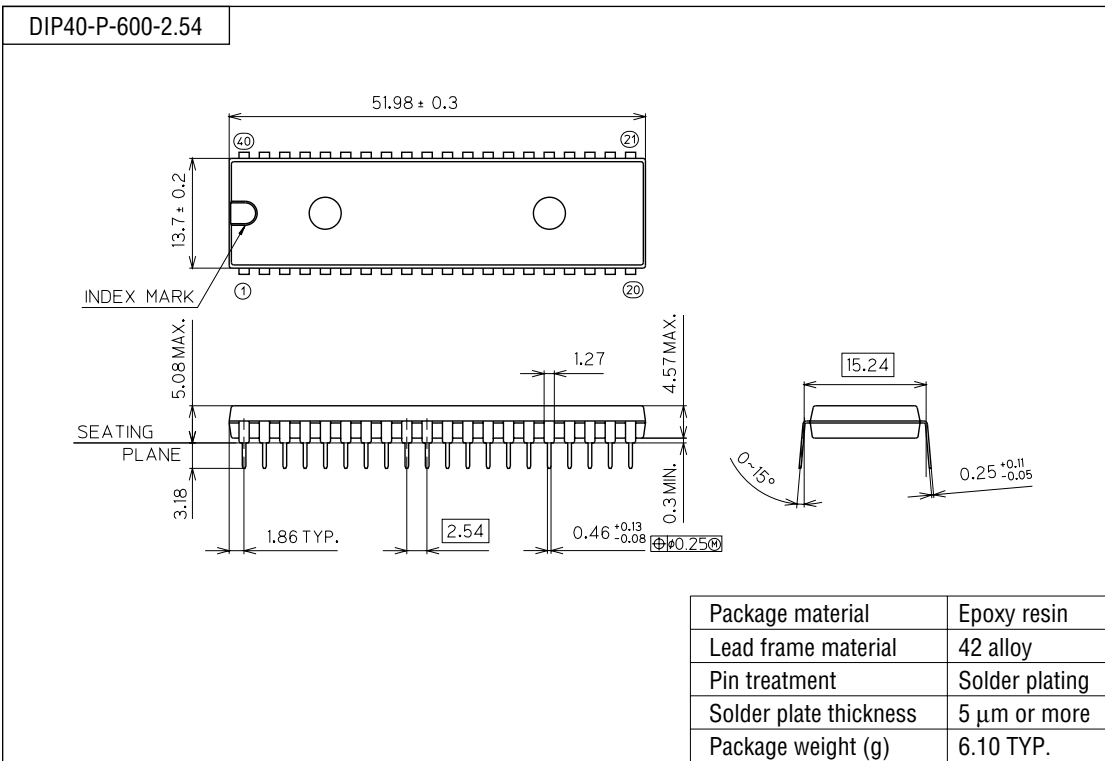
**4) Other notices**

1) The noise characteristics of the high-speed MSM80C88A-10 (for 10 MHz) are a little different from those of the MSM80C88A-2 and MSM80C88A. Therefore when devices are replaced for upgrading, it is recommended to perform noise evaluation.

2) The characteristics of the MSM80C88A-10 basically satisfy those of the MSM80C88A-2 and MSM80C88A but their timings are a little different. When critical timing is required in designing it is recommended to evaluate operating margins at various temperatures and voltages.

# PACKAGE DIMENSIONS

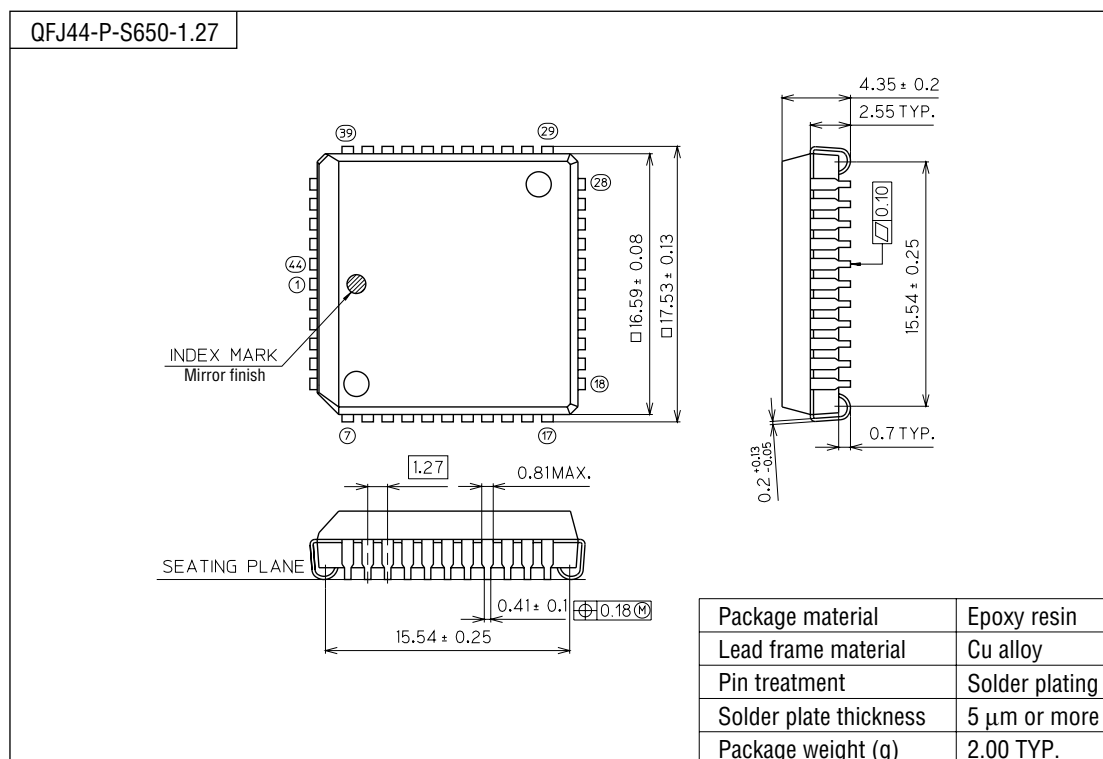
(Unit : mm)



## Notes for Mounting the Surface Mount Type Package

The SOP, QFP, TSOP, SOJ, QFJ (PLCC), SHP and BGA are surface mount type packages, which are very susceptible to heat in reflow mounting and humidity absorbed in storage. Therefore, before you perform reflow mounting, contact Oki's responsible sales person for the product name, package name, pin number, package code and desired mounting conditions (reflow method, temperature and times).

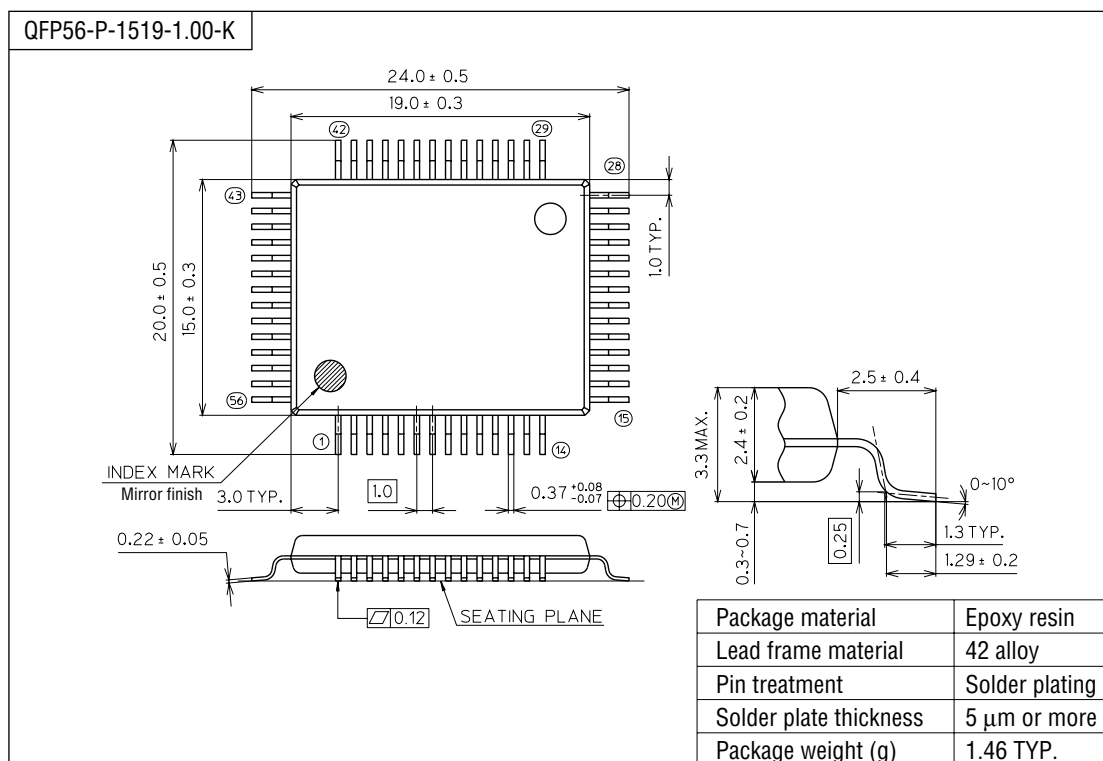
(Unit : mm)



### Notes for Mounting the Surface Mount Type Package

The SOP, QFP, TSOP, SOJ, QFJ (PLCC), SHP and BGA are surface mount type packages, which are very susceptible to heat in reflow mounting and humidity absorbed in storage. Therefore, before you perform reflow mounting, contact Oki's responsible sales person for the product name, package name, pin number, package code and desired mounting conditions (reflow method, temperature and times).

(Unit : mm)



### Notes for Mounting the Surface Mount Type Package

The SOP, QFP, TSOP, SOJ, QFJ (PLCC), SHP and BGA are surface mount type packages, which are very susceptible to heat in reflow mounting and humidity absorbed in storage.

Therefore, before you perform reflow mounting, contact Oki's responsible sales person for the product name, package name, pin number, package code and desired mounting conditions (reflow method, temperature and times).