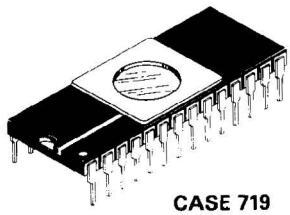
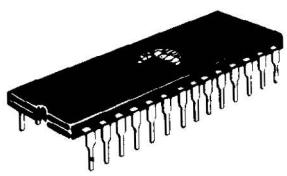
MC1468705F2

Advance Information

8-Bit EPROM Microcomputer Unit



CASE 719
CERAMIC



CASE 733A
CERDIP

The MC1468705F2 Microcomputer Unit (MCU) is an EPROM member of the M146805 CMOS Family of low-cost, single-chip microcomputers. The user programmable EPROM allows program changes and lower volume applications in comparison to the factory mask programmable versions. The EPROM versions also reduce the development costs and turnaround time for prototype evaluation of the mask ROM versions. This 8-bit microcomputer contains an on-chip oscillator, a CPU, RAM, EPROM, self-programming bootstrap ROM, I/O, and a timer.

In addition to power-saving STOP and WAIT modes, fully static design allows operation at frequencies down to dc, further reducing its already low power consumption. It is a low-power processor designed for low-end to mid-range applications in the consumer, automotive, industrial, and communications markets where very low power consumption is an important factor.

The following are the major features of the MC1468705F2 EPROM MCU.

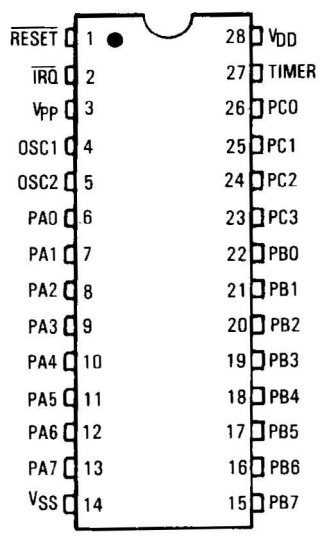
Hardware Features

- Typical Full Speed Operating Power of 10 mW at 5 V
- Typical WAIT Mode Power of 3 mW
- Typical STOP Mode Power of 40 μW
- 8-Bit Architecture
- Fully Static Operation
- 64 Bytes of On-Chip RAM
- 1079 Bytes of UV Erasable, User Programmable ROM (EPROM)
- 10 Bytes User Programmable Reset/Interrupt Vectors
- 16 Bidirectional I/O Lines
- Four Input-Only Lines
- Internal 8-Bit Timer with Software Programmable 7-Bit Prescaler
- External Timer Input
- External and Timer Interrupts
- Master Reset and Power-On Reset
- Single 3.0- to 5.5-Volt Supply
- On-Chip Oscillator with RC or Crystal Mask Options Selected by Mask Option Register
- Plug-In Compatible with the MC146805F2 Mask ROM Version
- Self-Programming Bootstrap Program in ROM Simplifies EPROM Programming
- Oscillator Internally Divided by Two or Four Selected by Mask Option Register
- Interrupts Triggered by Edge-Sensitive Only or Level- and Edge-Sensitive EPROM Programmable Option Selected by Mask Option Register

Software Features

- Similar to the MC6800
- Efficient Use of Program Space
- Versatile Interrupt Handling
- True Bit Manipulation
- Addressing Modes with Indexed Addressing for Tables
- Efficient Instruction Set
- Memory Mapped I/O
- Two Power Saving Standby Modes: WAIT and STOP
- Fully Compatible with M146805 CMOS Family Microcomputers

PIN ASSIGNMENT



This document contains information on a new product. Specifications and information herein are subject to change without notice.

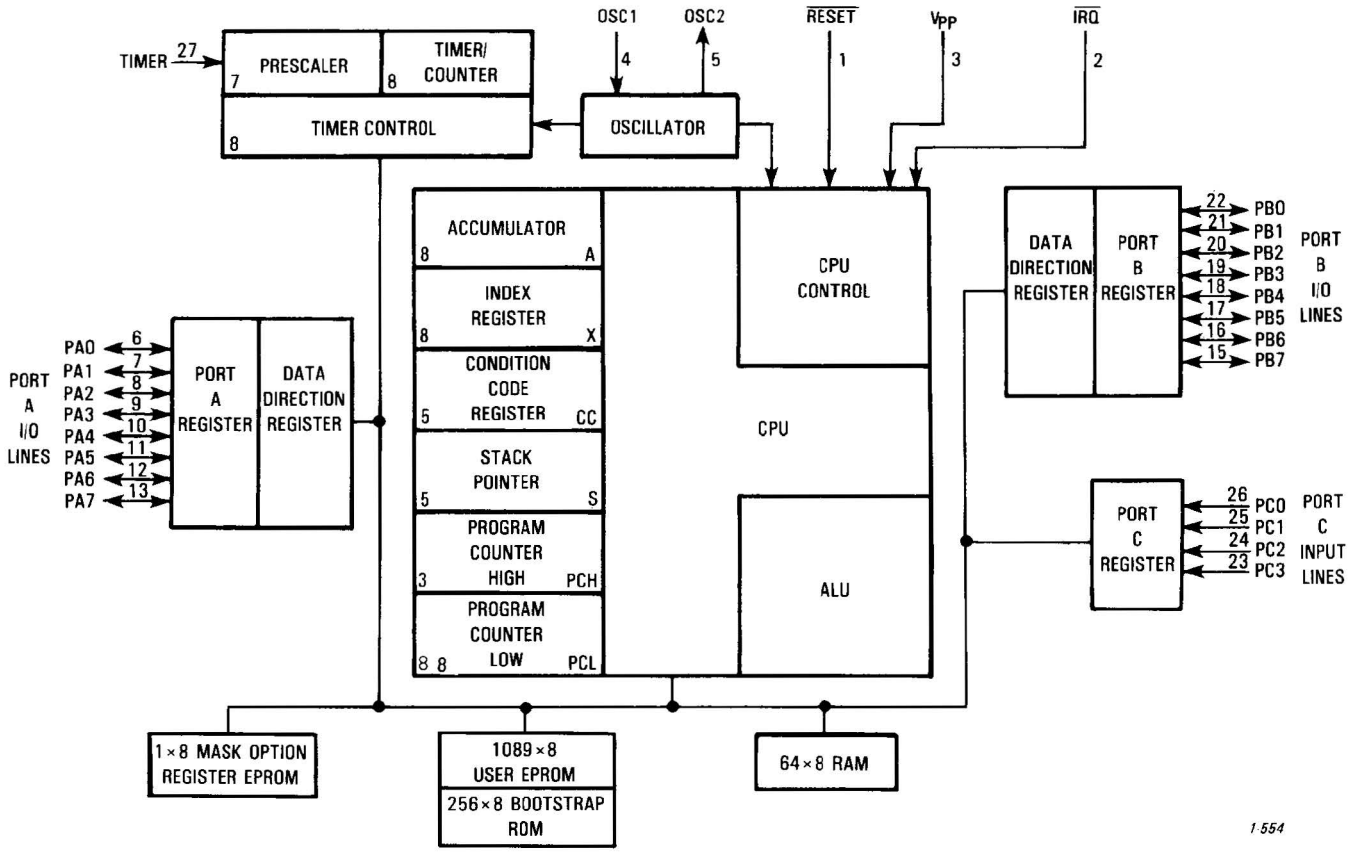


Figure 1. Block Diagram

MAXIMUM RATINGS (Voltages Referenced to V_{SS})

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	−0.3 to +8.0	V
All Input Voltages except OSC1	V_{in}	V_{SS} −0.5 to V_{DD} +0.5	V
EPROM Programming Voltage (V_{PP} Pin)	V_{in}	−0.3 to −13.5	V
Current Drain Per Pin Excluding V_{DD}, V_{SS}, and V_{PP} V_{PP}	I	 10 −30	mA
Operating Temperature Range MC1468705F2 MC1468705F2C	T_A	T_L to T_H 0 to 70 −40 to +85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages of electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high impedance circuit. For proper operation it is recommended the V_{in} and V_{out} be constrained to the range $V_{SS} \le (V_{in}$ or $V_{out}) \le V_{DD}$. Reliability of operation is enhanced if unused inputs except OSC2 and V_{PP} are connected to an appropriate logic voltage level (e.g., either V_{SS} or V_{DD}). Be sure that the EPROM window is shielded from light with an opaque cover at all times except when erasing.

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Ceramic Cerdip	θ_{JA}	 60 60	°C/W

PROGRAMMING OPERATION ELECTRICAL CHARACTERISTICS

(V_{DD} = 5.0 Vdc ± 5%, V_{SS} = 0 Vdc, T_A = 25°C unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Programming Voltage (V_{PP} Pin)	V_{PP}	− 12.5	− 13.0	− 13.5	V
V_{PP} Supply Current V_{PP} = − 13.5 V Maximum	I_{PP}	—	30	—	mA
Programming Oscillator Frequency	f_{oscp}	0.9	1.0	1.1	MHz
Bootstrap Programming Mode Voltage (PC0 Pin) I_{in} = 100 μA Maximum	V_{IRQP}	—	− 12.0	—	V
TIMER Pin Programming Mode Voltage	V_{TIMP}	—	V_{DD}	—	V

DC ELECTRICAL CHARACTERISTICS (V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, V_{PP} = 0 Vdc, T_A = T_L to T_H unless otherwise noted)

Characteristic	Symbol	Min	Max	Unit
Output Voltage, I_{Load} = 10.0 μA	V_{OL} V_{OH}	— V_{DD} − 0.1	0.1 —	V
Output High Voltage (I_{Load} = − 200 μA) PA0-PA7, PB0-PB7	V_{OH}	4.1	—	V
Output Low Voltage (I_{Load} = 800 μA) PA0-PA7, PB0-PB7	V_{OL}	—	0.4	V
Input High Voltage PA0-PA7, PB0-PB7, PC0-PC3 TIMER, $\overline{IRQ}$, $\overline{RESET}$, OSC1	V_{IH}	 V_{DD} − 2.0 V_{DD} − 0.8	 V_{DD} V_{DD}	V
Input Low Voltage All Inputs Except V_{PP} (See Note 2)	V_{IL}	V_{SS}	0.8	V
Input Low Voltage V_{PP} (Normal Operating Mode)	V_{IL}	V_{SS}	V_{SS}	V
Total Supply Current (C_L = 50 pF on Ports, no dc Loads, t_{cyc} = 1 μs) RUN (V_{IL} = 0.2 V, V_{IH} = V_{DD} − 0.2 V) WAIT (See Note 1) STOP (See Note 1)	I_{DD}	 — — —	 4 1.5 150	 mA mA μA
I/O Ports Input Leakage PA0-PA7, PB0-PB7	I_{IL}	—	± 10	μA
Input Current $\overline{RESET}$, $\overline{IRQ}$, TIMER, OSC1, PC0-PC3	I_{in}	—	± 1	μA
Capacitance Output Ports $\overline{RESET}$, $\overline{IRQ}$, TIMER, OSC1, PC0-PC3	 C_{out} C_{in}	 — —	 12 8	pF

NOTES:

1. Test conditions for I_{DD} are as follows:
   a. All ports programmed as inputs
   b. V_{IL} = 0.2 V (PA0-PA7, PB0-PB7, PC0-PC3)
   c. V_{IH} = V_{DD} − 0.2 V for $\overline{RESET}$, $\overline{IRQ}$, and TIMER
   d. OSC1 input is a square wave from 0.2 V to V_{DD} − 0.2 V
   e. OSC2 output load = 20 pF (WAIT I_{DD} is affected linearly by the OSC2 capacitance)
2. V_{PP} is pin 3 on the MC1468705F2 and is connected to V_{SS} in the normal operating mode.

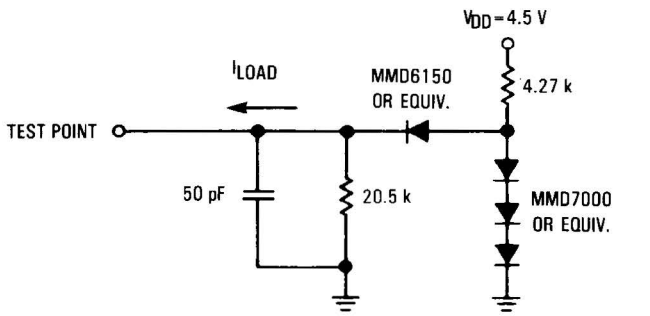
Table 1. Control Timing

(V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)

Characteristic	Symbol	Min	Max	Unit
Crystal Oscillator Startup Time (see Figure 5)	t_{OXOV}	—	100	ms
Stop Recovery Startup Time (Crystal Oscillator) (see Figure 6)	t_{ILCH}	—	100	ms
Timer Pulse Width (see Figure 4)	t_{TH}, t_{TL}	0.5	—	t_{cyc}
$\overline{\text{RESET}}$ Pulse Width (see Figure 5)	t_{RL}	1.5	—	t_{cyc}
Timer Period (see Figure 4)	t_{TLTL}	1.0	—	t_{cyc}
Interrupt Pulse Width Low (see Figure 14)	t_{ILIH}	1.0	—	t_{cyc}
Interrupt Pulse Period (see Figure 14)	t_{ILIL}	*	—	t_{cyc}
OSC1 Pulse Width	t_{OH}, t_{OL}	100	—	ns
Cycle Time	t_{cyc}	1000	—	ns
Frequency of Operation	f_{osc}			MHz
Crystal (÷ 4 Option)		—	4.0	
External Clock (÷ 4 Option)		dc	4.0	
Crystal (÷ 2 Option)		—	2.0	
External Clock (÷ 2 Option)		dc	2.0	

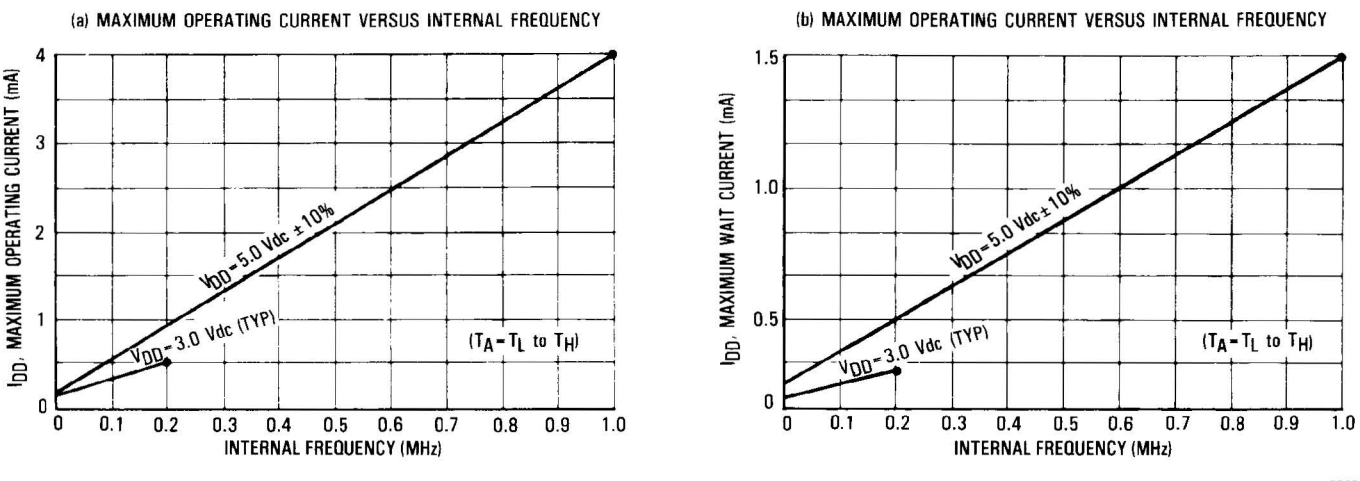
*The minimum period t_{ILIL} should not be less than the number of t_{cyc} cycles it takes to execute the interrupt service routines plus 20 t_{cyc} cycles.

1-564



1-555

Figure 2. Equivalent Test Load



1-556A 1-556B

Figure 3. Operating Current versus Internal Frequency

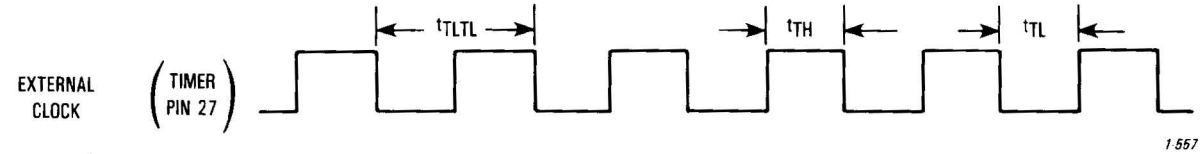


Figure 4. Timer Relationships

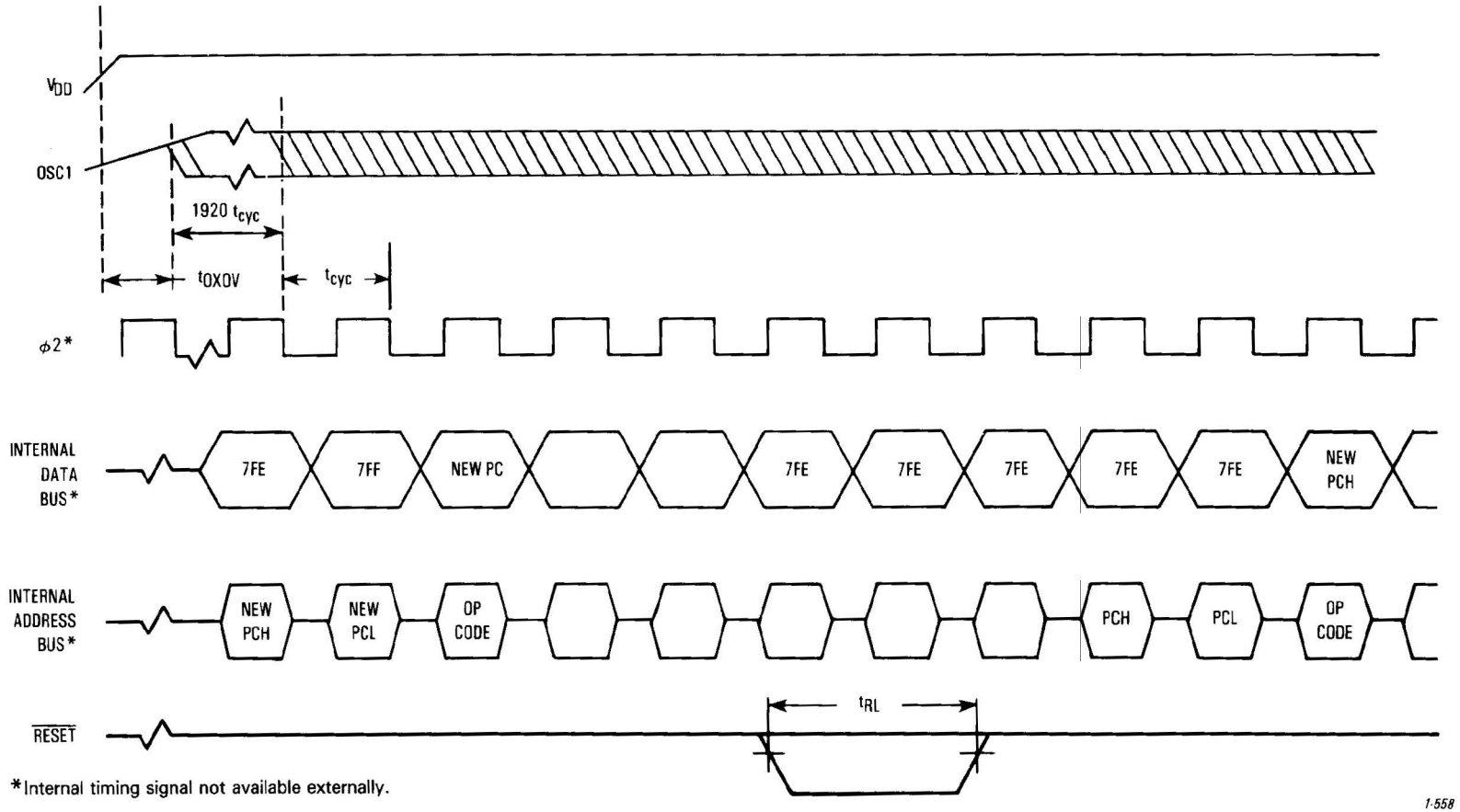
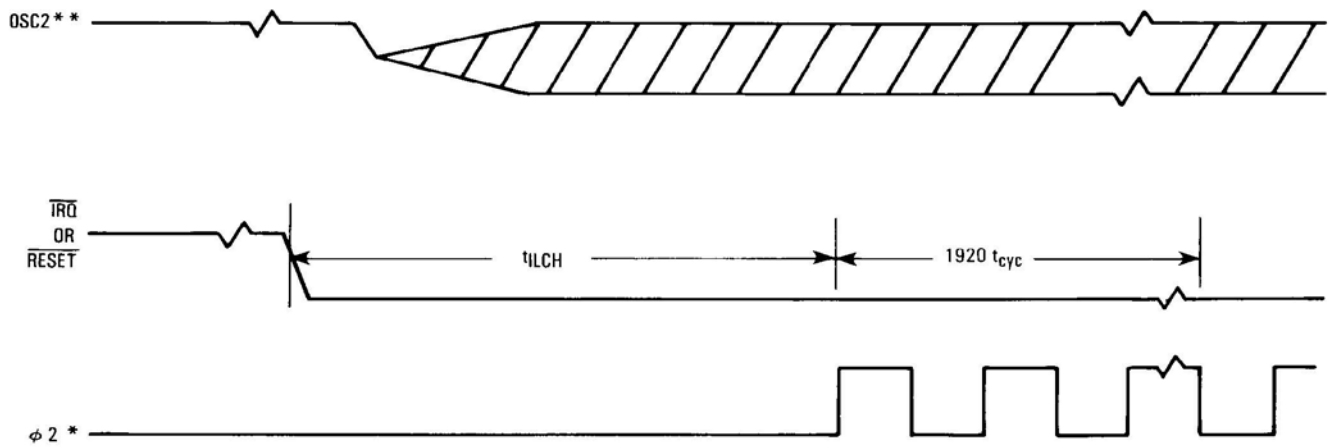


Figure 5. Power-On Reset and $\overline{\text{RESET}}$



*Internal timing signals not available externally.
**Represents the internal gating of the OSC1 input pin.

1-559

Figure 6. Stop Recovery and Power-On Reset

FUNCTIONAL PIN DESCRIPTION

V_{DD} and V_{SS}

Power is supplied to the MCU using these two pins. The V_{DD} pin is power and the V_{SS} pin is ground.

$\overline{IRQ}$ (MASKABLE INTERRUPT REQUEST)

The $\overline{IRQ}$ pin is a programmable option which provides two different choices of interrupt triggering sensitivity. These options are: 1) negative edge-sensitive triggering only, or 2) both negative-edge sensitive and level-sensitive triggering. In the latter case, either type of input to the $\overline{IRQ}$ pin will produce the interrupt.

The MCU completes the current instruction before it responds to the interrupt request. When the $\overline{IRQ}$ pin goes low for at least one t_{CYC}, a logic one is latched internally to signify an interrupt has been requested. When the MCU completes its current instruction, the interrupt latch is tested. If the interrupt latch contains a logic one, and the interrupt mask bit (I bit) in the condition code register is clear, the MCU then begins the interrupt sequence.

If the option is selected to include level-sensitive triggering, then the $\overline{IRQ}$ input requires an external resistor to V_{DD} for "wire-OR" operation. See **INTERRUPTS** for more detail. This pin also detects a negative voltage that is used to initiate the bootstrap mode program.

$\overline{RESET}$

The $\overline{RESET}$ input is not required for startup but can be used to reset the MCU internal state and provide an orderly software startup procedure. Refer to **RESETS** for a detailed description.

TIMER

The TIMER input may be used as an external clock for the on-chip timer. This pin is connected to V_{DD} for the bootstrap mode (EPROM programming). Refer to **TIMER** for additional information about the timer circuitry.

V_{PP}

The V_{PP} pin is used when programming the EPROM. By applying the negative programming voltage to this pin, one of the requirements is met for programming the EPROM. Refer to **PROGRAMMING FIRMWARE** and the **PROGRAMMING OPERATION ELECTRICAL CHARACTERISTICS** table.

NOTE

In normal operation, this pin is connected directly to V_{SS}.

OSC1, OSC2

The MC1468705F2 can be configured to accept either a crystal input or an RC network to control the internal oscillator. Additionally, the internal clocks can be derived by either a divide-by-two or divide-by-four of the internal oscillator frequency (f_{OSC}). Both of these options are programmable via the mask option register (MOR) in the EPROM array.

RC Oscillator

If the RC oscillator option is selected, then a resistor is connected to the oscillator pins as shown in Figure 7(c). The relationship between R and f_{OSC} is shown in Figure 8.

Crystal

The circuit shown in Figure 7(b) is recommended when using a crystal. The internal oscillator is designed to interface with the AT-cut parallel resonant quartz crystal resonator in the frequency range specified for f_{osc} in Table 1 Control Timing. Using an external CMOS oscillator is recommended when crystals outside the specified ranges are to be used. The crystal and components should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time. Refer to **MAXIMUM RATINGS** for V_{DD} specifications.

External Clock

An external clock should be applied to the OSC1 input with the OSC2 input not connected, as shown in Figure 7(d). An external clock may only be used with the crystal oscillator option selected in the mask option register. The t_{OXOV} or t_{ILCH} specifications do not apply when using an external clock input.

PA0-PA7

These eight I/O lines comprise port A. The state of any pin is software programmable. Refer to **PROGRAMMING** for a detailed description of I/O programming.

PB0-PB7

These eight lines comprise port B. The state of any pin is software programmable. Refer to **PROGRAMMING** for a detailed description of I/O programming.

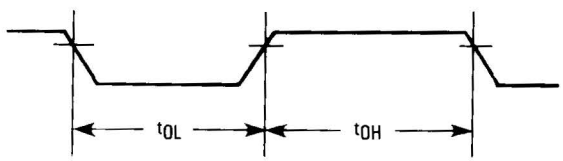
PC0-PC3

These four lines comprise port C, a fixed input port. When port C is read, the four most significant bits on the data bus are ones. There is no data direction register associated with port C.

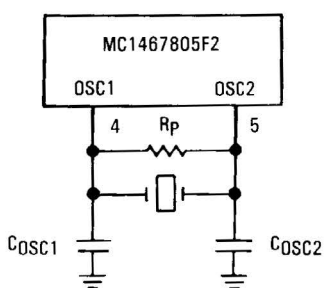
CRYSTAL PARAMETERS

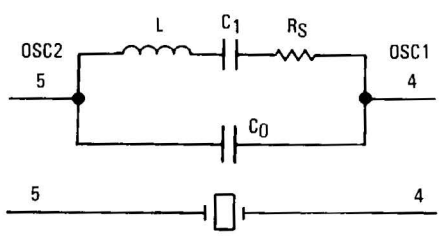
	1 MHz	4 MHz	Units
R_{SMAX}	400	75	Ω
C_0	5	7	pF
C_1	0.008	0.012	μF
C_{OSC1}	15-40	15-30	pF
C_{OSC2}	15-30	15-25	pF
R_P	10	10	MΩ
Q	30 k	40 k	–

(a) OSCILLATOR WAVEFORM

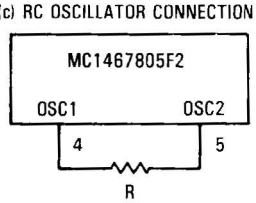


(b) CRYSTAL OSCILLATOR CONNECTIONS AND EQUIVALENT CRYSTAL CIRCUIT



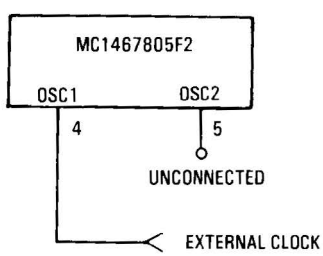


(c) RC OSCILLATOR CONNECTION*



*Approximately 10% to 25% accuracy (excludes resistor tolerance) external register

(d) EXTERNAL CLOCK SOURCE CONNECTIONS



1-560

Figure 7. Oscillator Connections

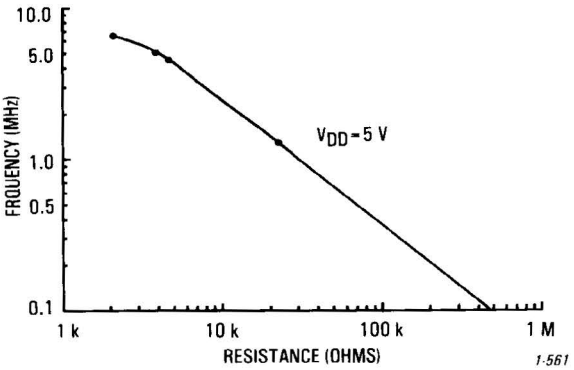


Figure 8. Typical Frequency versus Resistance for RC Oscillator Option only

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

Any port A or port B pin may be software programmed as an input or output by the state of the corresponding bit in the port data direction register (DDR). A port A or port B pin is configured as an output if its corresponding DDR bit is set. A pin is configured as an input if its corresponding DDR bit is cleared. At reset, all DDRs are cleared, which configures all port A and B pins as inputs. Port C is input only. A port A or B pin configured as an output will output the data in the corresponding bit of its port data latch. Refer to Figure 9 and Table 2.

EPROM PROGRAMMING

When programming the EPROM array within the MC1468705F2, ports are used in a special arrangement. See **PROGRAMMING FIRMWARE** (and Figure 20) for a detailed description.

MEMORY

As shown in Figure 10, the MCU is capable of addressing 2048 bytes of memory and I/O registers with its program counter. The MC1468705F2 MCU has implemented 1417 bytes of these locations. This consists of: 1079 bytes of user EPROM, 10 bytes of user programmable vectors, 256 bytes of bootstrap ROM, 64 bytes of user RAM, an EPROM mask option register (MOR), five bytes of I/O, and two timer registers.

The user EPROM is located in two areas. The main EPROM area is memory locations $080 to $4B6. The second area is reserved for ten interrupt/reset vector bytes at memory locations $7F6 through $7FF. The mask option register at memory location $7F5 completes the total EPROM area. The MCU uses 10 of the lowest 16 memory locations for program control and I/O features such as data ports, the port DDRs, and the timer. The 64 bytes of user RAM include up to 32 bytes for the stack. Except for the MOR, the memory mapping is identical to the MC146805F2; however, the MC1468705F2 has no self-check ROM because of the bootstrap ROM requirement.

The stack area is used during the processing of interrupt and subroutine calls to save the processor state. The contents of the CPU registers are pushed onto the stack in the order shown in Figure 12. The stack pointer decrements during pushes and the low order byte (PCL) of the program counter is stacked first; then the higher order three bits (PCH) are stacked. The stack pointer increments when it pulls data from the stack. A subroutine call causes only the program counter (PCL, PCH) contents to be pushed onto the stack; the remaining CPU registers are not pushed.

REGISTERS

The MC1468705F2 contains five registers, as shown in the programming model of Figure 11. The interrupt stacking order is shown in Figure 12.

ACCUMULATOR (A)

The accumulator is an 8-bit general purpose register used to hold operands, results of the arithmetic calculations, and data manipulations.

INDEX REGISTER (X)

The X register is an 8-bit register which is used during the indexed modes of addressing. It provides an 8-bit value which is used to create an effective address. The index register is also used for data manipulations with the read-modify-write type of instructions and as a temporary storage register when not performing addressing operations.

PROGRAM COUNTER (PC)

The program counter is an 11-bit register that contains the address of the next instruction to be executed by the processor.

STACK POINTER (SP)

The stack pointed is an 11-bit register containing the address of the next free location on the stack. When accessing memory, the six most significant bits are permanently configured to 000011. These six bits are appended to the five least significant register bits to produce an address within the range of $07F to $060.

The stack area of RAM is used to store the return address on subroutine calls and the machine state during interrupts. During external or power-on reset and during a reset stack pointer (RSP) instruction, the stack pointer is set to its upper limit ($07F). Nested interrupt and/or subroutines may use up to 32 (decimal) locations, beyond which the stack pointer wraps around and points to its upper limit; thereby, losing the previously stored information. A subroutine call occupies two RAM bytes on the stack, while an interrupt uses five RAM bytes.

CONDITION CODE REGISTER (CC)

The condition code register is a 5-bit register which indicates the results of the instruction just executed. These bits can be individually tested by a program and specified action taken as a result of their state. Each bit is explained in the following paragraphs.

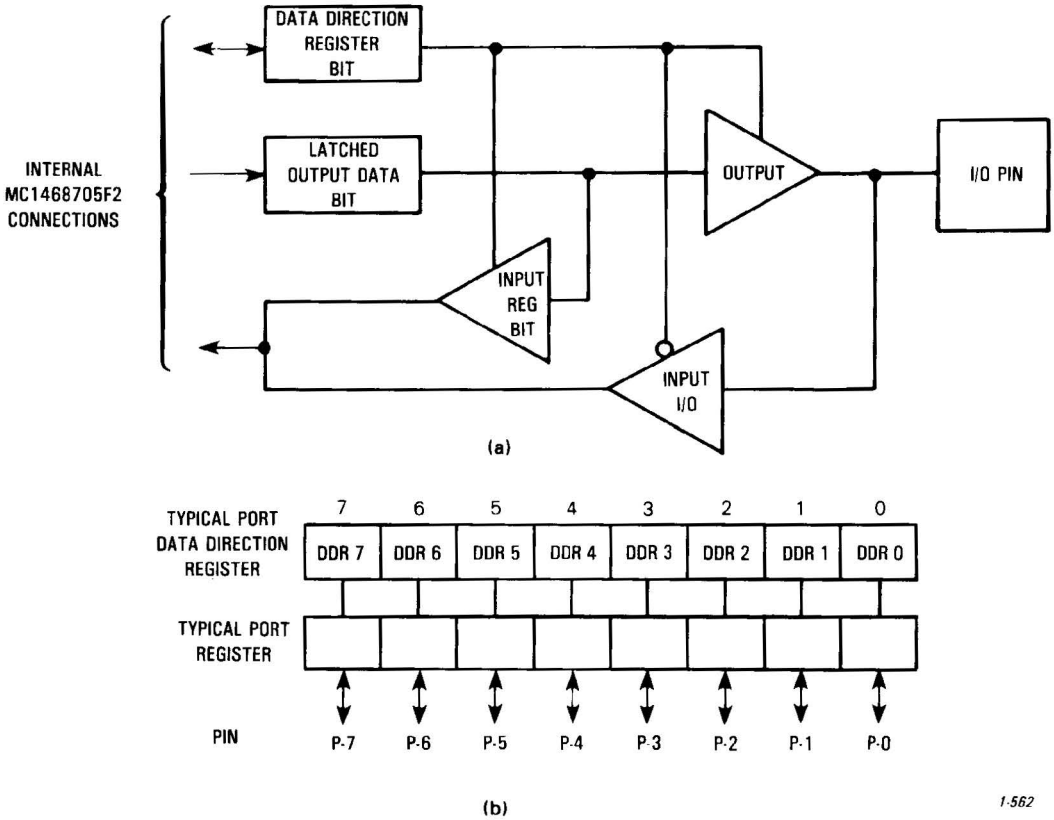


Figure 9. Typical Port A or B I/O Circuitry

Table 2. Port A or B I/O Pin Functions

R/$\overline{W}$*	DDR	I/O Pin Function
0	0	The I/O pin is in input mode. Data is written into the output data latch.
0	1	Data is written into the output data latch and output to the I/O pin.
1	0	The state of the I/O pin is read.
1	1	The I/O pin is in an output mode. The output data latch is read.

*R/$\overline{W}$ is an internal signal.

1-563

Half Carry Bits (H)

The H bit is set to a one when a carry occurs between bits 3 and 4 of the ALU during an ADD or ADC instruction. The H bit is useful in binary coded decimal subroutines.

Interrupt Mask Bit (I)

When the I bit is set, both the external interrupt and the timer interrupt are disabled. Clearing this bit enables the above interrupts. If an interrupt occurs while the I bit is set, the interrupt is latched and is processed after the I bit is next cleared.

Negative (N)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation is negative (bit 7 in the result is a logic one).

Zero (Z)

When set, this bit indicates that the result of the last arithmetic, logical, or data manipulation is zero.

Carry/Borrow (C)

Indicates that a carry or borrow out of the arithmetic logic unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, shifts, and rotates.

RESETS

The MC1468705F2 has two reset modes: an active low external reset pin ($\overline{RESET}$) and a power-on reset function; refer to Figure 5.

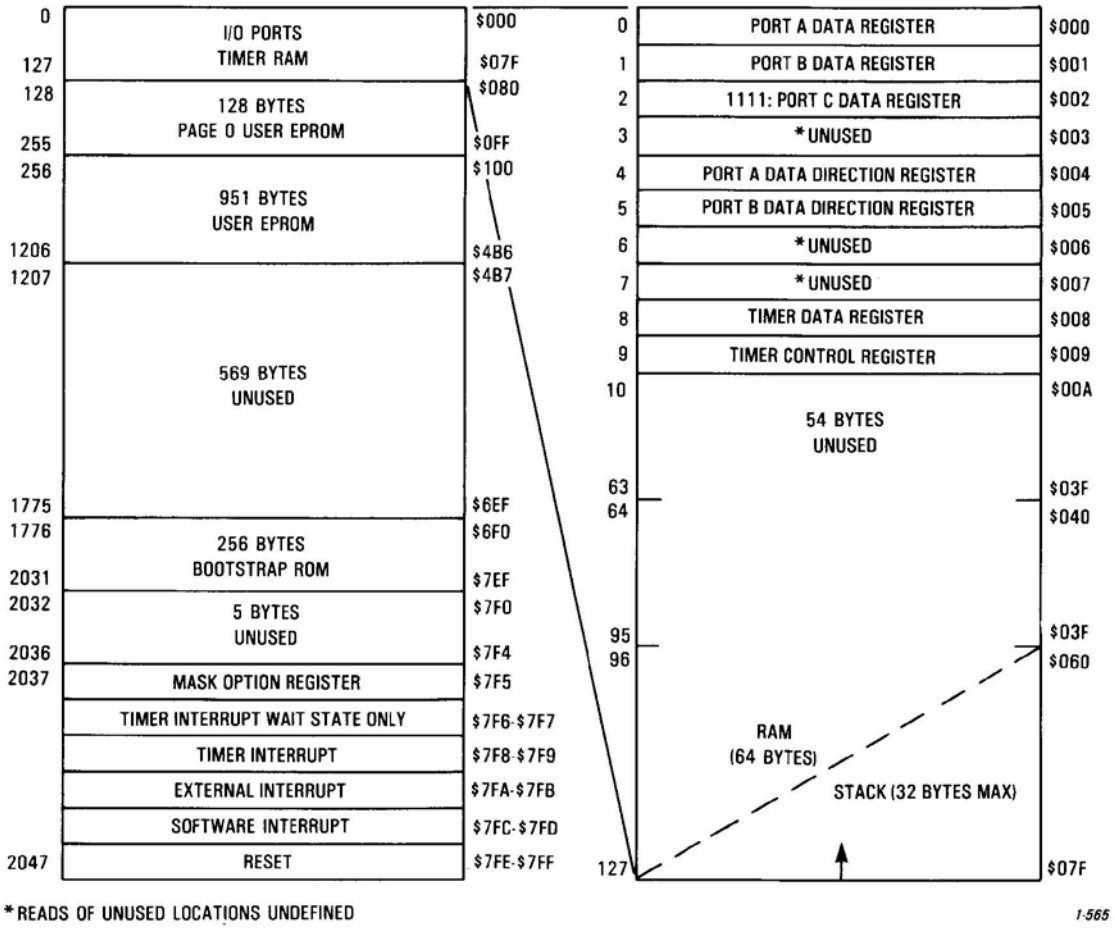


Figure 10. Address Map

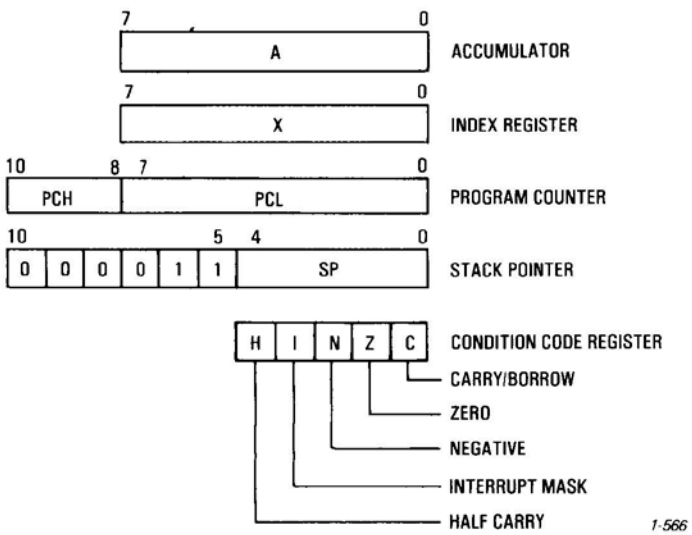
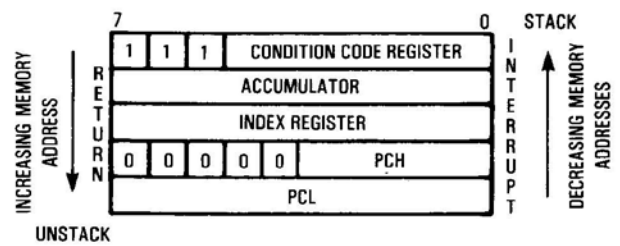


Figure 11. Programming Model



NOTE: Since the stack pointer decrements during pushes, the PCL is stacked first, followed by PCH, etc. Pulling from the stack is in the reverse order.

1-567

Figure 12. Stacking Order

$\overline{\text{RESET}}$

The $\overline{\text{RESET}}$ input pin is used to reset the MCU to provide an orderly software startup procedure. When using the external reset mode, the $\overline{\text{RESET}}$ pin must stay low for a minimum of one t_{CYC}. The $\overline{\text{RESET}}$ pin contains an internal Schmitt Trigger as part of its input to improve its noise immunity.

POWER-ON RESET

The power-on reset occurs when a positive transition is detected on V_{DD}. The power-on reset is used strictly for power turn-on conditions and should not be used to detect any drops in the power supply voltage. There is no provision for a power-down reset. The power-on circuitry provides for a 1920 t_{CYC} delay from the time that the oscillator becomes active. If the external $\overline{\text{RESET}}$ pin is low at the end of the 1920 t_{CYC} time out, the processor remains in the reset condition until $\overline{\text{RESET}}$ goes high.

Either of the two types of reset conditions causes the following to occur:

1) Timer control register interrupt request bit TCR7 is cleared to a logic zero to preclude premature timer interrupts.
2) Timer control register interrupt mask bit TCR6 is set to a logic one to preclude timer interrupt processing.
3) All data direction register bits are cleared to logic zeros to define all ports as input.
4) Stack pointer is preset to its upper limit, $07F.
5) The internal address bus is forced to the reset vector ($7FE).
6) Condition code register interrupt mask bit (I) is set to a logic one to mask any external interrupts.
7) STOP and WAIT latches are cleared to place MCU in normal operation.
8) External interrupt latch is cleared to ensure no external interrupt is processed.

All other functions, such as other registers (including I/O ports), the timer, mask option register, etc., are not cleared by the reset conditions.

BOOTSTRAP ROM

The bootstrap ROM contains a factory program which allows the MCU to present an address and fetch data from an external device and transfer it into the MC1468705F2 EPROM. The bootstrap program provides: timing of programming pulses, timing of Vpp input, and verfication after programming. See **PROGRAMMING FIRMWARE.**

MASK OPTION REGISTER (MOR)

The mask option register is an 8-bit user programmed (EPROM) register in which three of the bits are used. Bits in this register are used to select the type of system clock (crystal/RC oscillator), the divide-by-four or divide-by-two clock option (bus frequency), and the edge-sensitive or edge- and level-sensitive triggered interrupt recognition.

INTERRUPTS

Systems often require that normal processing be interrupted so that some external event may be serviced. The MC1468705F2 may be interrupted by one of three different methods: either one of two maskable hardware interrupts (external input or timer) or a nonmaskable software interrupt (SWI).

Interrupts cause the processor registers to be saved on the stack and the interrupt mask (I bit) to set preventing additional interrupts. The return from interrupt (RTI) instruction causes the register contents to be recovered from the stack followed by a return to normal processing. The stack order is shown in Figure 12.

Unlike $\overline{\text{RESET}}$, hardware interrupts do not cause the current instruction execution to be halted, but are considered pending until the current instruction execution is completed.

When the current instruction is complete, the processor checks all pending hardware interrupts and, if an interrupt is pending and is unmasked, proceeds with interrupt processing; otherwise, the next instruction is fetched and executed. Note that masked interrupts are latched for later interrupt servicing.

If both an external interrupt and a timer interrupt are pending at the end of an instruction execution, the external interrupt is serviced first. The SWI is executed the same as any other instruction. Refer to Figure 13 for the interrupt and instruction processing sequence.

Table 3 shows the execution priority of the $\overline{\text{RESET}}$, $\overline{\text{IRQ}}$, and timer interrupts, and instructions (including the software interrupt, SWI). Two conditions are shown, one with the I bit set and the other with the I bit clear; however, in either case $\overline{\text{RESET}}$ has the highest priority of execution.

If the I bit is set as per Table 3(a), the second highest priority is assigned to any instruction (including SWI). This is illustrated in Figure 13 which shows that the $\overline{\text{IRQ}}$ or TIMER interrupts are not executed when the I bit is set. If the I bit is clear as per Table 3(b), the priorities change in that the next instruction (including SWI) is not fetched until after the $\overline{\text{IRQ}}$ and TIMER interrupts have been recognized (and serviced). Also, when the I bit is clear, if both $\overline{\text{IRQ}}$ and TIMER interrupts are pending, the $\overline{\text{IRQ}}$ interrupt is always serviced before the TIMER interrupt.

NOTE

The conditions for Table 3 assume that, except for $\overline{\text{RESET}}$, the current instruction is completed, thus the MCU is at an instruction boundary.

Table 3. Interrupt Instruction Execution Priority and Vector Address

(a) I Bit Set

Interrupt/Instruction	Priority	Vector Address
$\overline{\text{RESET}}$	1	$7FE-$7FF
SWI (or other instruction)	2	$7FC-$7FD

NOTE: $\overline{\text{IRQ}}$ and timer interrupts are not executed when the I bit is set; therefore, they are not shown.

(b) I Bit Clear

Interrupt/Instruction	Priority	Vector Address
$\overline{\text{RESET}}$	1	$7FE-$7FF
$\overline{\text{IRQ}}$	2	$7FA-$7FB
	3	$7F8-$7F9
		$7F6-$7F7*
SWI (or other instruction)	4	$7FC-$7FD

*The timer vector address from the WAIT mode is $7F6-$7F7.

1-568

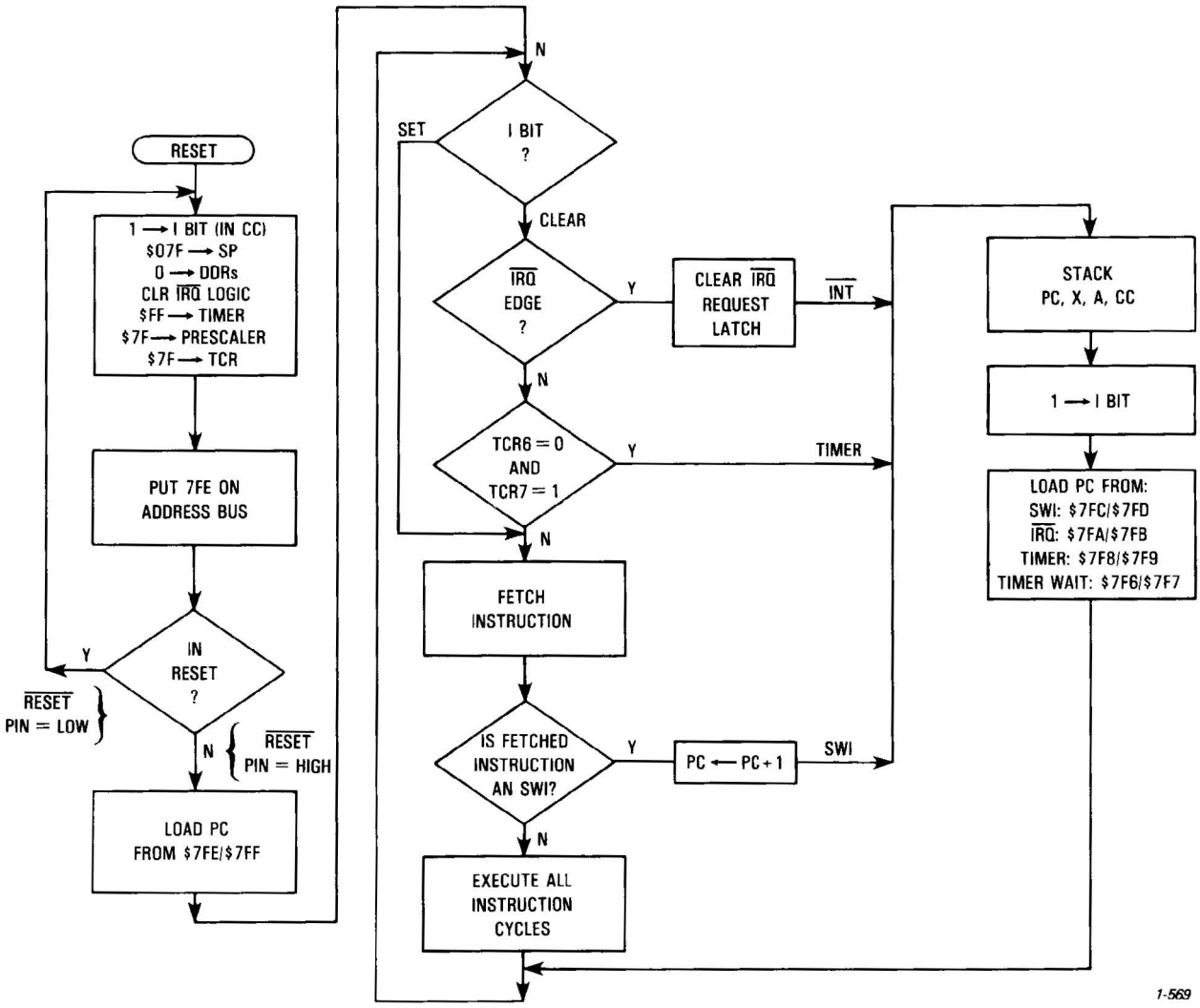


Figure 13. Reset and Interrupt Processing Flowchart

TIMER INTERRUPT

If the timer interrupt mask bit (TCR6) is cleared, then each time the timer decrements to zero (transitions from $01 to $00 to set TCR7) an interrupt request is generated. The actual processor interrupt is generated only if the interrupt mask bit (in the condition code register) is cleared. When the interrupt is recognized, the current state of the machine is pushed onto the stack and the interrupt mask bit in the condition code register is set. This masks further interrupts until the present one is serviced.

The processor now vectors to the timer interrupt service routine. The address for this service routine is specified by the contents of $7F8 and $7F9 unless the processor is in a WAIT mode in which case the contents of $7F6 and $7F7 specify the timer service routine address. Software must be used to clear the timer interrupt request bit (TCR7). At the end of the timer interrupt service routine, the software normally executes an RTI instruction which restores the machine state and starts executing the interrupted program.

The actual timer interrupt request can be delayed by controlling TCR6 (interrupt mask bit). If TCR6 is programmed to a logic one, no interrupt is generated even if TCR7 (interrupt request bit) is set. Then, TCR6 can be programmed (after a specific time) to a logic zero to generate the actual timer interrupt request.

EXTERNAL INTERRUPT

If the interrupt mask bit of the condition code register has been cleared and the external interrupt pin ($\overline{IRQ}$) is low, then the external interrupt is recognized. The action of the external interrupt is identical to the timer interrupt with the exception that the interrupt request input at $\overline{IRQ}$ is latched internally and the service routine address is specified by the contents of $7FA and $7FB.

Either a level-sensitive and edge-sensitive trigger, or an edge-sensitive only trigger are available as a mask option register (MOR) controlled programmable option. Figure 14

shows both a functional and mode timing diagram for the interrupt line. The timing diagram shows two different treatments of the interrupt line ($\overline{IRQ}$) to the processor. The first method shows single pulses on the interrupt line spaced far enough apart to be serviced. The minimum time between pulses is a function of the length of the interrupt service routine.

Once a pulse occurs, the next pulse should not occur until the MCU software has exited the routine (an RTI occurs). This time (t_{ILIL}) is obtained by adding 20 instruction cycles (t_{cyc}) to the total number of cycles it takes to complete the service routine; refer to Figure 14. The second configuration shows many interrupt lines "wire-ORed" to form the interrupts at the processor. Thus, if after servicing one interrupt the interrupt line remains low, then the next interrupt is recognized.

NOTE

The internal interrupt latch is cleared in the first part (read of vectors) of the service routine; therefore, one (and only one) external interrupt pulse could be latched during t_{ILIL} and serviced as soon as the I bit is cleared.

SOFTWARE INTERRUPT

The software interrupt (SWI) is an executable instruction. The action of the software interrupt instruction is similar to the hardware interrupts. The software interrupt is executed regardless of the state of the interrupt mask bit in the condition code register. The service routine address is specified by the contents of memory locations $7FC and $7FD. See Figure 13 for interrupt and instruction processing flowchart.

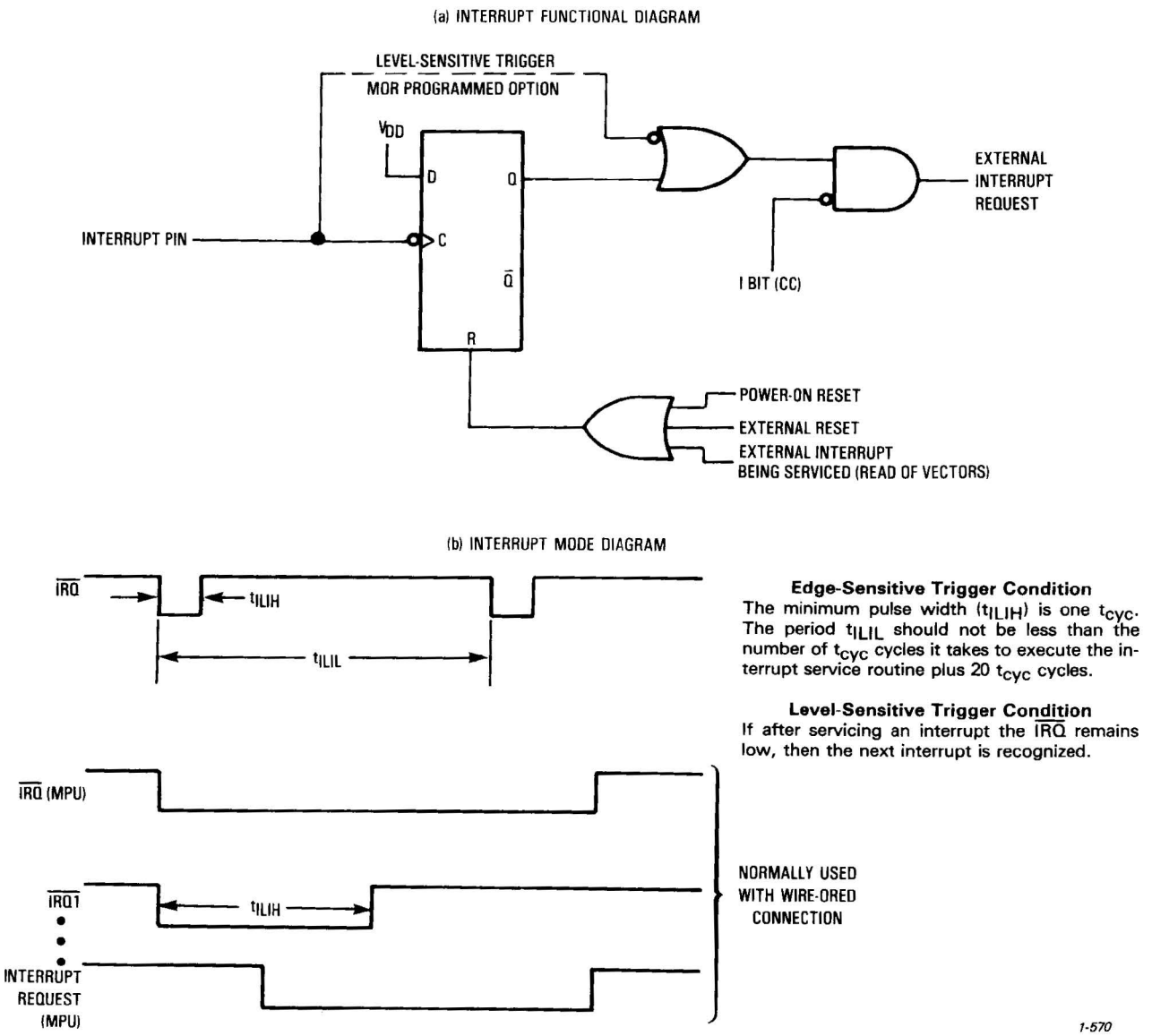


Figure 14. External Interrupt

LOW-POWER MODES

STOP

The STOP instruction places the MC1468705F2 in its lowest power consumption mode. In the STOP mode, the internal oscillator is turned off causing all internal processing and the timer to be halted; refer to Figure 15.

During the STOP mode, timer control register (TCR) bits 6 and 7 are altered to remove any pending timer interrupt requests and to disable any further timer interrupts. The timer prescaler is cleared. The I bit in the condition code register is cleared to enable external interrupts. All other registers and memory remain unaltered. All input/output lines remain unchanged. The processor can only be brought out of the STOP mode by an external interrupt or reset.

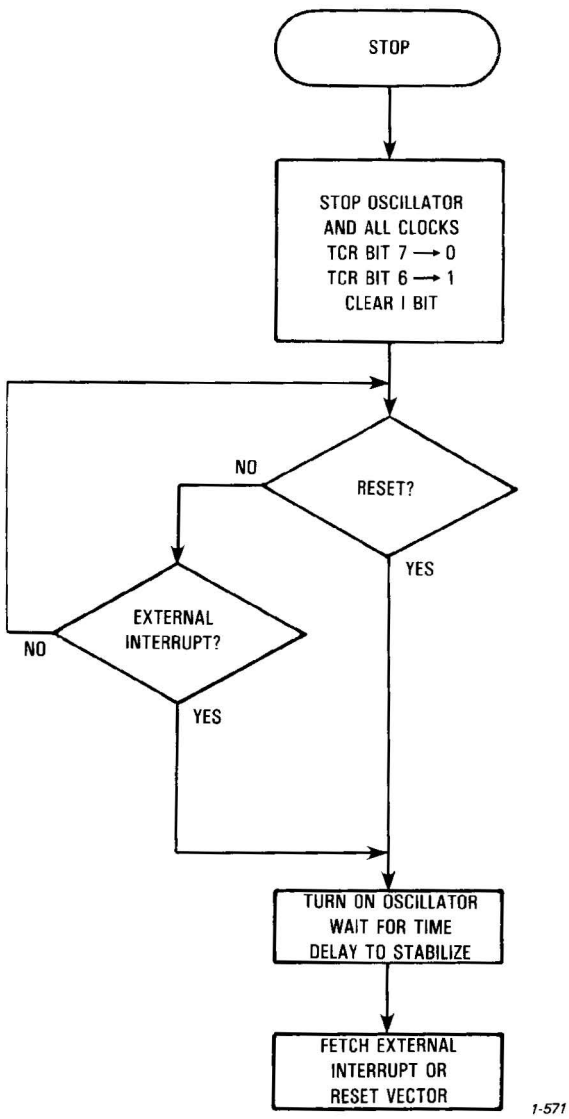


Figure 15. Stop Function Flowchart

WAIT

The WAIT instruction places the MC1468705F2 in a low power consumption mode, but the WAIT mode consumes somewhat more power than the STOP mode. In the WAIT mode, the internal clock is disabled from all internal circuitry except for the timer; refer to Figure 16. Thus, all internal processing is halted; however, the timer continues to count normally.

During the WAIT mode, the I bit in the condition code register is cleared to enable interrupts. All other registers, memory, and input/output lines remain in their previous state. The timer may be enabled to allow a periodic exit from the WAIT mode. If an external and a timer interrupt occur at the same time, the external interrupt is serviced first; then, if the timer interrupt request is not cleared in the external interrupt routine, the normal timer interrupt (not the timer wait interrupt) is serviced since the MCU is no longer in the WAIT mode.

MODES OF OPERATION

The MC1468705F2 has two modes of operation. These modes are the normal (single-chip) mode and the bootstrap mode (firmware used to program the EPROM). These two modes are entered as described below.

SINGLE-CHIP MODE

The normal operational mode of the MC1468705F2 is the single-chip mode. The single-chip mode will be entered if the following conditions are satisfied: 1) the $\overline{\text{RESET}}$ line is brought low, 2) the PC0 pin is within its normal operational range (V_{SS}-V_{DD}), and 3) the V_{PP} pin is connected to V_{SS}. The next rising edge of the $\overline{\text{RESET}}$ pin then causes the part to enter the single-chip mode.

BOOTSTRAP MODE

The bootstrap mode is entered if certain conditions are met on the TIMER, PC0, and $\overline{\text{RESET}}$ pins. A negative voltage (-12 V) must be present on the PC0 pin, and V_{DD} should be applied to the $\overline{\text{TIMER}}$ pin. Refer to Figure 17.

TIMER

The MCU timer contains an 8-bit software programmable counter (timer data register) with a 7-bit software selectable prescaler. Figure 18 shows a block diagram of the timer subsystem. The counter may be loaded under program control and decrements towards zero. When the counter decrements to zero, the timer interrupt request bit (i.e., bit 7 of the timer control register, TCR) is set. Then if the timer interrupt is not masked (i.e., bit 6 of the TCR and the I bit in the condition code register are both cleared), the processor receives an interrupt. After completion of the current instruction, the processor proceeds to store the appropriate registers on the stack, and then fetches the timer vector address from locations $7F8 and $7F9 (or $7F6 and $7F7 if in the WAIT mode) in order to begin servicing; refer to INTERRUPTS.

The counter continues to count after it reaches zero, allowing the software to determine the number of internal or external input clocks since the timer interrupt request bit was set.

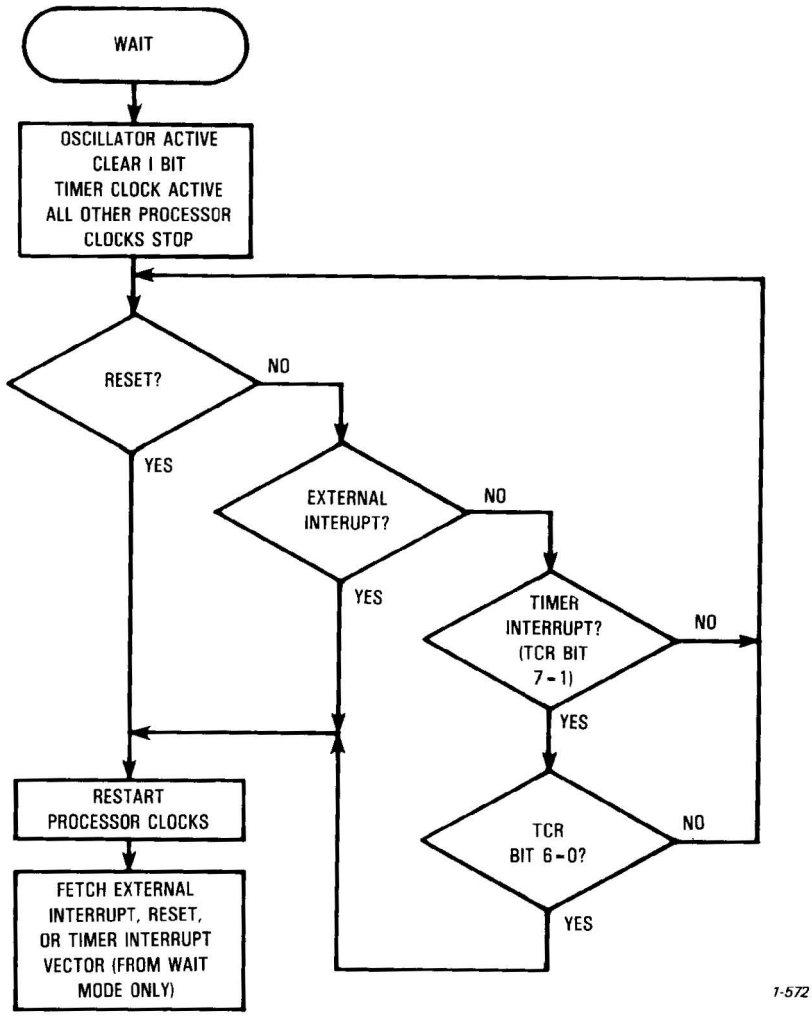


Figure 16. Wait Function Flowchart

The counter may be read at any time by the processor without disturbing the count. The contents of the counter become stable prior to the read portion of a cycle, and do not change during the read. The timer interrupt request bit (TCR7) remains set until cleared by the software. If the timer interrupt request bit (TCR7) is cleared before the timer interrupt is serviced, the interrupt is lost. The TCR7 bit may also be used as a scanned status bit in a non-interrupt mode of operation (TCR6 = 1).

The prescaler is a 7-bit divider which is used to extend the maximum length of the timer. Bit 0, bit 1, and bit 2 of the TCR are programmed to choose the appropriate prescaler output which is used as the counter input. The processor cannot write into or read from the prescaler; however, its contents are cleared to all zeros by the write operation into TCR when bit 3 of the written data equals a logic one. This allows for truncation-free counting.

The timer input can be configured for three different operating modes plus a disable mode, depending on the value written to the TCR4 and TCR5 timer control register bits. Refer to **TIMER CONTROL REGISTER**. Power-on reset and the STOP instruction affect the state of the counter.

TIMER INPUT MODE 1

If TCR4 and TCR5 are both programmed to a zero, the input to the timer is from an internal clock and the TIMER input pin is disabled. The internal clock mode can be used for periodic interrupt generation, as well as a reference in frequency and event measurement. The internal clock is the instruction cycle clock. During a WAIT instruction, the internal clock to the timer continues to run at its normal rate.

TIMER INPUT MODE 2

With TCR4 = 1 and TCR5 = 0, the internal clock and the TIMER input pin are ANDed to form the timer input signal. This mode can be used to measure external pulse widths. The external timer input pulse simply turns on the internal clock for the duration of the pulse. The resolution of the count in this mode is plus or minus one clock cycle; therefore, accuracy improves with longer input pulse widths.

TIMER INPUT MODE 3

If TCR4 = 0 and TCR5 = 1, then all inputs to the timer are disabled.

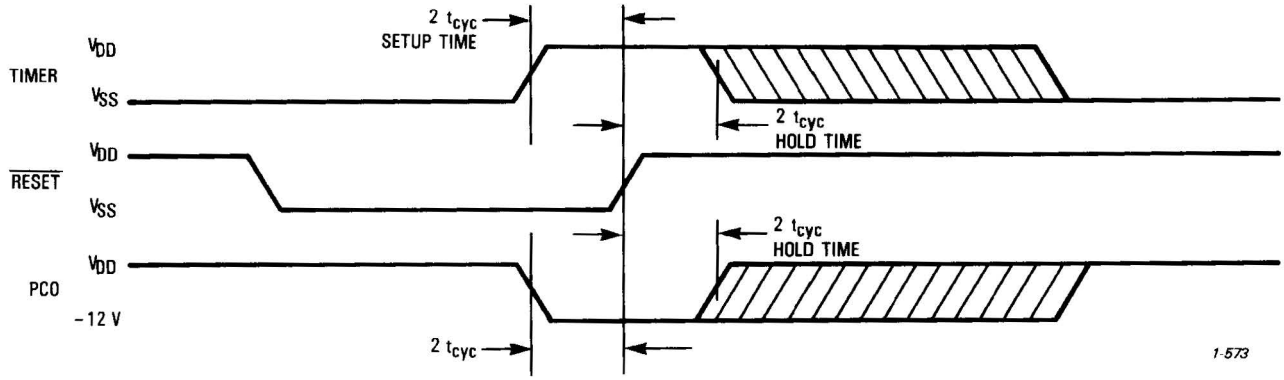
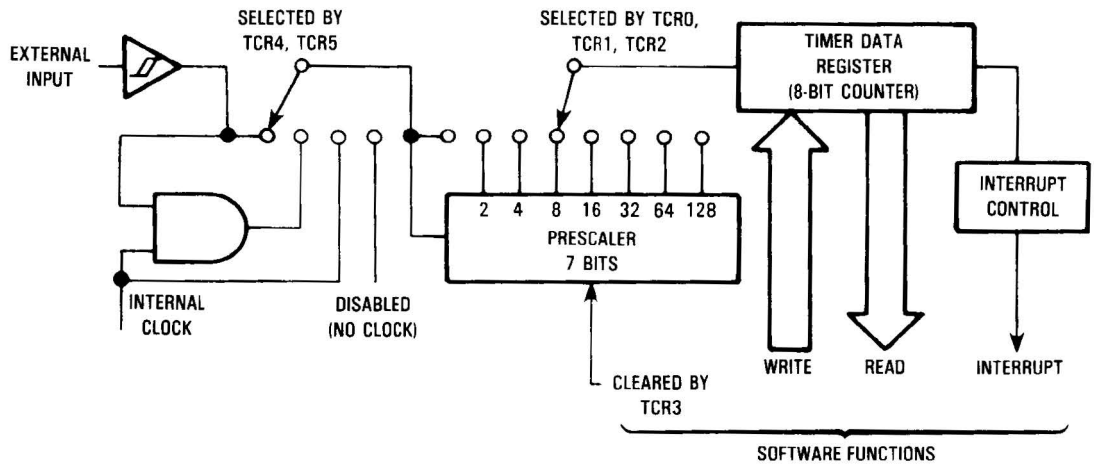


Figure 17. Bootstrap Mode



NOTES:
1. Prescaler and timer data register (8-bit counter) are clocked on the falling edge of the internal clock or external input.
2. The timer data register counts down continuously.

1-574

Figure 18. Timer Block Diagram

TIMER INPUT MODE 4

If TCR4 = 1 and TCR5 = 1, the internal clock input to the timer is disabled and the TIMER input pin becomes the input to the timer. The timer can, in this mode, be used to count external events as well as external frequencies for generating periodic interrupts. The counter is clocked on the falling edge of the external signal.

TIMER CONTROL REGISTER (TCR)

7	6	5	4	3	2	1	0	
TCR7	TCR6	TCR5	TCR4	TCR3	TCR2	TCR1	TCR0	$009

All bits in this register except bit 3 are read/write bits.

TCR7 — Timer interrupt request bit: bit used to indicate the timer interrupt when it is logic one.
1 — Set whenever the counter decrements to zero, or under program control.
0 — Cleared on external reset, power-on reset, STOP instruction, or program control.

TCR6 — Timer interrupt mask bit: when this bit is a logic one, it inhibits the timer interrupt to the processor.
1 — Set on external reset, power-on reset, STOP instruction, or program control.
0 — Cleared under program control.

TCR5 — External or internal bit: selects the input clock source to be either the external TIMER pin or the internal clock. This bit is unaffected by reset:
1 — Select external clock source.
0 — Select internal clock source (period = t_{cyc}).

TCR4 — External enable bit: control bit used to enable the external TIMER pin. This bit is unaffected by reset.
1 — Enable external TIMER pin.
0 — Disable external TIMER pin.

TCR5	TCR4	Result
0	0	Internal Clock to Timer
0	1	AND of Internal Clock and TIMER Pin to Timer
1	0	Inputs to Timer Disabled
1	1	TIMER Pin to Timer

TCR3 — Timer prescaler reset bit: writing a one to this bit resets the prescaler to zero. A read of this location always indicates a zero. This bit is unaffected by reset.

TCR2, TCR1, TCR0 — Prescaler select bits: decoded to select one of eight outputs of the prescaler. These bits are unaffected by reset.

Prescaler

TCR2	TCR1	TCR0	Result
0	0	0	÷1
0	0	1	÷2
0	1	0	÷4
0	1	1	÷8
1	0	0	÷16
1	0	1	÷32
1	1	0	÷64
1	1	1	÷128

MASK OPTIONS

The MC1468705F2 mask options are implemented as an EPROM byte at address $7F5 and are EPROM programmable.

MASK OPTION REGISTER (MOR)

7	6	5	4	3	2	1	0	
CLK*	DIV*	*	INT*					$7F5

*Reads of these locations always indicate a one.

A discussion of the function of each bit is as follows.

B7, CLK — Determines the clock oscillator type
0 — Crystal Oscillator
1 — RC Oscillator

B6, DIV — Determines division of clock oscillator
0 — Divide-by-four Oscillator Clock
1 — Divide-by-two Oscillator Clock

B4, INT — Determines type of interrupt trigger input
0 — Edge-Sensitive Triggered only
1 — Edge and Level-Sensitive

The EPROM in the erased state will read all zeros. While in bootstrap mode, the MC1468705F2 will operate under crystal oscillator, and divide-by-four options regardless of how the MOR register is programmed; however, the interrupt trigger input will remain as programmed in the MOR. When the MC1468705F2 is in the single-chip mode, the MCU operation is set up per the mask option register (MOR).

ERASING THE EPROM

The MC1468705F2 EPROM can be erased by exposure to high-intensity ultraviolet (UV) light with a wavelength of 2537 angstroms. The recommended integrated dose (UV intensity × exposure time) is 30 Ws/cm^2. The lamps should be used without shortwave filters and the MC1468705F2 should be positioned about one inch from the UV tubes. Ultraviolet erasure clears all bits of the MC1468705F2 EPROM to the zero state. Data is then entered by programming ones into the desired bit locations.

CAUTION

Be sure that the EPROM window is shielded from light at all times except when erasing. This protects both the EPROM and light-sensitive nodes.

PROGRAMMING FIRMWARE

A bootstrap program in ROM allows the MC1468705F2 to program the EPROM. The alternate vectoring used to implement the self-check in the MC146805F2 is used to start execution of this program.

When the V_{PP} voltage is placed on pin 3 (provided pin 26 has − 12 V applied through a 1 kilohm resistor and pin 27 has V_{DD} applied) and the bootstrap program is executed, the MC1468705F2 is able to program itself. This ability to program itself is a function of the external hardware and the interplay between the internal hardware and the firmware. The amount of time for programming is determined by a value stored in the counter by the firmware. When the part is placed in the bootstrap mode, the bootstrap vector is fetched and the bootstrap firmware starts to execute.

Note that the MCM2716 UV EPROM must be programmed first with a duplicate of the information that is to be transferred to the MC1468705F2 (see Figure 19). Non-EPROM addresses are ignored by the on-chip ROM bootstrap. Since the MC1468705F2 and the MCM2716 EPROM are to be inserted and removed from the circuit, they should be mounted in sockets. Additionally, the precautions below should be observed.

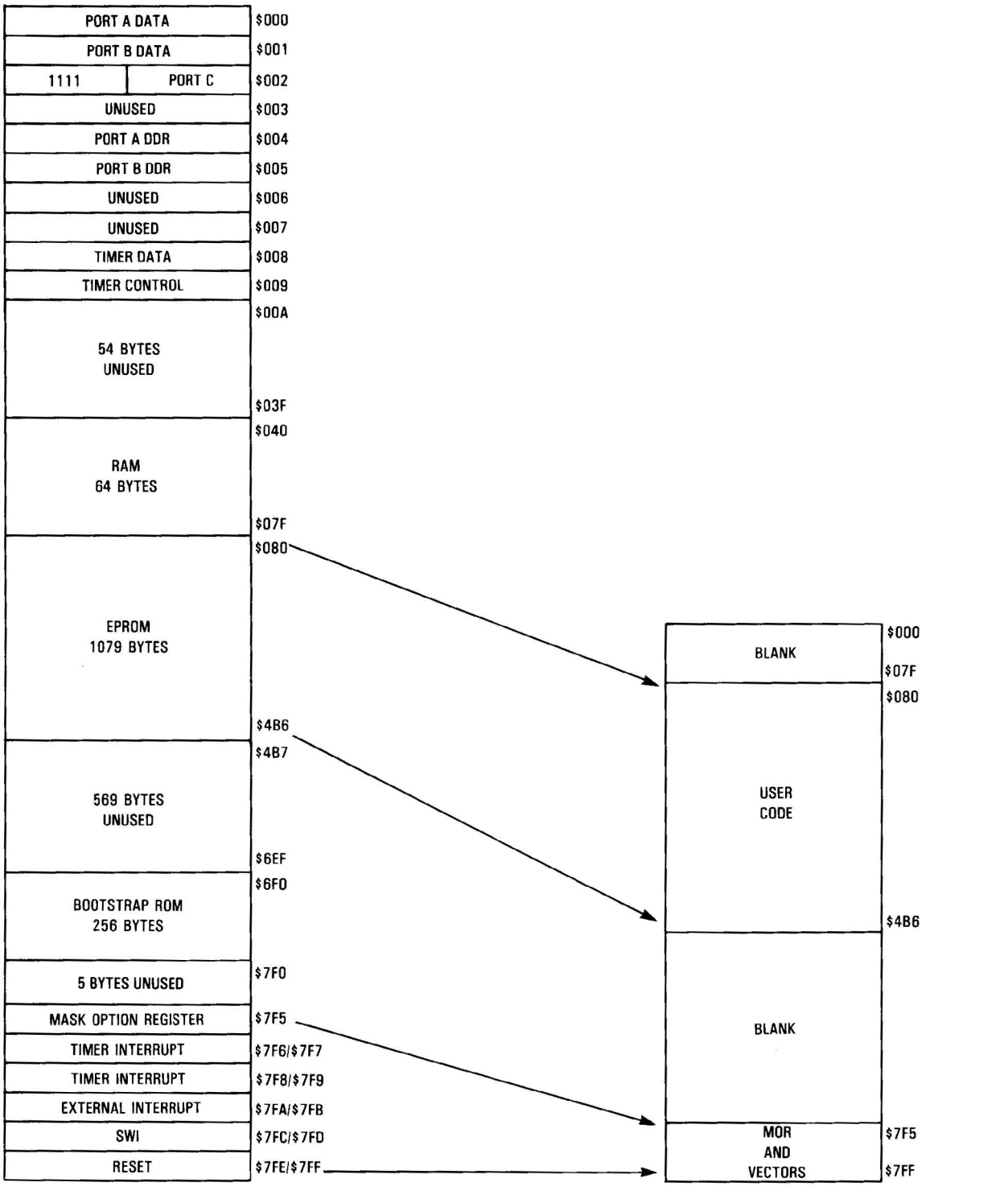
CAUTION

Be sure that S1 is open and S2 is closed (see Figure 20) when inserting the MC1468705F2 and 2716 EPROMs into their respective sockets. This ensures that $\overline{\text{RESET}}$ is held low while inserting the devices.

The MCM2716 memory locations which correspond to RAM locations or unused EPROM or ROM locations in the MC1468705F2 are programmed with $FF.

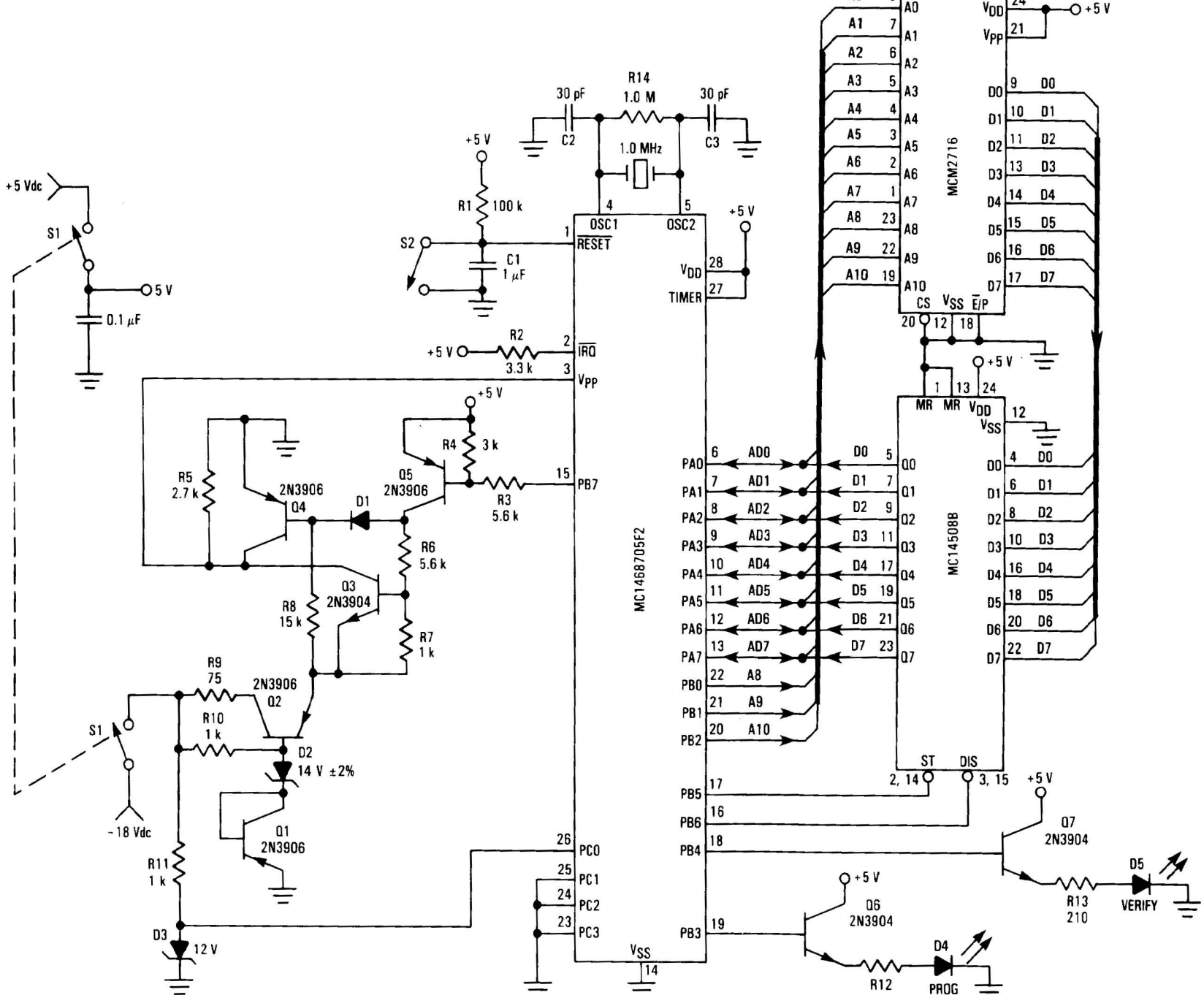
Refer to the schematic diagram of Figure 20. To program the MC168705F2, close S1 (to apply V_{DD} and to provide a positive voltage to the TIMER pin and a negative voltage to the PC0 pin). Next, open S2 to remove $\overline{\text{RESET}}$. When the reset cycle is complete, the internal ROM program initiates transfer of the external EPROM pattern one byte for each EPROM location.

The MC1468705F2 bootstrap provides the address signal to permit complete self-programming. Data is transferred from



1-575

Figure 19. MC1468705F2 Memory Mapping onto 2716



the 2716 location (which is equal to the A0-A10 address inputs) to the MC14508B latch. The strobe and enable outputs from the MC1468705F2 then allow the data to be transferred to the MC1468705F2 for loading into its equivalent EPROM location.

At the start of data transfer from the 2716 EPROM, the programming LED is turned on and remains on throughout the programming sequence. Transfer of the entire 2716 EPROM contents requires approximately 100 seconds. The internal counter is cleared and the loop is repeated to verify that the programmed data is precisely the same as the incoming data from the 2716 EPROM; if so, the programming LED turns off and the verified LED is turned on. Close S2 and open S1 prior to removing any device from its socket.

NOTE

Once the EPROM is programmed and connected for normal operation, be sure that V_{PP} (pin 3) is connected directly to V_{SS}.

INSTRUCTION SET

The MCU has a set of 61 basic instructions. They can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type. All the instructions within a given type are presented in individual tables.

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. The first operand is either the accumulator or the index register. The second operand is obtained from memory using one of the addressing modes. The operand for the jump unconditional (JMP) and jump to subroutine (JSR) instructions is the program counter. Refer to Table 4.

READ-MODIFY-WRITE INSTRUCTIONS

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read-modify-write sequence since it does not modify the value. Refer to Table 5.

BRANCH INSTRUCTIONS

Most branch instructions test the state of the condition code register and if certain criteria are met, a branch is executed. This adds an offset between −127 and +128 to the current program counter. Refer to Table 6.

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the first 128 bytes of the memory space (where all port registers, port DDRs, timer, timer control, and on-chip RAM reside). Bit manipulation in the EPROM mapped area will not affect data in the EPROM. An additional feature allows the software to test and branch on the state of any bit within the first 256 locations. The bit set, bit clear, and bit test and branch functions are all implemented with a single instruction. For the test and branch instructions, the value of the bit tested is automatically placed in the carry bit of the condition code register. Refer to Table 7.

CONTROL INSTRUCTIONS

These instructions are register reference instructions and are used to control processor operation during program execution. Refer to Table 8.

OPCODE MAP

Table 9 is an opcode map for the instructions used on the MCU.

ALPHABETICAL LISTING

The complete instruction set is given in alphabetical order in Table 10.

ADDRESSING MODES

The MCU uses ten different addressing modes to provide the programmer with an opportunity to optimize the code to all situations. The various indexed addressing modes make it possible to locate data tables, code conversion tables, and scaling tables anywhere in the memory space. Short indexed accesses are single byte instructions, while the longest instructions (three bytes) permit accessing tables throughout memory. Short absolute (direct) and long absolute (extended) addressing is also included. One and two byte direct addressing instructions access all data bytes in most applications. Extended addressing permits jump instructions to reach all memory. Table 10 shows the addressing modes for each instruction, with the effects each instruction has on the condition code register. An opcode map is shown in Table 9.

The term "effective address" (EA) is used in describing the various addressing modes, and is defined as the byte address to or from which the argument for an instruction is fetched or stored. The ten addressing modes of the processor are described below. Parentheses are used to indicate "contents of" the location or register referred to; e.g., (PC) indicates the contents of the location pointed to by the PC. An arrow indicates "is replaced by," and a colon indicates concatenation of two bytes. For additional details and graphical illustrations, refer to the *M6805 HMOS/M146805 CMOS Family Users' Manual* (M6805UM/AD2).

INHERENT

In inherent instructions, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator, and no other arguments, are included in this mode.

IMMEDIATE

In immediate addressing, the operand is contained in the byte immediately following the opcode. Immediate addressing is used to access constants which do not change during program execution (e.g., a constant used to initialize a loop counter).

$$EA = PC + 1;\ PC \leftarrow PC + 2$$

DIRECT

In the direct addressing mode, the effective address of the argument is contained in a single byte following the opcode byte. Direct addressing allows the user to directly address the lowest 256 bytes in memory with a single two byte instruction. This includes all on-chip RAM and I/O registers, and 128 bytes of on-chip ROM. Direct addressing is efficient in both memory and time.

$$EA = (PC + 1);\ PC \leftarrow PC + 2$$
$$\text{Address Bus High} \leftarrow 0;\ \text{Address Bus Low} \leftarrow (PC + 1)$$

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode. Instructions with extended addressing modes are capable of referencing arguments anywhere in memory with a single three-byte instruction. When using the Motorola assembler, the user need not specify whether an instruction uses direct or extended addressing. The assembler automatically selects the most efficient addressing mode.

$$EA = (PC + 1):(PC + 2);\ PC \leftarrow PC + 3$$
$$\text{Address Bus High} \leftarrow (PC + 1);\ \text{Address Bus Low} \leftarrow (PC + 2)$$

INDEXED, NO OFFSET

In the indexed, no offset addressing mode, the effective address of the argument is contained in the 8-bit index register. Thus, this addressing mode can access the first 256 memory locations. These instructions are only one byte long. This mode is used to move a pointer through a table or to address a frequently referenced RAM or I/O location.

$$EA = X;\ PC \leftarrow PC + 1$$
$$\text{Address Bus High} \leftarrow 0;\ \text{Address Bus Low} \leftarrow X$$

INDEXED, 8-BIT OFFSET

Here the EA is obtained by adding the contents of the byte following the opcode to that of the index register; therefore, the operand is located anywhere within the lowest 511 memory locations. For example, this mode of addressing is useful for selecting the mth element in an n element table. All instructions are two bytes. The contents of the index register (X) is not changed. The contents of (PC + 1) is an unsigned 8-bit integer. One byte offset indexing permits look-up tables to be easily accessed in either RAM or ROM.

$$EA = X + (PC + 1);\ PC \leftarrow PC + 2$$
$$\text{Address Bus High} \leftarrow K;\ \text{Address Bus Low} \leftarrow X + (PC + 1)$$

where:

K = The carry from the addition of X + (PC + 1)

INDEXED, 16-BIT OFFSET

In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This addressing mode can be used in a manner similar to indexed 8-bit offset, except that this three byte instruction allows tables to be anywhere in memory (e.g., jump tables in ROM). As with direct and extended, the M6805 assembler determines the most efficient form of indexed offset; 8- or 16-bit. The contents of the index register is not changed.

$$EA = X + [(PC + 1):(PC + 2)];\ PC \leftarrow PC + 3$$
$$\text{Address Bus High} \leftarrow (PC + 1) + K;$$
$$\text{Address Bus Low} \leftarrow X + (PC + 2)$$

where:

K = The carry from the addition of X + (PC + 2)

RELATIVE

Relative addressing is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte following the opcode (the offset) is added to the PC if and only if the branch condition is true. Otherwise, control proceeds to the next instruction. The span of relative addressing is limited to the range of −126 to +129 bytes from the branch instruction opcode location. The Motorola assembler calculates the proper offset and checks to see if it is within the span of the branch.

$$EA = PC + 2 + (PC + 1);\ PC \leftarrow EA \text{ if branch taken;}$$
$$\text{otherwise, } EA = PC \leftarrow PC + 2$$

BIT SET/CLEAR

Direct addressing and bit addressing are combined in instructions which set and clear individual memory and I/O bits. In the bit set and clear instructions, the byte is specified as a direct address in the location following the opcode. The first 256 addressable locations are thus accessed. The bit to be modified within that byte is specified with the first three bits of the opcode. The bit set and clear instructions occupy two bytes, one for the opcode (including the bit number) and the other to address the byte which contains the bit of interest.

$$EA = (PC + 1);\ PC \leftarrow PC + 2$$
$$\text{Address Bus High} \leftarrow 0;\ \text{Address Bus Low} \leftarrow (PC + 1)$$

BIT TEST AND BRANCH

Bit test and branch is a combination of direct addressing, bit set/clear addressing, and relative addressing. The actual bit to be tested, within the byte, is specified within the low order nibble of the opcode. The address of the data byte to be tested is located via a direct address in the location following the opcode byte (EA1). The signed relative 8-bit offset is in the third byte (EA2) and is added to the PC if the specified bit is set or cleared in the specified memory location. This single three byte instruction allows the program to branch based on the condition of any bit in the first 256 locations of memory.

$$EA1 = (PC + 1)$$
$$\text{Address Bus High} \leftarrow 0;\ \text{Address Bus Low} \leftarrow (PC + 1)$$
$$EA2 = PC + 3 + (PC + 2);\ PC \leftarrow EA2 \text{ if branch taken;}$$
$$\text{otherwise, } PC \leftarrow PC + 3$$

Table 4. Register/Memory Instructions

		Addressing Modes																	
		Immediate			Direct			Extended			Indexed (No Offset)			Indexed (8-Bit Offset)			Indexed (16-Bit Offset)		
Function	Mnemonic	Op-code	# Bytes	# Cycles	Op-code	# Bytes	# Cycles	Op-code	# Bytes	# Cycles	Op-code	# Bytes	# Cycles	Op-code	# Bytes	# Cycles	Op-code	# Bytes	# Cycles
Load A from Memory	LDA	A6	2	2	B6	2	3	C6	3	4	F6	1	3	E6	2	4	D6	3	5
Load X from Memory	LDX	AE	2	2	BE	2	3	CE	3	4	FE	1	3	EE	2	4	DE	3	5
Store A in Memory	STA	—	—	—	B7	2	4	C7	3	5	F7	1	4	E7	2	5	D7	3	6
Store X in Memory	STX	—	—	—	BF	2	4	CF	3	5	FF	1	4	EF	2	5	DF	3	6
Add Memory to A	ADD	AB	2	2	BB	2	3	CB	3	4	FB	1	3	EB	2	4	DB	3	5
Add Memory and Carry to A	ADC	A9	2	2	B9	2	3	C9	3	4	F9	1	3	E9	2	4	D9	3	5
Subtract Memory	SUB	A0	2	2	B0	2	3	C0	3	4	F0	1	3	E0	2	4	D0	3	5
Subtract Memory from A with Borrow	SBC	A2	2	2	B2	2	3	C2	3	4	F2	1	3	E2	2	4	D2	3	5
AND Memory to A	AND	A4	2	2	B4	2	3	C4	3	4	F4	1	3	E4	2	4	D4	3	5
OR Memory with A	ORA	AA	2	2	BA	2	3	CA	3	4	FA	1	3	EA	2	4	DA	3	5
Exclusive OR Memory with A	EOR	A8	2	2	B8	2	3	C8	3	4	F8	1	3	E8	2	4	D8	3	5
Arithmetic Compare A with Memory	CMP	A1	2	2	B1	2	3	C1	3	4	F1	1	3	E1	2	4	D1	3	5
Arithmetic Compare X with Memory	CPX	A3	2	2	B3	2	3	C3	3	4	F3	1	3	E3	2	4	D3	3	5
Bit Test Memory with A (Logical Compare)	BIT	A5	2	2	B5	2	3	C5	3	4	F5	1	3	E5	2	4	D5	3	5
Jump Unconditional	JMP	—	—	—	BC	2	2	CC	3	3	FC	1	2	EC	2	3	DC	3	4
Jump to Subroutine	JSR	—	—	—	BD	2	5	CD	3	6	FD	1	5	ED	2	6	DD	3	7

Table 5. Read-Modify-Write Instructions

		Addressing Modes														
		Inherent (A)			Inherent (X)			Direct			Indexed (No Offset)			Indexed (8-Bit Offset)		
Function	Mnemonic	Op-code	# Bytes	# Cycles	Op-code	# Bytes	# Cycles	Op-code	# Bytes	# Cycles	Op-code	# Bytes	# Cycles	Op-code	Op-code	# Cycles
Increment	INC	4C	1	3	5C	1	3	3C	2	5	7C	1	5	6C	2	6
Decrement	DEC	4A	1	3	5A	1	3	3A	2	5	7A	1	5	6A	2	6
Clear	CLR	4F	1	3	5F	1	3	3F	2	5	7F	1	5	6F	2	6
Complement	COM	43	1	3	53	1	3	33	2	5	73	1	5	63	2	6
Negate (2's Complement)	NEG	40	1	3	50	1	3	30	2	5	70	1	5	60	2	6
Rotate Left Thru Carry	ROL	49	1	3	59	1	3	39	2	5	79	1	5	69	2	6
Rotate Right Thru Carry	ROR	46	1	3	56	1	3	36	2	5	76	1	5	66	2	6
Logical Shift Left	LSL	48	1	3	58	1	3	38	2	5	78	1	5	68	2	6
Logical Shift Right	LSR	44	1	3	54	1	3	34	2	5	74	1	5	64	2	6
Arithmetic Shift Right	ASR	47	1	3	57	1	3	37	2	5	77	1	5	67	2	6
Test for Negative or Zero	TST	4D	1	3	5D	1	3	3D	2	4	7D	1	4	6D	2	5

Table 6. Branch Instructions

		Relative Addressing Mode		
Function	Mnemonic	Opcode	# Bytes	# Cycles
Branch Always	BRA	20	2	3
Branch Never	BRN	21	2	3
Branch IFF Higher	BHI	22	2	3
Branch IFF Lower or Same	BLS	23	2	3
Branch IFF Carry Clear	BCC	24	2	3
(Branch IFF Higher or Same)	(BHS)	24	2	3
Branch IFF Carry Set	BCS	25	2	3
(Branch IFF Lower)	(BLO)	25	2	3
Branch IFF Not Equal	BNE	26	2	3
Branch IFF Equal	BEQ	27	2	3
Branch IFF Half Carry Clear	BHCC	28	2	3
Branch IFF Half Carry Set	BHCS	29	2	3
Branch IFF Plus	BPL	2A	2	3
Branch IFF Minus	BMI	2B	2	3
Branch IFF Interrupt Mask Bit is Clear	BMC	2C	2	3
Branch IFF Interrupt Mask Bit is Set	BMS	2D	2	3
Branch IFF Interrupt Line is Low	BIL	2E	2	3
Branch IFF Interrupt Line is High	BIH	2F	2	3
Branch to Subroutine	BSR	AD	2	6

1-579

Table 7. Bit Manipulation Instructions

		Addressing Modes					
		Bit Set/Clear			Bit Test and Branch		
Function	Mnemonic	Opcode	# Bytes	# Cycles	Opcode	# Bytes	# Cycles
Branch IFF Bit n is Set	BRSET n (n=0...7)	—	—	—	2•n	3	5
Branch IFF Bit n is Clear	BRCLR n (n=0...7)	—	—	—	01+2•n	3	5
Set Bit n	BSET n (n=0...7)	10+2•n	2	5	—	—	—
Clear Bit n	BCLR n (n=0...7)	11+2•n	2	5	—	—	—

1-580

Table 8. Control Instructions

		Inherent		
Function	**Mnemonic**	**Opcode**	**# Bytes**	**# Cycles**
Transfer A to X	TAX	97	1	2
Transfer X to A	TXA	9F	1	2
Set Carry Bit	SEC	99	1	2
Clear Carry Bit	CLC	98	1	2
Set Interrupt Mask Bit	SEI	9B	1	2
Clear Interrupt Mask Bit	CLI	9A	1	2
Software Interrupt	SWI	83	1	10
Return from Subroutine	RTS	81	1	6
Return from Interrupt	RTI	80	1	9
Reset Stack Pointer	RSP	9C	1	2
No-Operation	NOP	9D	1	2
Stop	STOP	8E	1	2
Wait	WAIT	8F	1	2

1-581

Table 9. Instruction Set Opcode Map

	Bit Manipulation		Branch	Read-Modify-Write					Control		Register/Memory						
	BTB	BSC	REL	DIR	INH	INH	IX1	IX	INH	INH	IMM	DIR	EXT	IX2	IX1	IX	
Hi / Low	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111	8 1000	9 1001	A 1010	B 1011	C 1100	D 1101	E 1110	F 1111	Hi / Low
0 0000	BRSET0 5 3 BTB	BSET0 5 2 BSC	BRA 3 2 REL	NEG 5 2 DIR	NEG 3 1 INH	NEG 3 1 INH	NEG 6 2 IX1	NEG 5 1 IX	RTI 9 1 INH		SUB 2 2 IMM	SUB 3 2 DIR	SUB 4 3 EXT	SUB 5 3 IX2	SUB 4 2 IX1	SUB 3 1 IX	0 0000
1 0001	BRCLR0 5 3 BTB	BCLR0 5 2 BSC	BRN 3 2 REL						RTS 6 1 INH		CMP 2 2 IMM	CMP 3 2 DIR	CMP 4 3 EXT	CMP 5 3 IX2	CMP 4 2 IX1	CMP 3 1 IX	1 0001
2 0010	BRSET1 5 3 BTB	BSET1 5 2 BSC	BHI 3 2 REL								SBC 2 2 IMM	SBC 3 2 DIR	SBC 4 3 EXT	SBC 5 3 IX2	SBC 4 2 IX1	SBC 3 1 IX	2 0010
3 0011	BRCLR1 5 3 BTB	BCLR1 5 2 BSC	BLS 3 2 REL	COM 5 2 DIR	COMA 3 1 INH	COMX 3 1 INH	COM 6 2 IX1	COM 5 1 IX	SWI 10 1 INH		CPX 2 2 IMM	CPX 3 2 DIR	CPX 4 3 EXT	CPX 5 3 IX2	CPX 4 2 IX1	CPX 3 1 IX	3 0011
4 0100	BRSET2 5 3 BTB	BSET2 5 2 BSC	BCC 3 2 REL	LSR 5 2 DTR	LSRA 3 1 INH	LSRX 3 1 INH	LSR 6 2 IX1	LSR 5 1 IX			AND 2 2 IMM	AND 3 2 DIR	AND 4 3 EXT	AND 5 3 IX2	AND 4 2 IX1	AND 3 1 IX	4 0100
5 0101	BRCLR2 5 3 BTB	BCLR2 5 2 BSC	BCS 3 2 REL								BIT 2 2 IMM	BIT 3 2 DIR	BIT 4 3 EXT	BIT 5 3 IX2	BIT 4 2 IX1	BIT 3 1 IX	5 0101
6 0110	BRSET3 5 3 BTB	BSET3 5 2 BSC	BNE 3 2 REL	ROR 5 2 DIR	RORA 3 1 INH	RORX 3 1 INH	ROR 6 2 IX1	ROR 5 1 IX			LDA 2 2 IMM	LDA 3 2 DIR	LDA 4 3 EXT	LDA 5 3 IX2	LDA 4 2 IX1	LDA 3 1 IX	6 0110
7 0111	BRCLR3 5 3 BTB	BCLR3 5 2 BSC	BEQ 3 2 REL	ASR 5 2 DIR	ASRA 3 1 INH	ASRX 3 1 INH	ASR 6 2 IX1	ASR 5 1 IX		TAX 2 1 INH		STA 4 2 DIR	STA 5 3 EXT	STA 6 3 IX2	STA 5 2 IX1	STA 4 1 IX	7 0111
8 1000	BRSET4 5 3 BTB	BSET4 5 2 BSC	BHCC 3 2 REL	LSL 5 2 DIR	LSLA 3 1 INH	LSLX 3 1 INH	LSL 6 2 IX1	LSL 5 1 IX		CLC 2 1 INH	EOR 2 2 IMM	EOR 3 2 DIR	EOR 4 3 EXT	EOR 5 3 IX2	EOR 4 2 IX1	EOR 3 1 IX	8 1000
9 1001	BRCLR4 5 3 BTB	BCLR4 5 2 BSC	BHCS 3 2 REL	ROL 5 2 DIR	ROLA 3 1 INH	ROLX 3 1 INH	ROL 6 2 IX1	ROL 5 1 IX		SEC 2 1 INH	ADC 2 2 IMM	ADC 3 2 DIR	ADC 4 3 EXT	ADC 5 3 IX2	ADC 4 2 IX1	ADC 3 1 IX	9 1001
A 1010	BRSET5 5 3 BTB	BSET5 5 2 BSC	BPL 3 2 REL	DEC 5 2 DIR	DECA 3 1 INH	DECX 3 1 INH	DEC 6 2 IX1	DEC 5 1 IX		CLI 2 1 INH	ORA 2 2 IMM	ORA 3 2 DIR	ORA 4 3 EXT	ORA 5 3 IX2	ORA 4 2 IX1	ORA 3 1 IX	A 1010
B 1011	BRCLR5 5 3 BTB	BCLR5 5 2 BSC	BMI 3 2 REL							SEI 2 1 INH	ADD 2 2 IMM	ADD 3 2 DIR	ADD 4 3 EXT	ADD 5 3 IX2	ADD 4 2 IX1	ADD 3 1 IX	B 1011
C 1100	BRSET6 5 3 BTB	BSET6 5 2 BSC	BMC 3 2 REL	INC 5 2 DIR	INCA 3 1 INH	INCX 3 1 INH	INC 6 2 IX1	INC 5 1 IX		RSP 2 1 INH		JMP 2 2 DIR	JMP 3 3 EXT	JMP 4 3 IX2	JMP 3 2 IX1	JMP 2 1 IX	C 1100
D 1101	BRCLR6 5 3 BTB	BCLR6 5 2 BSC	BMS 3 2 REL	TST 4 2 DIR	TSTA 3 1 INH	TSTX 3 1 INH	TST 5 2 IX1	TST 4 1 IX		NOP 2 1 INH	BSR 6 2 REL	JSR 5 2 DIR	JSR 6 3 EXT	JSR 7 3 IX2	JSR 6 2 IX1	JSR 5 1 IX	D 1101
E 1110	BRSET7 5 3 BTB	BSET7 5 2 BSC	BIL 3 2 REL						STOP 2 1 INH		LDX 2 2 IMM	LDX 3 2 DIR	LDX 4 3 EXT	LDX 5 3 IX2	LDX 4 2 IX1	LDX 3 1 IX	E 1110
F 1111	BRCLR7 5 3 BTB	BCLR7 5 2 BSC	BIH 3 2 REL	CLR 5 2 DIR	CLRA 3 1 INH	CLRX 3 1 INH	CLR 6 2 IX1	CLR 5 1 IX	WAIT 2 1 INH	TXA 2 1 INH		STX 4 2 DIR	STX 5 3 EXT	STX 6 3 IX2	STX 5 2 IX1	STX 4 1 IX	F 1111

Abbreviations for Address Modes

INH	Inherent
IMM	Immediate
DIR	Direct
EXT	Extended
REL	Relative
BSC	Bit Set/Clear
BTB	Bit Test and Branch

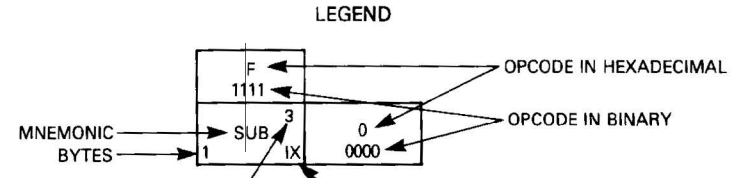


Table 10. Instruction Set

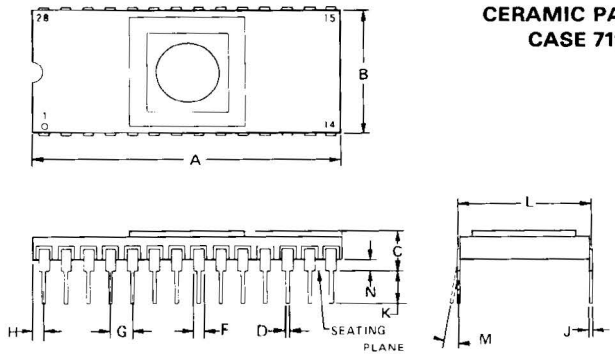
	Addressing Modes										Condition Codes				
Mnemonic	Inherent	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/ Clear	Bit Test & Branch	H	I	N	Z	C
ADC		X	X	X		X	X	X			Λ	•	Λ	Λ	Λ
ADD		X	X	X		X	X	X			Λ	•	Λ	Λ	Λ
AND		X	X	X		X	X	X			•	•	Λ	Λ	•
ASL	X		X			X	X				•	•	Λ	Λ	Λ
ASR	X		X			X	X				•	•	Λ	Λ	Λ
BCC					X						•	•	•	•	•
BCLR									X		•	•	•	•	•
BCS					X						•	•	•	•	•
BEQ					X						•	•	•	•	•
BHCC					X						•	•	•	•	•
BHCS					X						•	•	•	•	•
BHI					X						•	•	•	•	•
BHS					X						•	•	•	•	•
BIH					X						•	•	•	•	•
BIL					X						•	•	•	•	•
BIT		X	X	X		X	X	X			•	•	Λ	Λ	•
BLO					X						•	•	•	•	•
BLS					X						•	•	•	•	•
BMC					X						•	•	•	•	•
BMI					X						•	•	•	•	•
BMS					X						•	•	•	•	•
BNE					X						•	•	•	•	•
BPL					X						•	•	•	•	•
BRA					X						•	•	•	•	•
BRN					X						•	•	•	•	•
BRCLR										X	•	•	•	•	Λ
BRSET										X	•	•	•	•	Λ
BSET									X		•	•	•	•	•
BSR					X						•	•	•	•	•
CLC	X										•	•	•	•	0
CLI	X										•	0	•	•	•
CLR	X		X			X	X				•	•	0	1	•
CMP		X	X	X		X	X	X			•	•	Λ	Λ	Λ
COM	X		X			X	X				•	•	Λ	Λ	1
CPX		X	X	X		X	X	X			•	•	Λ	Λ	Λ
DEC	X		X			X	X				•	•	Λ	Λ	•
EOR		X	X	X		X	X	X			•	•	Λ	Λ	•
INC	X		X			X	X				•	•	Λ	Λ	•
JMP			X	X		X	X	X			•	•	•	•	•
JSR			X	X		X	X	X			•	•	•	•	•
LDA		X	X	X		X	X	X			•	•	Λ	Λ	•
LDX		X	X	X		X	X	X			•	•	Λ	Λ	•
LSL	X		X			X	X				•	•	Λ	Λ	Λ
LSR	X		X			X	X				•	•	0	Λ	Λ
NEG	X		X			X	X				•	•	Λ	Λ	Λ
NOP	X										•	•	•	•	•
ORA		X	X	X		X	X	X			•	•	Λ	Λ	•
ROL	X		X			X	X				•	•	Λ	Λ	Λ
ROR	X		X			X	X				•	•	Λ	Λ	Λ
RSP	X										•	•	•	•	•
RTI	X										?	?	?	?	?
RTS	X										•	•	•	•	•
SBC		X	X	X		X	X	X			•	•	Λ	Λ	Λ
SEC	X										•	•	•	•	1
SEI	X										•	1	•	•	•
STA			X	X		X	X	X			•	•	Λ	Λ	•
STOP	X										•	0	•	•	•
STX			X	X		X	X	X			•	•	Λ	Λ	•
SUB		X	X	X		X	X	X			•	•	Λ	Λ	Λ
SWI	X										•	1	•	•	•
TAX	X										•	•	•	•	•
TST	X		X			X	X				•	•	Λ	Λ	•
TXA	X										•	•	•	•	•
WAIT	X										•	0	•	•	•

Condition Code Symbols

- H Half Carry (From Bit 3)
- I Interrupt Mask
- N Negative (Sign Bit)
- Z Zero
- C Carry/Borrow
- Λ Test and Set if True, Cleared Otherwise
- • Not Affected
- ? Load CC Register From Stack
- 0 Cleared
- 1 Set

1-583

PACKAGE DIMENSIONS

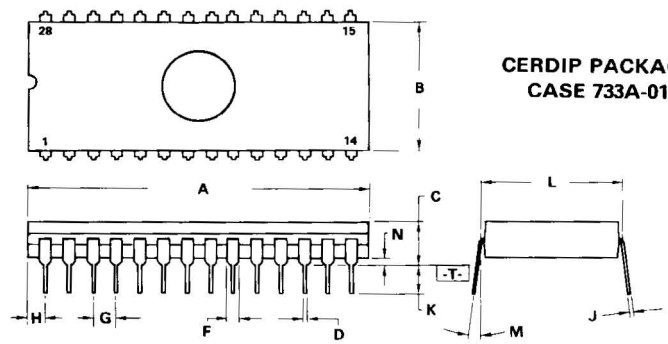


CERAMIC PACKAGE
CASE 719-04

NOTES:
1. LEADS, TRUE POSITIONED WITHIN 0.25 mm (0.010) DIAMETER (AT SEATING PLANE) AT MAXIMUM MATERIAL CONDITION.
2. DIMENSION "L" TO CENTER OF LEADS WHEN FORMED PARALLEL.

#22

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	35.20	35.92	1.386	1.414
B	14.73	15.34	0.580	0.604
C	3.18	5.08	0.125	0.200
D	0.38	0.53	0.015	0.021
F	0.76	1.40	0.030	0.055
G	2.54 BSC		0.100 BSC	
H	0.76	1.78	0.030	0.070
J	0.20	0.30	0.008	0.012
K	2.54	4.57	0.100	0.180
L	14.99	15.49	0.590	0.610
M	–	10°	–	10°
N	0.51	1.52	0.020	0.060



CERDIP PACKAGE
CASE 733A-01

NOTES:
1. DIMENSION "A" IS A DATUM. T IS BOTH A DATUM AND A SEATING PLANE.
2. POSITIONAL TOLERANCE FOR LEADS: (28 PLACES) ⌖ | ϕ 0.25 (0.010) Ⓜ | T | A Ⓜ
3. DIMENSIONS "A" & B INCLUDE MENISCUS.
4. DIMENSION L TO CENTER OF LEADS WHEN FORMED PARALLEL.
5. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
6. CONTROLLING DIMENSION: INCH.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	36.45	37.84	1.435	1.490
B	12.70	15.36	0.500	0.605
C	4.06	6.09	0.160	0.240
D	0.38	0.55	0.015	0.022
F	1.27	1.65	0.050	0.065
G	2.54 BSC		0.100 BSC	
J	0.20	0.30	0.008	0.012
K	3.17	4.06	0.125	0.160
L	15.24 BSC		0.600 BSC	
M	0°	15°	0°	15°
N	0.51	1.27	0.020	0.050

ORDERING INFORMATION

Package Type	Temperature	Order Number
Ceramic	0° to 70°C	MC1468705F2L
L Suffix	−40° to +85°C	MC1468705F2CL
Cerdip	0° to 70°C	MC1468705F2S
S Suffix	−40° to +85°C	MC1468705F2CS

Literature Distribution Centers:

USA: Motorola Literature Distribution; P.O. Box 20912; Phoenix, Arizona 85036.

EUROPE: Motorola Ltd.; European Literature Center; Fairfax House; 69 Buckingham Ct.; Aylesbury Bucks; HP202NF United Kingdom.

HONG KONG: Motorola Inc.; International Semiconductor Group; P.O. Box 80300; Cheung Sha Wan Post Office; Hong Kong.

MOTOROLA

A18621-1 PRINTED IN USA 11-85 IMPERIAL LITHO C35251 18,000

ADI104981