MC6804J1

Technical Summary

8-Bit Microcomputer Unit

MC6804J1 HMOS (high-density NMOS) microcomputer unit (MCU) is a member of the M6804 Family of serial processing microcomputers. This device displays all the versatility of an MCU whose design-ability to process 8-bit variables one bit at a time already makes it tremendously cost effective.

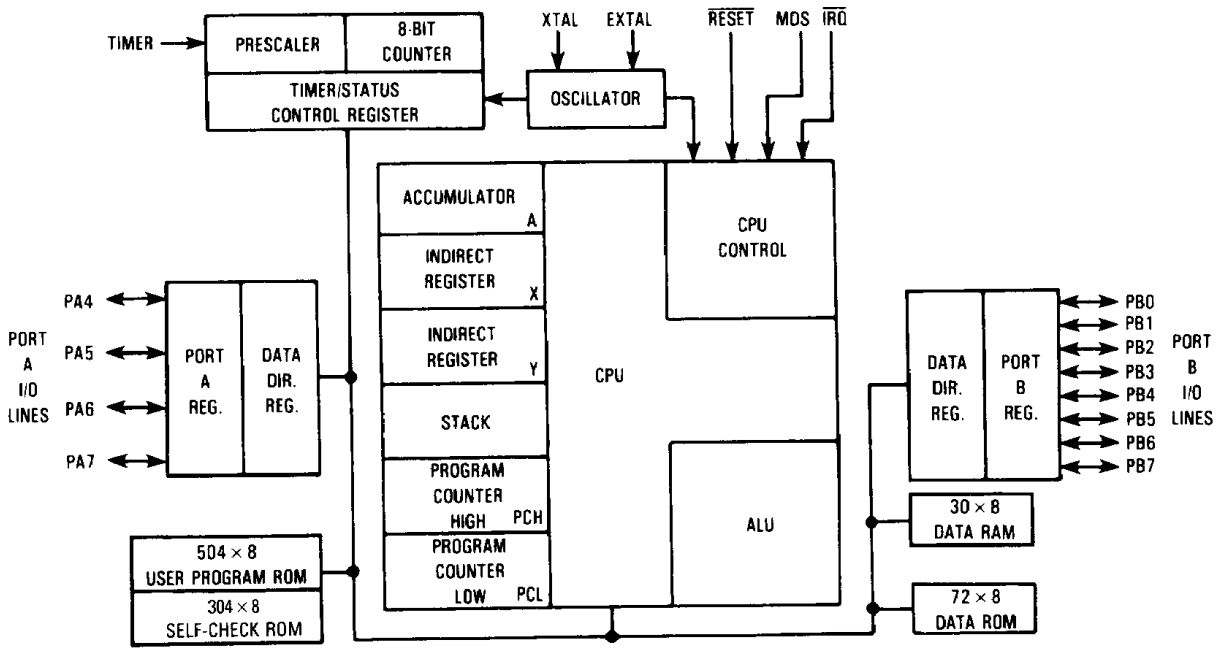
This technical summary contains limited information on the MC6804J1. For detailed information, refer to the advanced information data sheet for the MC6804J1, MC6804J2, MC6804P2, and MC68704P2 8-bit microcomputers (MC6804J1/D) or to the *M6804 MCU Manual* (DLE404/D).

Major hardware and software features of the MC6804P2 MCU are:

- On-Chip Clock Generator
- Memory Mapped I/O
- Software Programmable 8-Bit Timer with 7-Bit Prescaler
- Single Instruction Memory Examine/ Change
- 30 Bytes of Data RAM
- True Bit Manipulation
- Bit Test and Branch Instruction
- 304 Bytes Self-Check ROM
- Conditional Branches
- Timer Pin is Software Programmable as Clock Input or Timer Output
- 504 Bytes of User Program Space ROM
- User Selectable Constant Current Pullup Devices available on LSTTL and Open-Drain Interface Ports
- Mask Selectable Edge- or Level-Sensitive Interrupt Pin

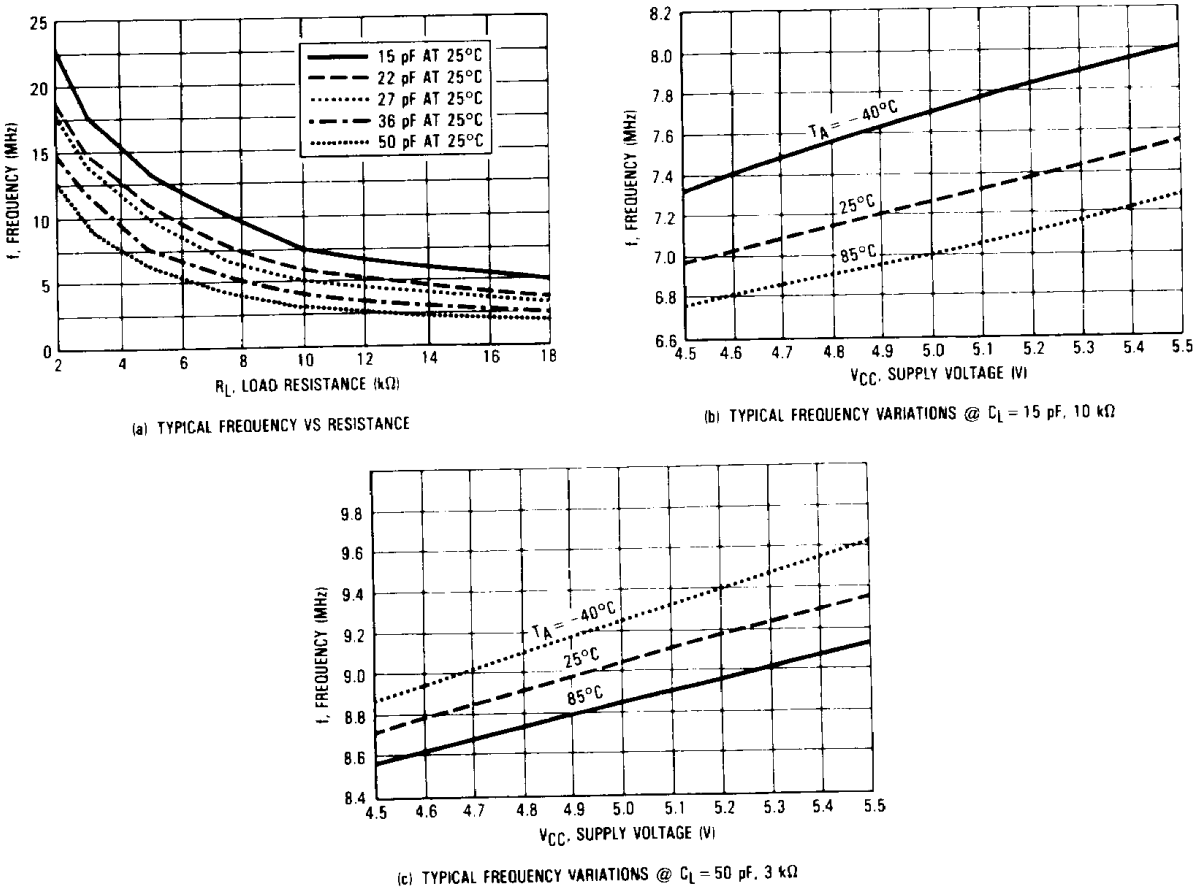
3

BLOCK DIAGRAM



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

IMAGE UNAVAILABLE



(a) TYPICAL FREQUENCY VS RESISTANCE

(b) TYPICAL FREQUENCY VARIATIONS @ C_L = 15 pF, 10 kΩ

(c) TYPICAL FREQUENCY VARIATIONS @ C_L = 50 pF, 3 kΩ

Figure 2. Typical Frequency Selection for Resistor/Capacitor Oscillator Options

inputs or outputs under software control of the data direction registers.

PROGRAMMING

INPUT/OUTPUT PROGRAMMING

There are 12 input/output pins. All pins of each port are programmable as inputs or outputs under the control of the data direction registers (DDR).

The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic one for output, or a logic zero for input, as shown in Figure 3. When the registers are programmed as outputs, the latched data is readable regardless of the logic levels at the output pin due to output loading.

All the I/O pins are LSTTL compatible as both inputs and outputs. In addition, both ports may use either or both of two manufacturing mask options; open drain output, or internal pull-up resistor for CMOS compatibility.

Any write to a port writes to all of its data bits even though the port DDR may be set to input. This can be used as a tool to initialize the data registers and avoid undefined outputs. However, care must be exercised when using read-modify-write instructions. The data read corresponds to the pin level if the DDR is an input or to the latched output data when the DDR is an output.

The 12 bidirectional lines may be configured by port to be the standard configuration (LSTTL), or either mask option; LSTTL/CMOS, or open drain. Port B outputs are LED compatible.

Port Data Registers ($00, $01)

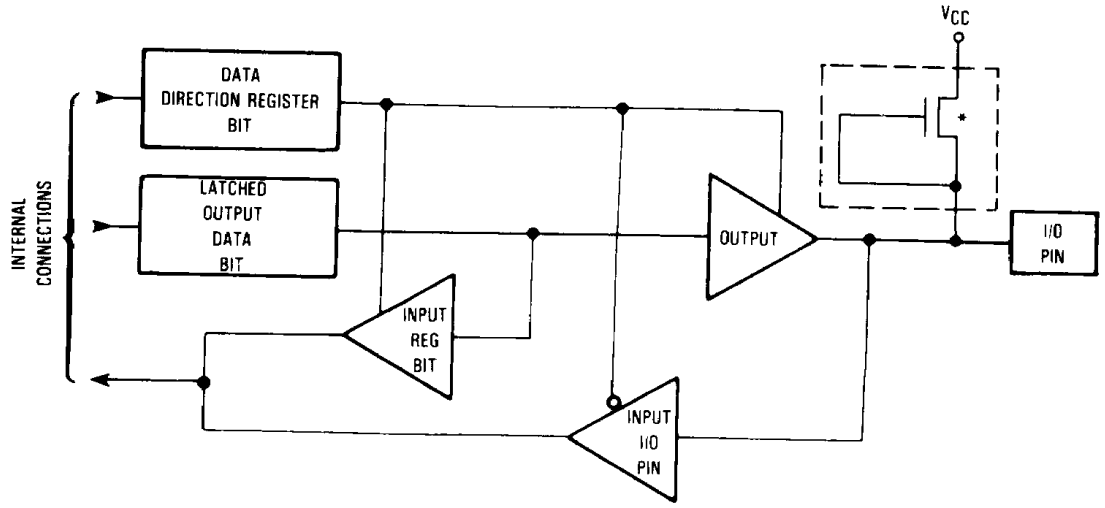
The port data registers are not initialized on reset. These registers should be initialized before changing the DDR bits to avoid undefined levels.

Port A ($00)

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

Port B ($01)

7	6	5	4	3	2	1	0



DATA DIRECTION REGISTER BIT	OUTPUT DATA BIT	OUTPUT STATE	INPUT TO MCU
1	0	0	0
1	1	1	1
0	X	HI-Z	PIN

*For CMOS option transistor acts as resistor (approximately 40 kΩ) to V_{CC}.
For LSTTL/open-drain options transistor acts as low current clamping diode to V_{CC}.

Figure 3. Typical I/O Port Circuitry

With regard to Port A only, the four LSB bits are unused. These bits are "don't care" (X) bits when written to but are always logic high when read.

Port Data Direction Registers ($04, $05)

Port DDRs configure the port pins as either outputs or inputs. Each port pin can be programmed individually to be an input or an output. A zero in the pin's corresponding DDR bit programs it as an input; a logic one programs it as an output. On reset, all the DDRs are initialized to a logic zero state to put the ports in the input mode.

Port A ($04)

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

Port B ($05)

7	6	5	4	3	2	1	0

With regard to Port A, the four LSB bits are cleared (logic zero) by reset. These bits must not be set (logic one).

MEMORY

The MCU memory map (Figure 4) consists of 4352 bytes of addressable memory, I/O register locations, and four levels of stack space. This MCU has three separate memory spaces: program space, data space, and stack space.

The MCU is capable of addressing 4096 bytes of program space memory with its program counter and 256 bytes of data space memory with its instructions. Program space memory includes self-check ROM, program ROM, self-check and user program vectors, and reserved memory locations.

A non-accessible subroutine stack space RAM is provided. This stack space consists of a last-in-first-out (LIFO) register. This register is used with inherent addressing to stack the return address for subroutines and interrupts.

Indirect X and Y register locations $80 and $81 are generally used as pointers for such tasks as indirect addressing to data space locations. Short direct addressing allows access to the four data space addresses $80-$83 with single byte opcodes. The operations allowed are increment, decrement, load, and store. Data space locations $82 and $83 can be used for 8-bit counter locations.

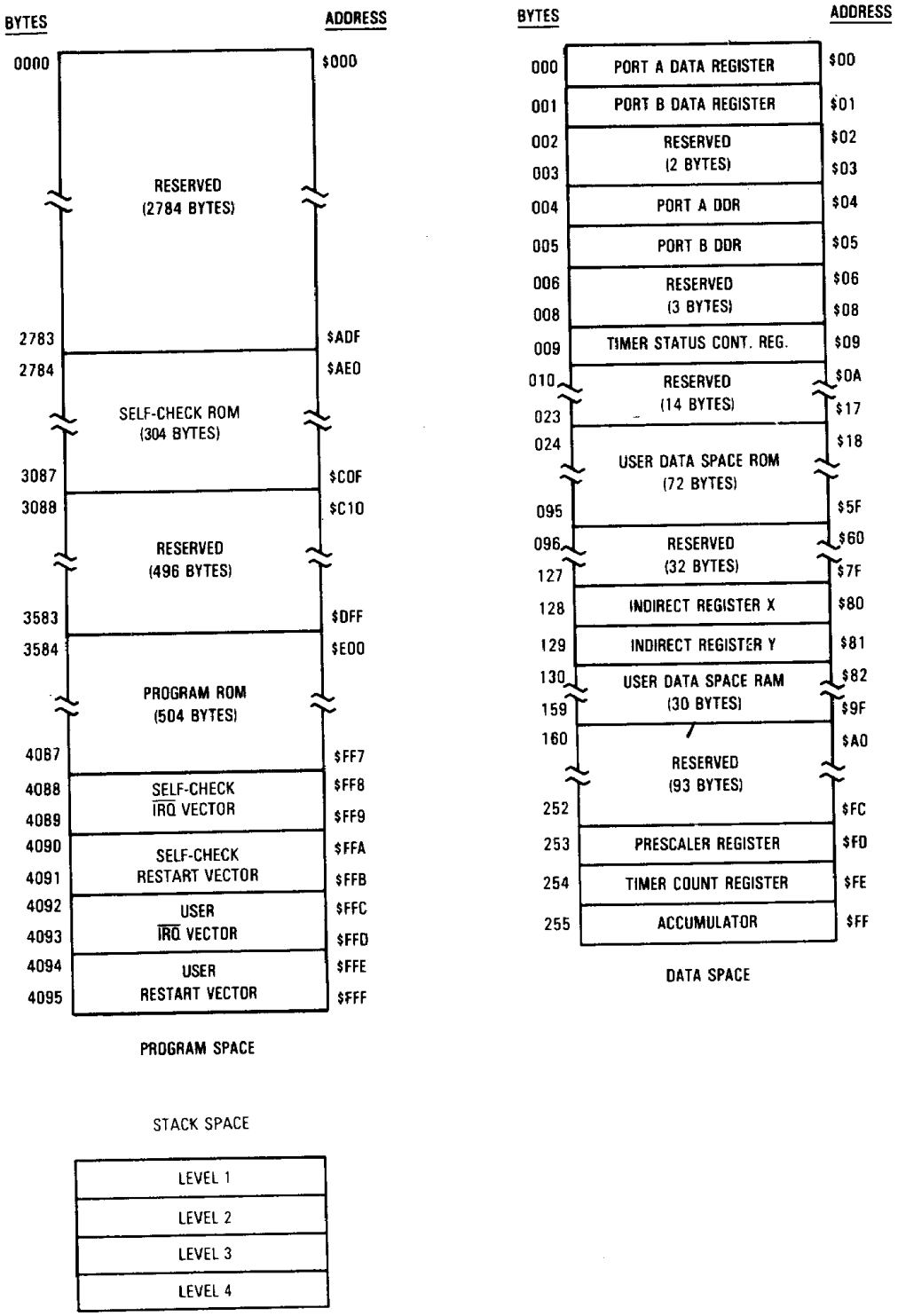
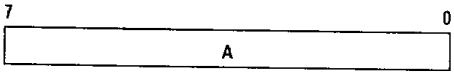


Figure 4. Memory Map

REGISTERS

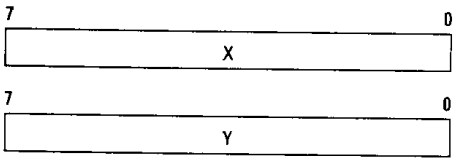
ACCUMULATOR (A)

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



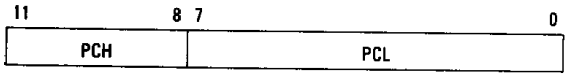
INDIRECT REGISTERS (X,Y)

These two registers are used to maintain pointers to other memory locations in data space. They are used in the register-indirect addressing mode and can be accessed with the direct, indirect, short direct, or bit set/clear modes.



PROGRAM COUNTER (PC)

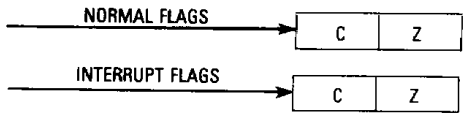
The program counter is a 12-bit register that contains the address of the next byte to be fetched. The program counter is contained in low byte (PCL) and high nibble (PCH).



FLAGS (C,Z)

The first flag, the carry (C) bit, is set on a carry or borrow out of the arithmetic logic unit (ALU). It is cleared if the arithmetic operation does not result in a carry or borrow. The C bit is also set to the value of the bit tested in a bit test instruction. It participates in the rotate left (ROLA) instruction, as well.

The second flag, the zero (Z) bit, is set if the result of the last arithmetic or logic operation was equal to zero. Otherwise, it is cleared. Bit test instructions do not affect the Z bit.



There are two sets of these flags. One set is for interrupt processing (interrupt mode flags). The other set is for normal operations (program mode flags). When an interrupt occurs, a context switch is made from the program flags to the interrupt flags. An RTI forces the context switch back. While in either mode, only the flags for that mode are available. A context switch does not affect the value of the C or Z bits. Both sets of flags are cleared by $\overline{\text{RESET}}$.

STACK

A last-in-first-out (LIFO) stack is incorporated in the MCU that eliminates the need for a stack pointer. This non-accessible subroutine stack space is implemented in separate RAM, 12 bits wide. Whenever a subroutine call or interrupt occurs, the contents of the PC are shifted into the top register of the stack. At the same time, the top register is shifted one level deeper. This happens to all registers, with the bottom register falling out of the stack.

Whenever a return from subroutine or interrupt occurs, the top register is shifted into the PC and all lower registers are shifted one level higher. The stack RAM is four levels deep. If the stack is pulled more than four times with no pushes, then the address that was stored in the bottom level of the stack is shifted into the PC.

SELF CHECK

The MCU implements two forms of internal check: self check and ROM verify. Self check performs an extensive functional check of the MCU using a signature analysis technique. ROM verify uses a similar method to check the contents of program ROM.

Self-check mode is selected by holding the MDS and PA7 pins logic high and the PA6 pin logic low as $\overline{\text{RESET}}$ goes low to high. ROM verify mode is entered by holding MDS, PA7, and PA6 logic high as $\overline{\text{RESET}}$* goes low to high. Unimplemented program space ROM locations are also tested. Monitoring the self-check mode's stages for successful completion requires external circuitry, see *M6804 MCU Manual* (DLE404/D).

RESET

$\overline{\text{RESET}}$

All resets of the MC6804J1 are caused by the external reset input ($\overline{\text{RESET}}$). A reset can be achieved by pulling the $\overline{\text{RESET}}$ pin to logic low for a minimum of 96 oscillator cycles.

During reset, a delay of 96 oscillator cycles is needed before allowing the $\overline{\text{RESET}}$ input to go high. If power is being applied, $\overline{\text{RESET}}$ must be held low long enough for the oscillator to stabilize and then provide the 96 clocks. Connecting a capacitor and resistor to the $\overline{\text{RESET}}$ input, as shown in Figure 5 below, typically provides sufficient delay.

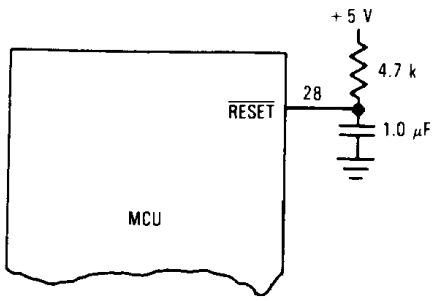


Figure 5. Powerup RESET Delay Circuit

INTERRUPT

The MCU can be interrupted by applying a logic low signal to the $\overline{IRQ}$ pin. However, a manufacturing mask option determines whether the falling edge or the actual low level of the $\overline{IRQ}$ pin is sensed to indicate an interrupt.

EDGE-SENSITIVE OPTION

When the $\overline{IRQ}$ pin is pulled low, the internal interrupt request latch is set. Prior to each instruction fetch, this interrupt request latch is tested. If its output is low, an interrupt sequence is initiated at the end of the current instruction, provided the interrupt mask is cleared. Figure 6 contains a flowchart that illustrates both the reset and interrupt sequences.

The interrupt sequence consists of one cycle during which:

The interrupt request latch is cleared;

The interrupt mode flags are selected;

The program counter (PC) is saved on the stack;

The interrupt mask is set; and

The $\overline{IRQ}$ vector jump address is loaded into the PC.

The $\overline{IRQ}$ vector jump address is $FFC-$FFD in the single-chip mode and $FF8-$FF9 in the self-check mode. The contents of these locations are not decoded as an address to which the PC should jump. Instead, they are decoded like any other EPROM program word. So, it is essential that the vector contents specify a JMP instruction in addition to the starting address of the interrupt service routine. If required, this routine should save the values of the accumulator and the X and Y registers, since these values are not stored on the stack.

Internal processing of the interrupt continues until a return from interrupt (RTI) instruction is processed. During RTI the interrupt mask is cleared and the program mode flags are selected. The next instruction of the program is then fetched and executed.

When the interrupt was initially detected and the interrupt sequence started, the interrupt request latch was cleared so that the next interrupt could be detected. These steps occurred even as the first interrupt was being serviced. However, even though the second interrupt edge set the interrupt request latch during the first interrupt's processing, the second interrupt's sequence can not begin until completion of the interrupt service routine for the first interrupt. Completion of an interrupt service routine is always accomplished using an RTI instruction to return to the main program. The interrupt mask, which is not directly available to the programmer, is cleared during the last cycle of the RTI instruction.

LEVEL-SENSITIVE OPTION

Actual operation of the level-sensitive and edge-sensitive options are similar. However, the level-sensitive option does not have an interrupt request latch. Since there is no interrupt request latch, the logic level of the $\overline{IRQ}$ pin is checked to detect the interrupt. Also, in the interrupt sequence there is no need to clear the interrupt request latch. These differences are shown in Figure 6.

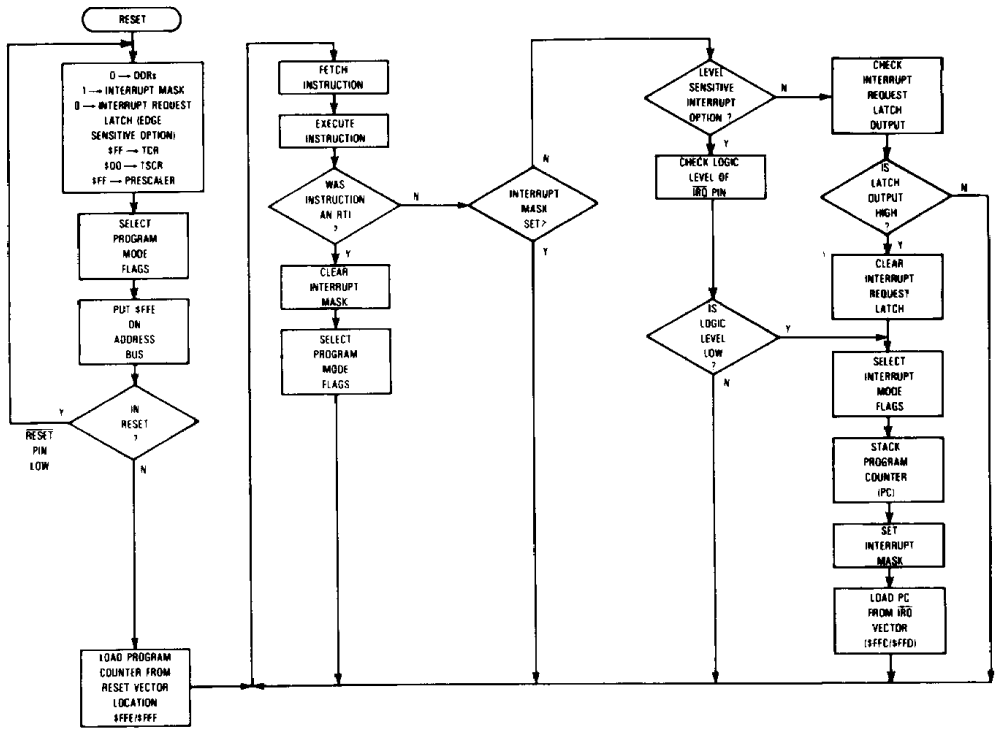


Figure 6. Reset and Interrupt Flowchart

POWERUP AND TIMING

During the powerup sequence, the interrupt mask is closed. This precludes any false interrupts. The PC is also loaded with the appropriate restart vector (jump instruction).

To open the interrupt mask, the user should do a JSR to an initialization subroutine that ends with an RTI instead of an RTS. The RTI opens the interrupt mask. Typical $\overline{\text{RESET}}$ and $\overline{\text{IRQ}}$ processes and their relationship to the interrupt mask are shown in Figure 7.

Maximum interrupt response time is six machine cycles. This includes five cycles for the longest instruction plus one for stacking the PC and switching flags.

TIMER

A block diagram of the MC6804J1 timer circuitry is shown in Figure 8. The timer logic in the MCU is comprised of a simple 8-bit counter called the timer counter. This counter is decremented by a 7-bit prescaler at a rate determined by the timer status/control register (TSCR).

PRESCALER

The prescaler is a 7-bit counter used to extend the maximum interval of the overall timer. This counter is clocked by a signal from the TIMER pin or by the internal sync pulse. It divides the frequency received by some factor

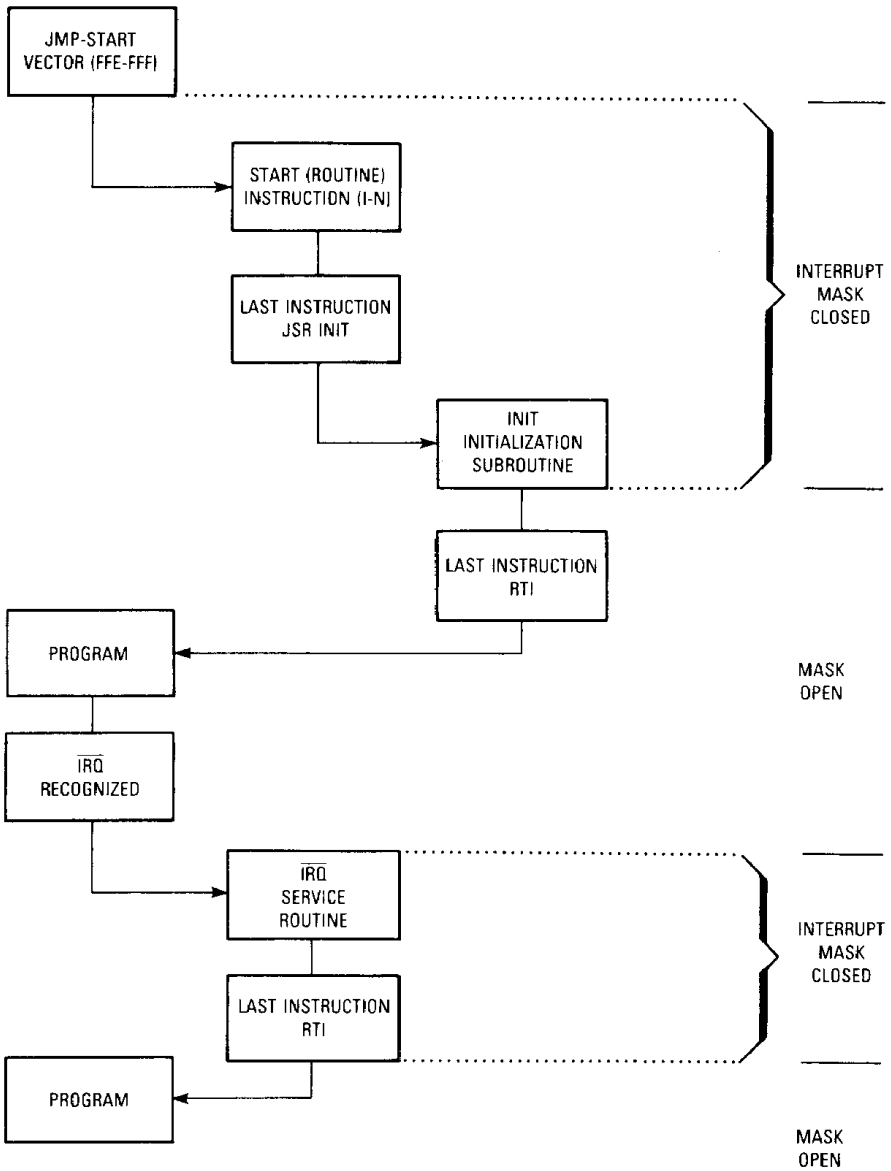


Figure 7. Interrupt Mask

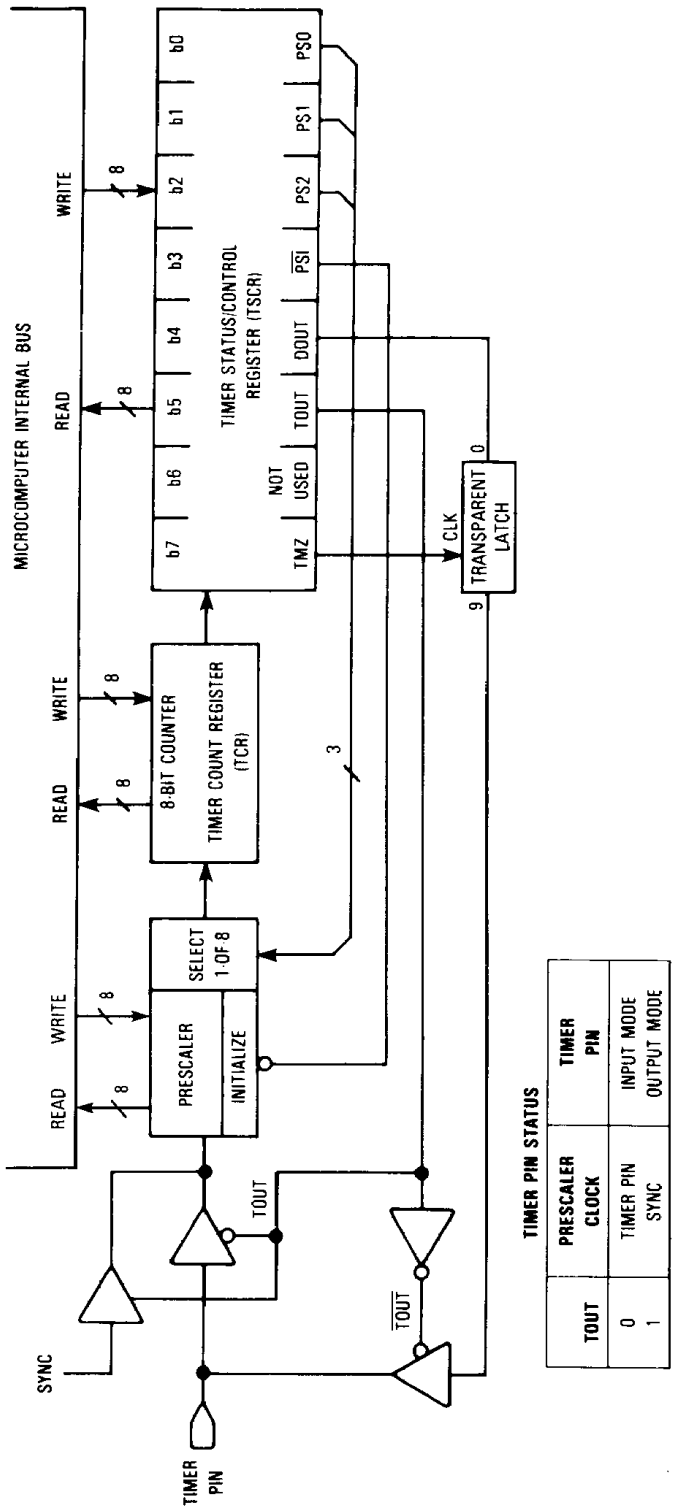


Figure 8. Timer Block Diagram

to create the prescaler output. The factor by which the TIMER pin signal is divided is called the prescaler tap. The value of this tap is selected by three bits of the TSCR (PS0-PS2). These bits control the division of the prescaler input within the range of divide-by-2^0, to divide-by-2^7.

TIMER COUNTER

The timer counter, which may be read or loaded under program control, is decremented from a maximum value of 256 toward zero by the prescaler output. Both are decremented on rising clock edges.

The prescaler register and timer count register are readable and writeable. A write to either one will take precedence over the normal counter function. For example, if a value is written to the timer count register, and this write and a decrement-to-zero occur at the same time, the write takes precedence. TSCR bit one (TMZ) is not set until the next timer time out.

TIMER PIN

The TIMER pin may be programmed as either an input or an output. Its status depends on the value of TSCR bit 5 (TOUT). This relationship is shown in the TIMER pin status section of Figure 8. The frequency of the internal clock applied to the TIMER pin must be less than t_{byte}, which is (f_{osc}/48).

TIMER INPUT MODE

In the timer input mode, TOUT is logic zero and the TIMER pin is connected directly to prescaler input. So, the prescaler is clocked by the signal from the TIMER pin. The prescaler divides the TIMER pin clock input by the prescaler tap. The prescaler output then clocks the 8-bit timer count register. When this register is decremented to zero, it sets TSCR bit one (TMZ). This TMZ bit can be tested under program control.

TIMER OUTPUT MODE

In the output mode, the TIMER pin is output. TOUT is a logic one. The prescaler is clocked by the internal sync pulse. This pulse is a divide-by-48 of the internal oscillator (f_{osc}/48). From this point on, operation is similar to that described for the input mode. However, in the output mode, once the prescaler decrements the timer counter to zero, the high TMZ bit state is used to latch the data at TSCR bit 4 (DOUT), onto the TIMER pin.

NOTE

TMZ is normally set to logic one when TCR decrements to zero and the timer times out. However, it may be set by a write of $00 to the timer counter or by a write to bit 7 of TSCR.

TIMER COUNT REGISTER ($FE)

The timer count register reflects the current count in the internal 8-bit counter. The register is the timer counter and can be read or written.

7							0
MSB							LSB

RESET:

1	1	1	1	1	1	1	1

TIMER STATUS/CONTROL REGISTER (TSCR) ($09)

7	6	5	4	3	2	1	0
TMZ	—	TOUT	DOUT	$\overline{PSI}$	PS2	PS1	PS0

RESET:

0	0	0	0	0	0	0	0

TMZ — Timer Zero
- 1 = Timer count register has decremented to zero since the last time the TMZ bit was read.
- 0 = This bit is cleared by a read of the TSCR if TMZ is read as logic one.

Bit 6
Not used by this register.

TOUT — Timer Output
- 1 = Output mode is selected for the timer.
- 0 = Input mode is selected for the timer.

DOUT — Data Output
Latched data at this bit is sent to the TIMER pin when both the TMZ and TOUT bits are logic high.

$\overline{PSI}$ — Prescaler Initialization
- 1 = Prescaler begins to decrement.
- 0 = Prescaler is initialized and counting is inhibited.

PS0-PS2
These bits are used to select the prescaler tap. The coding of the bits is shown below:

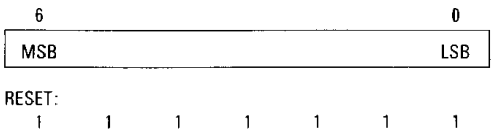
PS2	PS1	PS0	Divide By
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

It is recommended that MVI or loading and storing instructions be used when changing bit values in the TSCR. Read-modify-write instructions can cause the TMZ to assume an unexpected state.

During reset, the TSCR is set to all zeroes; the TIMER pin is in the high impedance input mode; and DOUT LATCH is forced to a logic high. At the same time, PS0-PS2 coding sets the prescaler tap at divide-by-one, and bit 3 initializes the prescaler.

TIMER PRESCALER REGISTER ($FD)

The timer prescaler register reflects the current count of the 7-bit prescaler. This register is the prescaler counter and can be read or written.



INSTRUCTION SET

The MCU has a set of 42 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.

3

REGISTER/MEMORY INSTRUCTIONS

Most of these instructions use two operands. One operand is the accumulator; the other is obtained from memory using one of the addressing modes. Refer to the following list of instructions.

Function	Mnemonic
Load A from Memory	LDA
Load XP from Memory	LDX
Load YP from Memory	LDY
Store A in Memory	STA
Add to A	ADD
Subtract from A	SUB
AND Memory to A	AND
Transfer A to XP	TAX
Transfer A to YP	TAY
Transfer YP to A	TYA
Transfer XP to A	TPA
Clear A	CLRA
Clear XP	CLRX
Clear YP	CLRY
Arithmetic Compare with Memory	CMP
Move Immediate Value to Memory	MVI
Arithmetic Left Shift of A	ASLA
Complement A	COMA
Rotate A Left and Carry	ROLA

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. All INC and DEC forms along with all bit manipulation instructions use this method. Refer to the following list of instructions.

Function	Mnemonic
Increment Memory Location	INC
Increment A	INCA
Increment XP	INCX
Increment YP	INCY
Decrement Memory Location	DEC
Decrement A	DECA
Decrement XP	DECX
Decrement YP	DECY

BRANCH INSTRUCTIONS

This set of instructions branches if a particular condition is met; otherwise, no operation is performed. Branch instructions are two byte instructions. Refer to the following list of instructions.

Function	Mnemonic
Branch if Carry Clear	BCC
Branch if Higher or Same	(BHS)
Branch if Carry Set	BCS
Branch if Lower	(BLO)
Branch if Not Equal	BNE
Branch if Equal	BEQ

BIT MANIPULATION INSTRUCTIONS

The MCU is capable of setting or clearing any bit which resides in the 256 bytes of data space, where all port registers, port DDRs, timer, timer control, and on-chip RAM reside. An additional feature allows the software to test and branch on the state of any bit within these 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 also placed in the carry bit of the condition code register. Refer to the following list of instructions.

Function	Mnemonic
Branch If Bit n is Set	BRSET n(n = 0 . . . 7)
Branch If Bit n is Clear	BRCLR n(n = 0 . . . 7)
Set Bit n	BSET n(n = 0 . . . 7)
Clear Bit n	BCLR n(n = 0 . . . 7)

CONTROL INSTRUCTIONS

These instructions are used to control processor operation during program execution. The jump conditional

(JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to the following list of instructions.

Function	Mnemonic
Return from Subroutine	RTS
Return from Interrupt	RTI
No Operation	NOP
Jump to Subroutine	JSR
Jump Unconditional	JMP

IMPLIED INSTRUCTIONS

Since the accumulator and all other registers are located in RAM, many implied instructions exist. Some of the instructions recognized and translated by the assembler are shown below:

Mnemonic	Becomes	Mnemonic	Becomes
ASLA	ADD $FF	INCX	INC $80
BHS	BCC	INCY	INC $81
BLO	BCS	LDXI	MVI $80 DATA
CLRA	SUB $FF	LDYI	MVI $81 DATA
CLRX	MVI $80 #0	NOP	BEQ (PC) + 1
CLRY	MVI $81 #0	TAX	STA $80
DECA	DEC $FF	TAY	STA $81
DECX	DEC $80	TXA	LDA $80
DECY	DEC $81	TYA	LDA $81
INCA	INC $FF		

Some examples of valuable instructions not specifically recognized by the assembler are shown below:

Mnemonic	Meaning
BCLR 7,$FF	Ensures A is plus
BSET 7, $FF	Ensures A is minus
BRCLR 7, $FF	Branch if A is plus
BRSET 7, $FF	Branch if A is minus
BRCLR 7, $80	Branch if X is plus (BXPL)
BRSET 7, $80	Branch if X is minus (BXMI)
BRCLR 7, $81	Branch if Y is plus (BYPL)
BRSET 7, $81	Branch if Y is minus (BYMI)

OPCODE MAP

Table 1 is a listing of all the instruction set opcodes applicable to the MC6804J1 MCU.

ADDRESSING MODES

The MCU has nine different addressing modes to provide the programmer with an opportunity to optimize the code for all situations. It deals with objects in three different address spaces: program space, data space, and stack space. The term "effective address" (EA) is used in describing the various addressing modes. Effective address is defined as the address from which the argument for an instruction is fetched or stored.

IMMEDIATE

In the immediate addressing mode, the operand is located in program ROM. It is contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants that do not change during program execution, such as a constant used to initialize a loop counter.

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 256 bytes of data space with a single two-byte instruction.

SHORT DIRECT

In the short direct addressing mode, the MCU has four locations in data space RAM it can use, ($80, $81, $82, and $83). The opcode determines the data space RAM location, and the instruction is only one byte. Short direct addressing is a subset of the direct addressing mode. The X and Y registers are at locations $80 and $81, respectively.

EXTENDED

In the extended addressing mode, the effective address of the argument is obtained by concatenating the four least-significant bits of the opcode with the byte following the opcode to form a 12-bit address. Instructions using the extended addressing mode, such as JMP or JSR, are capable of branching anywhere in program space. An extended addressing mode instruction is two bytes long.

RELATIVE

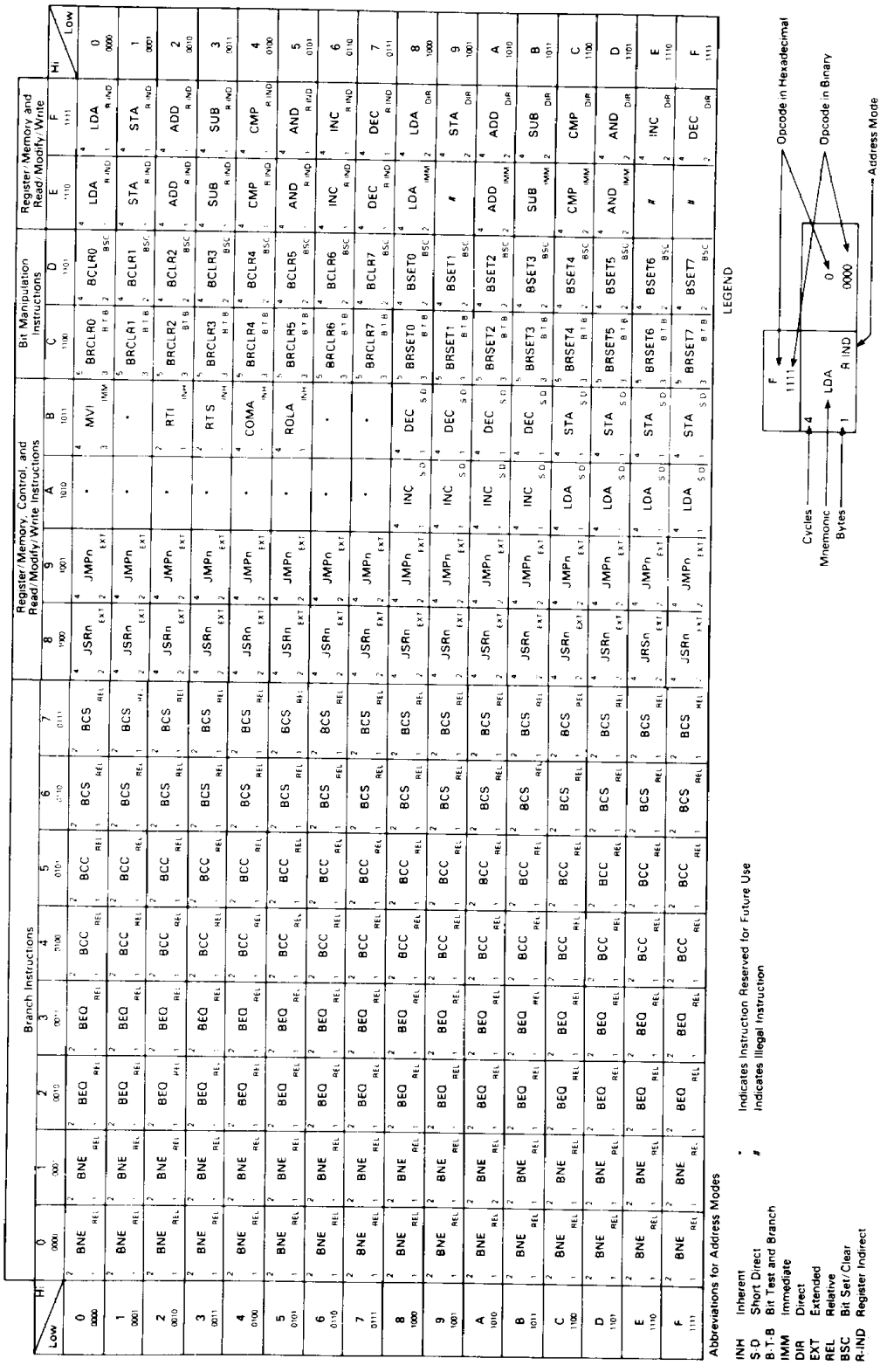
The relative addressing mode is only used in conditional branch instructions. In relative addressing, the contents of the 8-bit signed byte (the offset) following the opcode is added to the PC if, and only if, the branch conditions are true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from −15 to +16 from the opcode address. The programmer need not calculate the offset when using the Motorola assembler, since it calculates the proper offset and checks to see that it is within the span of the branch.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode. The byte following the opcode specifies the direct addressing of the byte in which the specified bit is to be set or cleared. Thus, any bit in the 256 locations of data space memory that can be written to can be set or cleared with a single two-byte instruction.

3

Table 1. Opcode Map



CAUTION

The corresponding DDRs for ports A and B are write only registers (registers at $04 and $05). A read operation on these registers is undefined. Since BSET and BCLR are read-modify-write functions, they cannot be used to set or clear a DDR bit; all "unaffected" bits would be set. Write all DDR bits in a port using a single-store instruction.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit that is to be tested and its condition (set or clear) is included in the opcode. The data space address of the byte to be tested is in the single byte immediately following the opcode byte. The third byte is sign extended to 12 bits and becomes the offset added to the PC if the condition is true. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the 256 locations of data space. The span of branching is from −125 to +130 from the opcode address. The state of the tested bit is also transferred to the carry flag.

REGISTER-INDIRECT

In the register-indirect addressing mode, the operand is at the address in data space pointed to by the contents of one of the indirect registers, X or Y. The particular indirect register is selected by bit 4 of the opcode. Bit 4 decodes into an address that represents the register, $80 or $81. A register-indirect instruction is one byte long.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. These instructions are one byte long.

3

ELECTRICAL SPECIFICATIONS

MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range (Comm.)	T_A	0 to 70	°C
Operating Temperature Range (Ind.)	T_A	−40 to +85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Junction Temperature	T_J	150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or 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 that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{CC}$. Reliability of operation is enhanced if unused inputs except EXTAL are connected to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance	θ_{JA}	70	°C/W

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:

T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{PORT}$
P_{INT} = $I_{CC} \times V_{CC}$, Watts — Chip Internal Power
P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.0 Vdc ± 0.5 Vdc, V_{SS} = GND, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Internal Power Dissipation — No Port Loading	P_{INT}	—	120	165	mW
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Input Capacitance	C_{in}	—	10	—	pF
Input Current ($\overline{IRQ}$, $\overline{RESET}$)	I_{in}	—	2	20	μA

SWITCHING CHARACTERISTICS

(V_{CC} = +5.0 Vdc ± 0.5 Vdc, V_{SS} = GND, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Oscillator Frequency	f_{osc}	4.0	—	11.0	MHz
Bit Time	t_{bit}	0.364	—	1.0	μs
Byte Cycle Time	t_{byte}	4.36	—	12.0	μs
$\overline{IRQ}$ and TIMER Pulse Width	t_{WL}, t_{WH}	$2 \times t_{byte}$	—	—	—
$\overline{RESET}$ Pulse Width	t_{RWL}	$2 \times t_{byte}$	—	—	—
$\overline{RESET}$ Delay Time (External Capacitance = 1.0 μF)	t_{RHL}	100	—	—	ms

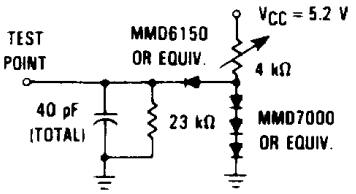


Figure 9. LSTTL Equivalent Test Load (Port B)

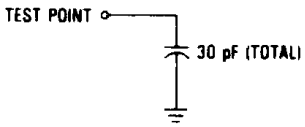


Figure 10. CMOS Equivalent Test Load (Ports A, B, C)

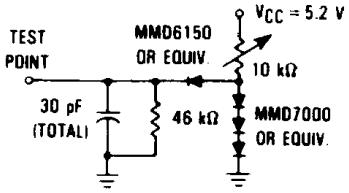


Figure 11. LSTTL Equivalent Test Load (Ports A, C, and TIMER)

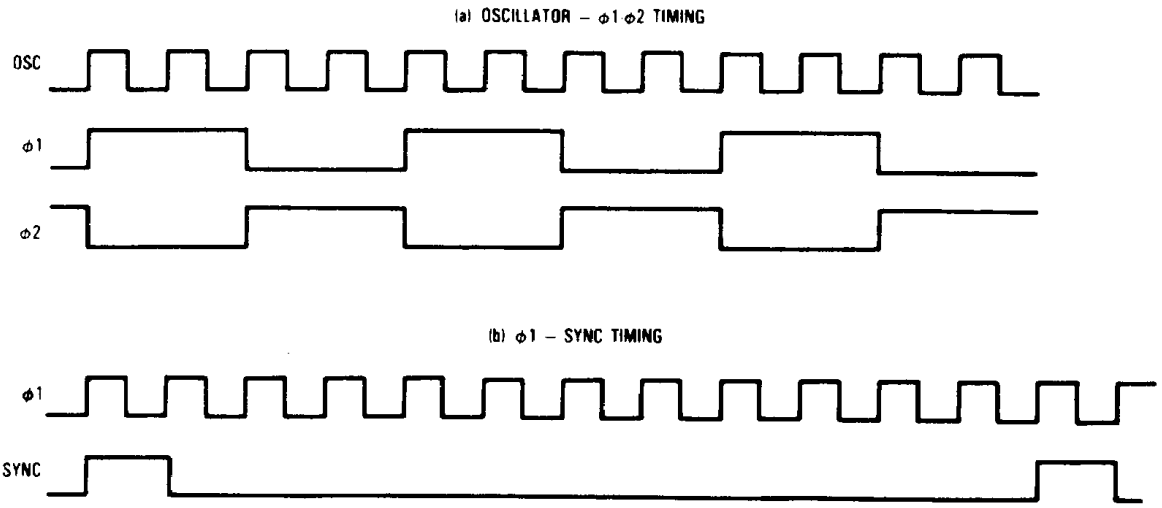


Figure 12. Clock Generator Timing Diagram

PORT DC ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.0 Vdc ±0.5 Vdc, V_{SS} = GND, T_A = 0°C to 70°C, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Ports A and Timer (Standard)					
Output Low Voltage, I_{Load} = 0.4 mA	V_{OL}	—	—	0.5	V
Output High Voltage, I_{Load} = −50 μA	V_{OH}	2.3	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	4	40	μA
Port A (Open Drain)					
Output Low Voltage, I_{Load} = 0.4 mA	V_{OL}	—	—	0.5	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	4	40	μA
Open Drain Leakage (V_{out} = V_{CC})	I_{LOD}	—	4	40	μA
Port A (CMOS Drive)					
Output Low Voltage, ILoad = 0.4 mA (Sink)	V_{OL}	—	—	0.5	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	V_{CC} − 1.0	—	—	V
Output High Voltage, I_{Load} = −50 μA	V_{OH}	2.3	—	—	V
Input High Voltage, I_{Load} = −300 μA Max	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage, I_{Load} = −300 μA Max	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current (V_{in} = 0.4 V to V_{CC})	I_{TSI}	—	—	−300	μA
Port B (Standard)					
Output Low Voltage, I_{Load} = 1.0 mA	V_{OL}	—	—	0.5	V
Output Low Voltage,I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.5	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.3	—	—	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	8	80	μA
Port B (Open Drain)					
Output Low Voltage, I_{Load} = 1.0 mA	V_{OL}	—	—	0.5	V
Output Low Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.5	V
Input High Voltage	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current	I_{TSI}	—	8	80	μA
Open Drain Leakage (V_{out} = V_{CC})	I_{LOD}	—	8	80	μA
Port B (CMOS Drive)					
Output Low Voltage, I_{Load} = 1.0 mA	V_{OL}	—	—	0.5	V
Output High Voltage, I_{Load} = 10 mA (Sink)	V_{OL}	—	—	1.5	V
Output High Voltage, I_{Load} = −10 μA	V_{OH}	V_{CC} − 1.0	—	—	V
Output High Voltage, I_{Load} = −100 μA	V_{OH}	2.3	—	—	V
Input High Voltage, I_{Load} = −300 μA Max	V_{IH}	2.0	—	V_{CC}	V
Input Low Voltage, I_{Load} = −300 μA Max	V_{IL}	−0.3	—	0.8	V
Hi-Z State Input Current (V_{in} = 0.4 V to V_{CC})	I_{TSI}	—	—	−300	μA
Ports A and B (Low Current Clamping Diode*)					
Input High Current V_{IH} = V_{CC} + 1.0 V	I_{IH}	—	—	100	μA
Input Low Current V_{IL} = 0.8 V	I_{IL}	—	—	−4.0	μA

*Denotes not tested unless specified on ordering form.

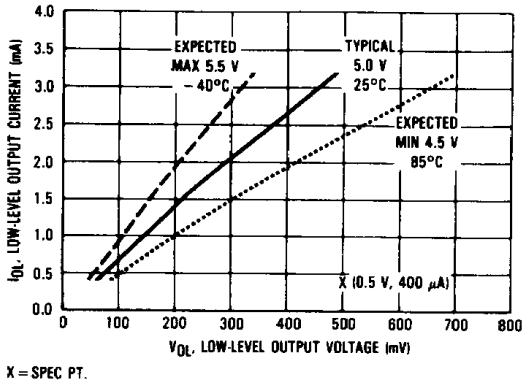


Figure 13. Typical V_{OL} vs I_{OL} for Port A and TIMER

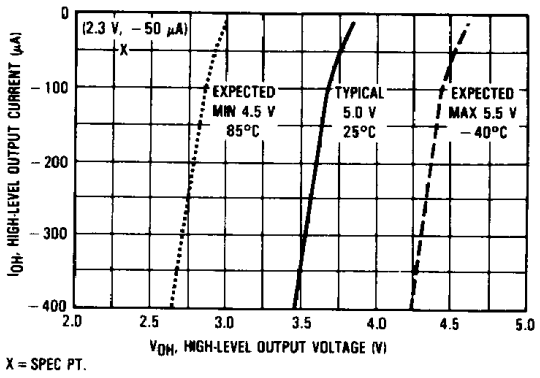


Figure 14. Typical V_{OH} vs I_{OH} for Port A and TIMER

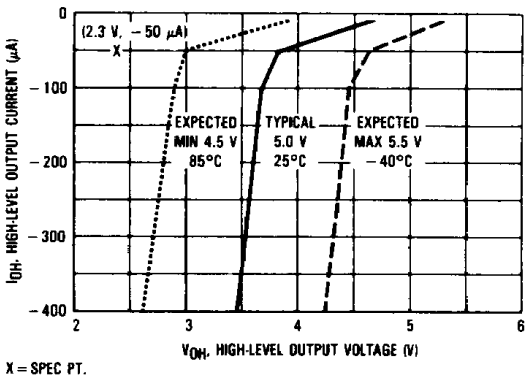


Figure 15. Typical V_{OH} vs I_{OH} for Port A with CMOS Pullups

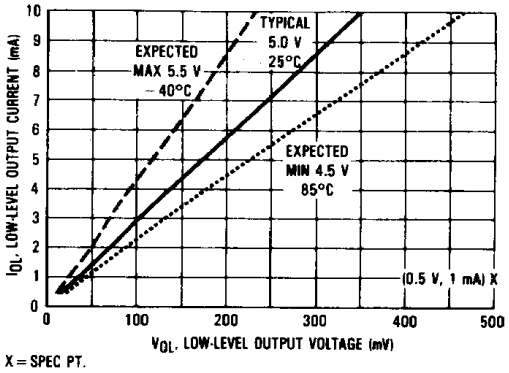


Figure 16. Typical V_{OL} vs I_{OL} for Port B

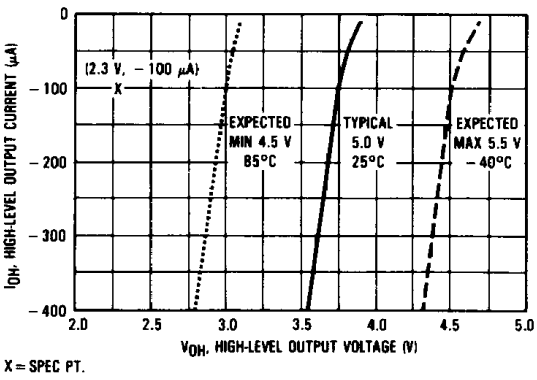


Figure 17. Typical V_{OH} vs I_{OH} for Port B

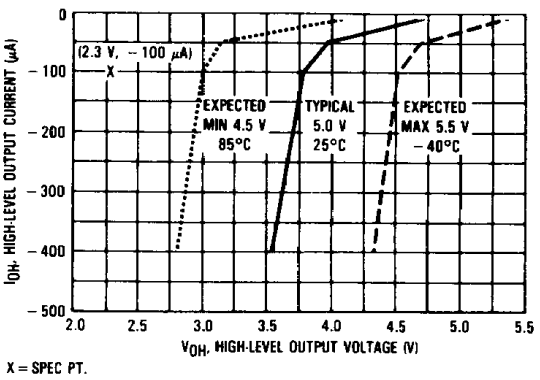


Figure 18. Typical V_{OH} vs I_{OH} for Port B with CMOS Pullups

ORDERING INFORMATION

The following information is required when ordering a custom MCU. The information may be transmitted to Motorola using the following media:

MDOS™, disk file
MS™-DOS/PC-DOS disk file (360K)
EPROM(s) 2516, 2716, 2532, 2732

To initiate a ROM pattern for the MCU, it is necessary to first contact the local field service office, sales person, or Motorola representative.

FLEXIBLE DISKS

Several types of flexible disks (MDOS or MS-DOS/PC-DOS disk file) may be submitted for pattern generation. They should be programmed with the customer program, using positive logic sense for address and data. The diskette should be clearly labeled with the customer's name, date, project or product name, and the filename containing the pattern.

In addition to the program pattern, a file containing the program source code listing can be included. This data will be kept confidential and used to expedite the process in case of any difficulty with the pattern file.

MDOS Disk File

MDOS is Motorola's Disk Operating System available on the EXORciser® development system. The disk media submitted must be a single-sided, single-density, 8-inch MDOS compatible floppy diskette. The diskette must contain the minimum set of MDOS system files in addition to the pattern file.

The .LO output of the M6804 cross assembler should be furnished. In addition, the file must be produced using the ROLLOUT command, so that it contains the absolute image of the M6804 memory. It is necessary to include the entire memory image of both program and data space. All unused bytes, including those in the user space, must be set to logic zero.

MS-DOS/PC-DOS Disk File

MS-DOS is Microsoft's Disk Operating System. PC-DOS is IBM®'s Personal Computer Disk Operating System. Disk media submitted must be standard density (360K), double-sided 5 1/4 inch compatible floppy diskette. The diskette must contain the object file code in Motorola's S-record format. The S-record format is a character-based object file format generated by M6804 cross assemblers and linkers on IBM PC style machines.

EPROMS

Four K of EPROM are necessary to contain the entire MC6804J1 program. Two 2516 or 2716 type EPROMs or a single 2532 or 2732 type EPROM can be submitted for pattern generation. The EPROM is programmed with the customer program using positive logic sense for address and data. Submissions on two EPROMs must be clearly marked. All unused bytes, including the user's space, must be set to zero.

If the MC6804J1 MCU ROM pattern is submitted on one 2532 or 2732 EPROM, or on two 2516 or 2716 type EPROMs, memory map addressing is one-for-one. The data space ROM runs from EPROM address $018 to $05F and program space ROM runs from EPROM address $E00 to $FF7, with vectors from $FFC to $FFF.

For shipment to Motorola, EPROMs should be placed in a conductive IC carrier and packed securely. Styrofoam is not acceptable for shipment.

Verification Media

All original pattern media, EPROMs or floppy disks, are filed for contractual purposes and are not returned. A computer listing of the ROM code will be generated and returned along with a listing verification form. The listing should be thoroughly checked and the verification form should be completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for the creation of the customer mask. To aid in the verification process, Motorola will program customer supplied blank EPROM(s) or DOS disks from the data file used to create the custom mask.

ROM Verification Units (RVUs)

Ten MCUs containing the customers ROM pattern will be sent for program verification. These units will have been made using the custom mask, but are for the purpose of ROM verification only. For expediency, the MCUs are unmarked, packaged in ceramic, and tested with five volts at room temperature. These RVUs are free with the minimum order quantity, but are not production parts. These RVUs are not guaranteed by Motorola Quality Assurance.

Ordering Information

The following table provides generic information pertaining to the package type and temperature for the MC6804J1. This MCU device is available in the 20-pin dual-in-line (DIP) package.

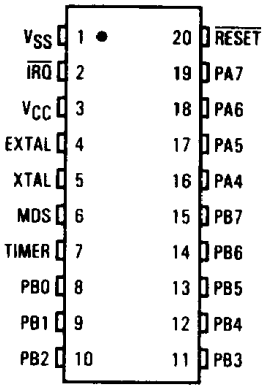
Generic Information

Package Type	Temperature	Order Number
Plastic (P Suffix)	0°C to 70°C	MC6804J1P
	−40°C to +85°C	MC6804J1CP

MDOS is a trademark of Motorola Inc.
MS-DOS is a trademark of Microsoft, Inc.
EXORciser is a registered trademark of Motorola Inc.
IBM is a registered trademark of International Business Machines Corporation.

MECHANICAL DATA

PIN ASSIGNMENTS



3