MOTOROLA

MC6805T2

Advance Information

HMOS

(HIGH DENSITY, N-CHANNEL, SILICON-GATE, DEPLETION LOAD)

8-BIT MICROCOMPUTER WITH PLL LOGIC

8-BIT MICROCOMPUTER UNIT WITH PLL LOGIC

The MC6805T2 Microcomputer Unit (MCU) with PLL logic is a member of the M6805 HMOS Family of low-cost single-chip microcomputers. This 8-bit microcomputer contains a CPU, on-chip clock, ROM, RAM, I/O, timer, and the PLL logic for an RF synthesizer. It is designed for the user who needs an economical microcomputer with the proven capabilities of the M6800-based instruction set, as well as the necessary logic required for frequency synthesis applications. The following are some of the hardware and software highlights of the MC6805T2 MCU.

HARDWARE FEATURES

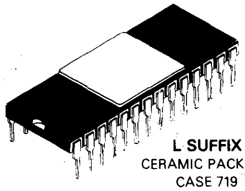
- 8-Bit Architecture
- 64 Bytes of RAM
- Memory Mapped I/O
- 2508 Bytes of User ROM
- Timer Start/Stop and Source Select
- 19 TTL/CMOS Compatible Bidirectional I/O Lines (Eight Lines are LED Compatible)
- On-Chip Ciock Generator
- Zero-Crossing Detection
- Self-Check Mode
- Master Reset
- Complete Development System Support on EXORciser
- 5 V Single Supply
- 14-Bit Binary Variable Divider
- 10-Stage Mask-Programmable Reference Divider
- Three-State Phase and Frequency Comparator
- Suitable for TV Frequency Synthesizers

SOFTWARE FEATURES

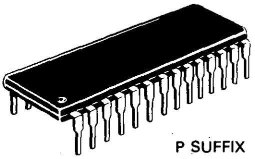
- Similar to M6800
- Byte Efficient Instruction Set
- Easy to Program
- True Bit Manipulation
- Bit Test and Branch Instructions
- Versatile Interrupt Handling
- Versatile Index Register
- Powerful Indexed Addressing for Tables
- Full Set of Conditional Branches
- Memory Usable as Registers/Flags
- Single Instruction Memory Examine/Change
- 10 Powerful Addressing Modes
- All Addressing Modes Apply to ROM, RAM, and I/O

USER SELECTABLE OPTIONS

- Internal 8-Bit Timer with Selectable Clock Source (External Timer Input or Internal Machine Clock)
- Timer Prescaler Option (Seven Bits 2^n)
- Eight Bidirectional I/O Lines with TTL or TTL/CMOS Interface Option or Open-Drain Drive
- Four Vectored Interrupts; Timer, Software, and two External
- Low Voltage Inhibit Option

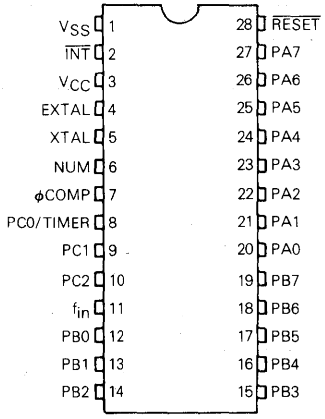


L SUFFIX
CERAMIC PACKAGE
CASE 719



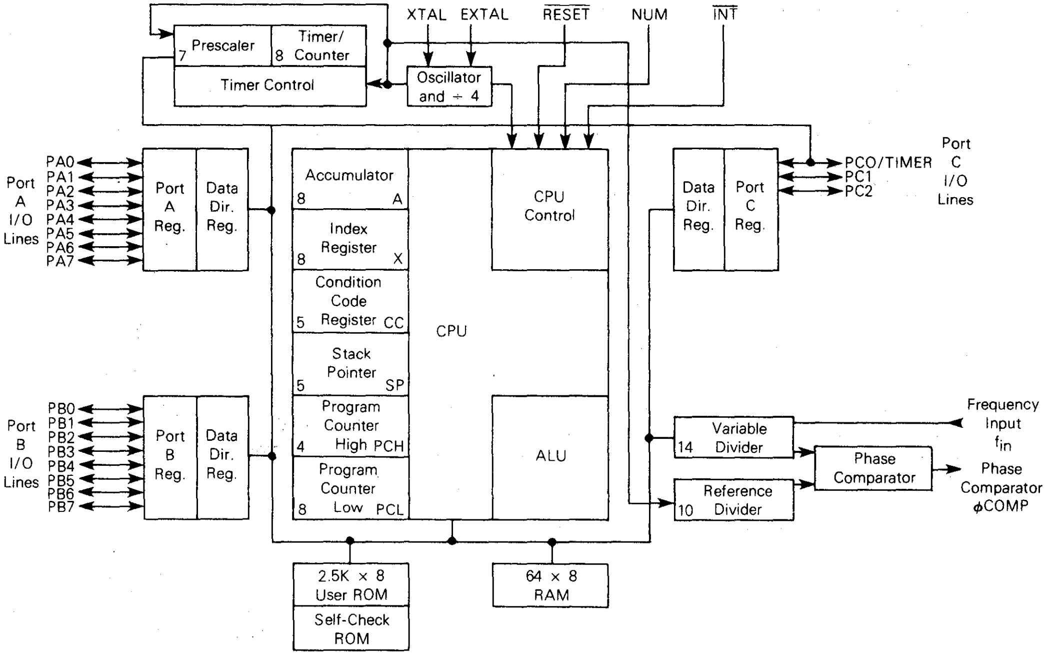
P SUFFIX
PLASTIC PACKAGE
CASE 710

PIN ASSIGNMENT



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

FIGURE 1 — MC6805T2 HMOS MICROCOMPUTER BLOCK DIAGRAM



MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage (Except Pin 6)	V_{in}	−0.3 to +7.0	V
Operating Temperature Range	T_A	0 to +70	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance			
Plastic		120	
Ceramic	θ_{JA}	50	°C/W
Cerdip		60	

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 are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

POWER CONSIDERATIONS

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

$T_J = T_A (P_D \bullet \theta_{JA})$ (1)

Where:

$T_A \equiv$ Ambient Temperature, °C

$\theta_{JA} \equiv$ Package Thermal Resistance, Junction-to-Ambient, °C/W

$P_D \equiv P_{INT} + P_{PORT}$

$P_{INT} \equiv I_{CC} \times V_{CC}$, Watts — Chip Internal Power

$P_{PORT} \equiv$ Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} \ll P_{INT}$ 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:

$P_D = K \div (T_J + 273°C)$ (2)

Solving equations 1 and 2 for K gives:

$K = P_D \bullet (T_A + 273°C) + \theta_{JA} \bullet P_D^2$ (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} = 5.25 Vdc ± 0.5 Vdc, V_{SS} = GND, T_A = 0 to 70°C unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage $\overline{\text{RESET}}$ (4.75 ≤ V_{CC} ≤ 5.75) (V_{CC} < 4.75) $\overline{\text{INT}}$ (4.75 ≤ V_{CC} ≤ 5.75) (V_{CC} < 4.75) All Other Except f_{in}	V_{IH}	 4.0 V_{CC} − 0.5 4.0 V_{CC} − 0.5 2.0	 – – * * –	 V_{CC} V_{CC} V_{CC} V_{CC} V_{CC}	V
Input High Voltage ϕCOMP Normal Mode Self-Check Mode	V_{IH}	 – –	 – 10.0	 V_{CC} + 1 10.0	V
Input Low Voltage $\overline{\text{RESET}}$ $\overline{\text{INT}}$ All Other Except f_{in}	V_{IL}	 V_{SS} V_{SS} V_{SS}	 – * –	 0.8 1.5 0.8	V
$\overline{\text{INT}}$ Zero-Crossing Input Voltage, through a Capacitor	V_{INT}	2.0	–	4.0	$V_{ac\ p\text{-}p}$
Internal Power Dissipation – No Port Loading, V_{CC} = 5.75 V, T_A = 0°C	P_{INT}	–	400	–	mW
Input Capacitance XTAL All Others	C_{in}	 – –	 25 10	 – –	pF
AC Coupled Input Voltage Swing on f_{in}	V_{FIP}	0.5	1.2	–	$V_{ac\ p\text{-}p}$
Input Current (V_{IH} = V_{CC}) on Pin 11 (f_{in})	I_{FH}	–	–	40	µA
Output Low Current (V_{OL} = 1.0 V) on Pin 7 (ϕCOMP)	I_{CML}	–	−300	–	µA
Output High Current (V_{OH} = V_{CC} − 1 V) on Pin 7 (ϕCOMP)	I_{CMH}	–	200	–	µA
Leakage Current (V_{in} = V_{CC}) on Pin 7 (ϕCOMP)	I_{OFF}	–	2	–	nA
$\overline{\text{RESET}}$ Hysteresis Voltage (See Figures 10 and 11) "Out of Reset" "Into Reset"	 V_{IRES+} V_{IRES-}	 2.1 0.8	 – –	 4.0 2.0	V
Input Current TIMER (V_{in} = 0.4 V) $\overline{\text{INT}}$ (V_{in} = 0.4 V) EXTAL (V_{in} = 2.4 V to V_{CC}) (V_{in} = 0.4 V) $\overline{\text{RESET}}$ (V_{in} = 0.8 V) (External Capacitor Charging Current)	I_{in}	 – – – – −4.0	 – 20 – – –	 20 50 10 −1600 −40	µA
Low Voltage Receiver	V_{LVR}	–	–	4.75	V
Low Voltage Inhibit 0°C to 70°C −40°C to 85°C	V_{LVI}	2.75 3.1	3.5 3.5	– –	V

See MC68(7)05 Series Data Sheet for port I/V curves and input protection schematics.

*Due to internal biasing, this $\overline{\text{INT}}$ input (when unused) floats to approximately 2.0 volts.

SWITCHING CHARACTERISTICS (V_{CC} = 5.25 Vdc ± 0.5 Vdc, V_{SS} = GND, T_A = 0 to 70°C unless otherwise noted)

Characteristics	Symbol	Min	Typ	Max	Unit
Oscillator Frequency	f_{osc}	0.4	–	4.2	MHz
Cycle Time (4/f_{osc})	t_{cyc}	0.95	–	10	µs
$\overline{\text{INT}}$ and TIMER Pulse Width (See TIMER and INTERRUPT Sections)	t_{WH}, t_{WL}	t_{cyc} + 250	–	–	ns
$\overline{\text{RESET}}$ Pulse Width	t_{RWL}	t_{cyc} + 250	–	–	ns
$\overline{\text{RESET}}$ Delay Time (External Capacitance = 1.0 µF)	t_{RHL}	–	100	–	ms
Input Frequency	f_{in}	1	–	16	MHz
Input Frequency Rise Time (f_{in} = max)	t_{INR}	–	–	20	ns
Input Frequency Fall Time (f_{in} = max)	t_{INF}	–	–	20	ns
Duty Cycle of f_{in}		40	–	60	%
Injection Pulse Active Time	t_{ERR}	–	70	–	ns
$\overline{\text{INT}}$ Zero-Crossing Detection Input Frequency	f_{INT}	0.03	–	1.0	kHz
External Clock Input Duty Cycle (EXTAL)	–	40	50	60	%

PORT ELECTRICAL CHARACTERISTICS (V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = GND, T_A = 0° to 70°C unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Port A with CMOS Drive Enabled					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	–	–	0.4	V
Output High Voltage, I_{Load} = – 100 μA	V_{OH}	2.4	–	–	V
Output High Voltage, I_{Load} = – 10 μA	V_{OH}	V_{CC}+1	–	–	V
Input High Voltage, I_{Load} = –300 μA (max)	V_{IH}	2.0	–	V_{CC}	V
Input Low Voltage, I_{Load} = –500 μA (max)	V_{IL}	V_{SS}	–	0.8	V
Hi-Z State Input Current (V_{in} = 2.0 V to V_{CC})	I_{IH}	–	–	– 300	μA
Hi-Z State Input Current (V_{in} = 0.4 V)	I_{IL}	–	–	– 500	μA
Port B					
Output Low Voltage, I_{Load} = 3.2 mA	V_{OL}	–	–	0.4	V
Output Low Voltage, I_{Load} = 10 mA (sink)	V_{OL}	–	–	1.0	V
Output High Voltage, I_{Load} = – 200 μA	V_{OH}	2.4	–	–	V
Darlington Current Drive (Source), V_O = 1.5 V	I_{OH}	– 1.0	–	– 10	mA
Input High Voltage	V_{IH}	2.0	–	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	–	0.8	V
Hi-Z State Input Current	I_{TSI}	–	2	10	μA
Port C and Port A with CMOS Drive Disabled					
Output Low Voltage, I_{Load} = 1.6 mA	V_{OL}	–	–	0.4	V
Output High Voltage, I_{Load} = – 100 μA	V_{OH}	2.4	–	–	V
Input High Voltage	V_{IH}	2.0	–	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	–	0.8	V
Hi-Z State Input Current	I_{TSI}	–	2	10	μA

FIGURE 2 — TTL EQUIVALENT TEST LOAD (PORT B)

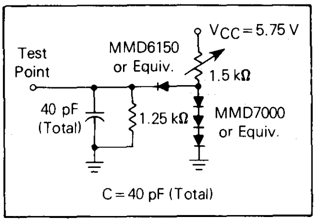


FIGURE 3 — CMOS EQUIVALENT TEST LOAD (PORT A)

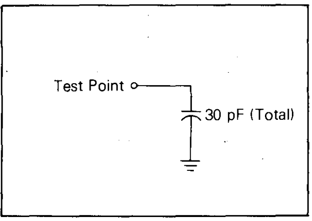
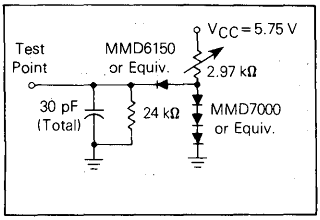


FIGURE 4 — TTL EQUIVALENT TEST LOAD (PORTS A AND C)



SIGNAL DESCRIPTION

The input and output signals for the MCU are described in the following paragraphs.

V_{CC} AND V_{SS}

Power is supplied to the MCU using these two pins. V_{CC} is power and V_{SS} is the ground connection.

$\overline{\text{INT}}$

This pin provides the capability for asynchronously applying an external interrupt to the MCU. Refer to **INTERRUPTS** for additional information.

XTAL AND EXTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal or an external signal can be connected to these pins to provide input to the internal oscillator. Lead length and stray capacitance on these two pins should be minimized. Refer to **INTERNAL OSCILLATOR** for recommendations about these pins.

f_{in}

This pin provides the high frequency digital input to the variable divider portion of the on-chip frequency synthesizer. The reference frequency for the phase lock loop is divided down from the crystal oscillator. Refer to the **PHASE LOCK LOOP** for details on the frequency synthesizer features.

ϕCOMP

This three-state output is the result of comparing the internal reference frequency to the variable divider signal. Refer to **PHASE LOCK LOOP** for details. In self-check, ϕCOMP is raised to ≈9 Volts.

$\overline{\text{RESET}}$

A low voltage level on this Schmitt trigger input will reset the MPU. Refer to **RESETS** for additional information.

NUM

This pin is not for user application and must be connected to V_{SS}.

INPUT/OUTPUT LINES (PA0-PA7, PB0-PB7, PC0-PC2)

These 19 lines are arranged into two 8-bit ports (A and B) and one 3-bit port (C). All lines are programmable as either inputs or outputs under software control of the data direction registers. Refer to **INPUT/OUTPUTS** for additional information. The PC0/TIMER pin also serves as an external input to the internal timer. Refer to the **TIMER** section for information on the timer modes.

MEMORY

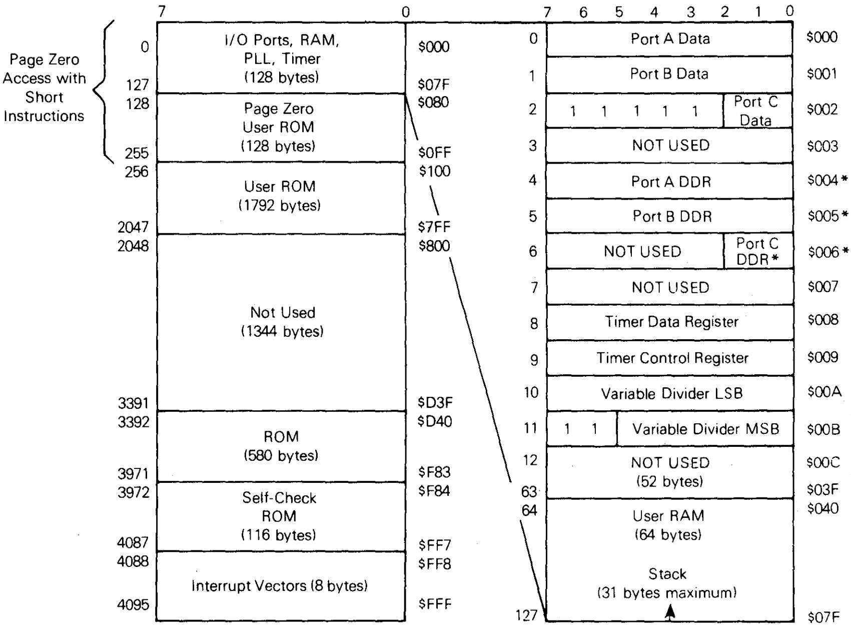
The MCU memory is configured as shown in Figure 5. The MCU is capable of addressing 4096 bytes of memory and I/O registers with its program counter. The MCU has implemented 2698 of these memory locations. This consists of: 2508 bytes user ROM, 116 bytes self-check ROM, 64 bytes of user RAM, six bytes of port I/O, two timer registers, and two PLL registers. The user ROM is split into two areas. The first area begins at memory location $080 and continues through $7FF. The lower 128 bytes of this ROM area (part of page zero) allows the user to access ROM locations utilizing the direct and table look-up indexed adressing modes. The second user ROM area begins at memory location $D40 and continues through $F83. The last eight user ROM locations, at the top of memory, are for the interrupt vectors.

The MCU reserves the first 16 memory locations for I/O features, of which 10 have been implemented. These locations are used for the ports, the port DDRs, the timer, and the PLL registers.

Sixty-four bytes of user RAM are provided. Of the 64 bytes, 31 bytes are shared with the stack area. The stack must be used with care when data shares the stack area.

The shared stack area is used during the processing of interrupt and subroutine calls to save the processor state. The register contents are pushed onto the stack in the order shown in Figure 6. Since the stack pointer decrements during pushes, the low order byte (PCL) of the program counter is stacked first; then the high order four bits (PCH) are stacked. This ensures that the program counter is loaded correctly following pulls from the stack, since the stack pointer increments when it pulls data from the stack. A subroutine call results in only the program counter (PCH, PCL) contents being pushed onto the stack, the remaining CPU registers are not pushed.

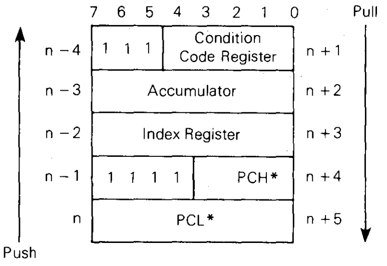
FIGURE 5 — MC6805T2 MCU ADDRESS MAP



Page Zero Access with Short Instructions
7
0
0
127
128
255
256
2047
2048
3391
3392
3971
3972
4087
4088
4095
I/O Ports, RAM, PLL, Timer (128 bytes)
Page Zero User ROM (128 bytes)
User ROM (1792 bytes)
Not Used (1344 bytes)
ROM (580 bytes)
Self-Check ROM (116 bytes)
Interrupt Vectors (8 bytes)
$000
$07F
$080
$0FF
$100
$7FF
$800
$D3F
$D40
$F83
$F84
$FF7
$FF8
$FFF
7 6 5 4 3 2 1 0
0 Port A Data $000
1 Port B Data $001
2 1 1 1 1 1 Port C Data $002
3 NOT USED $003
4 Port A DDR $004*
5 Port B DDR $005*
6 NOT USED Port C DDR* $006*
7 NOT USED $007
8 Timer Data Register $008
9 Timer Control Register $009
10 Variable Divider LSB $00A
11 1 1 Variable Divider MSB $00B
12 NOT USED (52 bytes) $00C
63 $03F
64 User RAM (64 bytes) $040
Stack (31 bytes maximum)
127 $07F

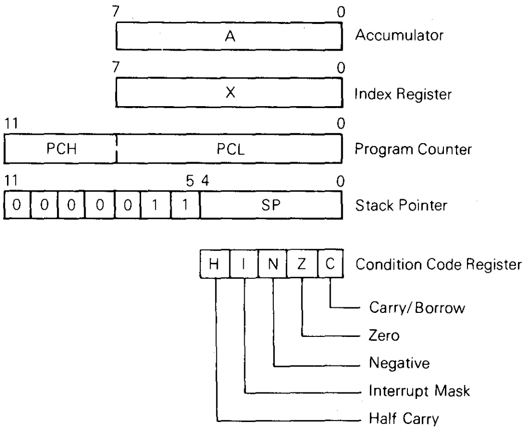
*Caution: Data direction registers (DDRs) are write-only; they read as $FF.

FIGURE 6 — INTERRUPT STACKING ORDER



*For subroutine calls, only PCH and PCL are stacked.

FIGURE 7 — PROGRAMMING MODEL



REGISTERS

The MCU has five registers available to the programmer. They are shown in Figure 7 and are explained in the following paragraphs.

ACCUMULATOR (A)

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

INDEX REGISTER (X)

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit value that may be added to an instruction value to create an effective address. The index register can also be used for data manipulations using the read/modify/write instructions. The index register may also be used as a temporary storage area.

PROGRAM COUNTER (PC)

The program counter is a 12-bit register that contains the address of the *NEXT* instruction to be executed.

STACK POINTER (SP)

The stack pointer is a 12-bit register that contains the address of the next free location on the stack. Initially, the stack pointer is set to location $07F and is decremented as data is being pushed onto the stack and incremented as data is being pulled from the stack. The seven most-significant bits of the stack pointer are permanently set to 0000011. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set to location $07F. Subroutines and interrupts may be nested down to location $061 (31 bytes maximum) which allows the programmer to use up to 15 levels of subroutine calls. 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 in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program and specific action taken as a result of their state. Each individual condition code register bit is explained in the following paragraphs.

HALF CARRY (H) — Set during ADD and ADC instructions to indicate that a carry occurred between bits 3 and 4.

INTERRUPT (I) — This bit is set to mask (disable) the timer and external interrupt ($\overline{\text{INT}}$). If an interrupt occurs while this bit is set, the interrupt is latched and is processed as soon as the interrupt bit is cleared.

NEGATIVE (N) — Used to indicate that the result of the *LAST* arithmetic, logical, or data manipulation was negative (bit 7 in result equal to a logic one).

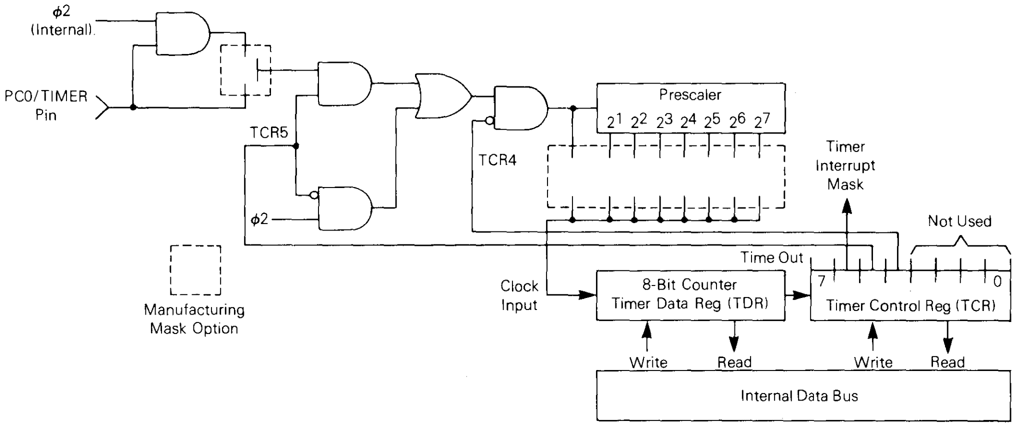
ZERO (Z) — Used to indicate that the result of the *LAST* arithmetic, logical, or data manipulation was zero.

CARRY/BORROW (C) — Used to indicate 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 plus shifts and rotates.

TIMER

The MC6805T2 timer circuitry is shown in Figure 8. The 8-bit counter may be loaded under program control and is decremented toward zero by the clock input (prescaler output). When the timer reaches zero the timer interrupt request

FIGURE 8 — TIMER BLOCK DIAGRAM



bit (bit 7) in the timer control register (TCR) is set. The timer interrupt can be masked (disabled) by setting the timer interrupt mask bit (bit 6) in the TCR. The interrupt bit (I bit) in the condition code register also prevents a timer interrupt from being processed. The MCU responds to this interrupt by saving the present CPU state on the stack, fetching the timer interrupt vector from locations $FF8 and $FF9 and executing the interrupt routine; see **INTERRUPTS**. *The timer interrupt request bit (TCR7) must be cleared by software.*

The clock input to the timer is established via bit 5 (TCR5) in the timer control register. When this bit is set (external mode), the timer clock source is the PC0/TIMER pin. In this mode a mask option is used to select either the ϕ2 gated with PC0/TIMER or the positive transition on PC0/TIMER as timer clock source. This allows easily performed pulse width or pulse count measurements. When TCR5 is low, logic zero, the timer clock source is the internal ϕ2.

Bit 4 in the timer control register (TCR4) disables the timer clock source when set to logic one.

The maximum frequency of a signal that can be recognized by the PC0/TIMER pin logic is dependent on the parameter labeled t_{WL}, t_{WH}. The pin logic that recognizes the high (or low) state on the pin must also recognize the low (or high) state on the pin in order to "re-arm" the internal logic. Therefore, the period can be calculated as follows: (assumes 50/50 duty cycle for a given period).

$$t_{cyc} \times 2 + 250 \text{ ns} = \text{period} = \frac{1}{\text{freq}}$$

The period is not simply $t_{WL} + t_{WH}$. This computation is allowable, but it does reduce the maximum allowable frequency by defining an unnecessarily long period (250 nanoseconds twice).

When the ϕ2 signal is used as the source, it can be gated by an input applied to the PC0/TIMER input pin allowing the user to easily perform pulse-width measurements. (Note: for ungated ϕ2 clock inputs to the timer prescaler, the PC0/TIMER pin should be tied to V_{CC}.) The source of the clock input is one of the mask options that is specified before manufacture of the MC6805T2.

A prescaler option, divide by 2^n, can be applied to the clock input that extends the timing interval up to a maximum of 128 counts before decrementing the counter. This prescaling mask option is also specified before manufacture.

The timer continues to count past zero, falling through to $FF from zero, and then continuing to count. Thus, the counter can be read at any time by reading the timer data register (TDR). This allows a program to determine the length of time since a timer interrupt has occurred, and not disturb the counting process.

At power-up or reset, the prescaler and counter are initialized with all logic ones; the timer interrupt request bit (TCR7) is cleared; the timer interrupt mask bit (TCR6) is set; the external timer source bit (TCR5) is cleared and the timer disable bit (TCR4) is cleared.

SELF-CHECK

The self-check capability of the MCU provides an internal check to determine if the part is functional. Connect the MCU as shown in Figure 9 and monitor the output of Port C bit 3 for an oscillation of approximately 3 hertz. A 9-volt level on the ϕCOMP input, pin 7, energizes the ROM-based self-check feature. The self-check program exercises the RAM, ROM, timer, interrupts, and I/O ports.

RESETS

The MCU is reset whenever the $\overline{\text{RESET}}$ input line senses a logic zero. This can be accomplished in two different ways: (1) during power-up when a capacitor is used to hold the $\overline{\text{RESET}}$ pin low for a specified time (t_{RHL}); and (2) any time after power-up that the $\overline{\text{RESET}}$ line falls to a logic zero for a period longer than one t_{cyc}. See Figures 10 and 11.

The internal circuit connected to the $\overline{\text{RESET}}$ pin consists of a Schmitt trigger which senses the $\overline{\text{RESET}}$ line logic level. The Schmitt trigger provides an internal reset voltage if it senses a logic 0 on the $\overline{\text{RESET}}$ pin. During power-up, the Schmitt trigger switches on (removes reset) when the $\overline{\text{RESET}}$ pin voltage rises to V_{IRES+}. When the $\overline{\text{RESET}}$ pin voltage falls to a logic 0 for a period longer than one t_{cyc}, the Schmitt trigger switches off to provide an internal reset voltage. The "switch off" voltage occurs at V_{IRES-}. A typical reset Schmitt trigger hysteresis curve is shown in Figure 11(b).

Upon power-up, a delay of t_{RHL} is needed before allowing the reset input to go high. This time allows the internal clock generator to stabilize. Connecting a capacitor to the $\overline{\text{RESET}}$ input, as shown in Figure 11(a), will provide sufficient delay. See Figure 15 under **INTERRUPTS** for the complete reset sequence.

INTERNAL OSCILLATOR

The internal oscillator circuit has been designed to require a minimum of external components. The use of a crystal or an external signal may be used to drive the internal oscillator. The different connection methods are shown in Figures 12 and 13. The crystal specifications and suggested PC board layout are given in Figure 14.

The crystal oscillator startup time is a function of many variables: crystal parameters (especially R_S), oscillator load capacitances, IC parameters, ambient temperature, and supply voltage. To ensure rapid oscillator startup, neither the crystal characteristics nor the load capacitance should exceed recommendations.

INTERRUPTS

The MC6805T2 MCU can be interrupted three different ways: through the external interrupt ($\overline{\text{INT}}$) input pin, the internal timer interrupt request, or the software interrupt instruction (SWI). When any interrupt occurs, processing is suspended, the present CPU state is pushed onto the stack, the interrupt bit (I) in the condition code register is set, the address of the interrupt routine is obtained from the appropriate interrupt vector address, and the interrupt routine is executed. Stacking the CPU registers, setting the I bit, and vector fetching requires a total of 11 t_{cyc} periods for completion.

A flowchart of the interrupt sequence is shown in Figure 15. The interrupt service routine must end with a return from interrupt instruction (RTI) which allows the MCU to resume processing of the program prior to the interrupt (by unstacking the previous CPU state).

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

As shown in Figure 15, when execution of the current instruction is complete, the processor checks all pending hardware interrupts and if unmasked (I bit = 0), proceeds with interrupt processing; otherwise, the next instruction is fetched and executed. Masked interupts are latched internally for later interrupt service.

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 (software interrupt) instruction is executed as any other instruction and as such will take precedence over hardware interrupts only if the I bit is set (interrupts masked).

Table 1 shows the execution priority of the $\overline{\text{RESET}}$, $\overline{\text{INT}}$ and timer interrupts, and instructions (including the software interrupt, SWI). Two conditions are shown, one with the I bit set and the other with 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 1(a), the second highest priority is assigned to any instruction including SWI. This is illustrated in Figure 15 which shows that the $\overline{\text{INT}}$ or timer interrupts are not tested when the I bit is set. If the I bit is cleared as per Table 1(b), the priorities change in that the next instruction (including SWI) is not fetched until after the $\overline{\text{INT}}$ and timer interrupts have been tested (and serviced). Also, when the I bit is clear, if both $\overline{\text{INT}}$ and timer interrupts are pending, the $\overline{\text{INT}}$ interrupt is always serviced before the timer interrupt.

The external interrupt is internally synchronized and will set a latch on the falling edge of $\overline{\text{INT}}$. A sinusoidal input signal (f_{INT} maximum) can be used to generate an external interrupt, as shown in Figure 16(a) for use as a zero-crossing detector with hysteresis included. This allows for applications such as time-of-day routines and engaging/disengaging ac power control devices. As shown in Figure 17(a), off-chip clamping limits the ac input to the V_{INT} specification while still providing an interrupt at every zero crossing of the ac signal.

For digital applications, as shown in Figure 16(b), the $\overline{\text{INT}}$ input can be driven directly by a digital signal. The maximum frequency of a signal that can be recognized by the $\overline{\text{INT}}$ pin logic is dependent on the parameter labeled t_{WL}, t_{WH}. The logic that recognizes the high (or low) state on the pin must also recognize the low (or high) state on the pin in order to "re-arm" the internal logic. Therefore, the period can be calculated as follows: (assumes 50/50 duty cycle for a given period).

$$t_{cyc} \times 2 + 250\ \text{ns} = \text{period} = \frac{1}{\text{freq}}$$

The period is not simply $t_{WL} + t_{WH}$. This computation is allowable, but it does reduce the maximum allowable frequency by defining an unnecessarily longer period (250 nanoseconds twice). For the $\overline{\text{INT}}$ function, the maximum allowable frequency is also determined by the software response of the INT service routine.

A software interrupt (SWI) is an executable instruction which is executed regardless of the state of the I bit in the condition code register; however, if the I bit is clear, a software interrupt (or any other instruction) cannot be executed until after the $\overline{\text{INT}}$ and timer interrupts have been serviced. SWIs are usually used as breakpoints for debugging or as system calls.

FIGURE 9 — SELF-CHECK CONNECTIONS

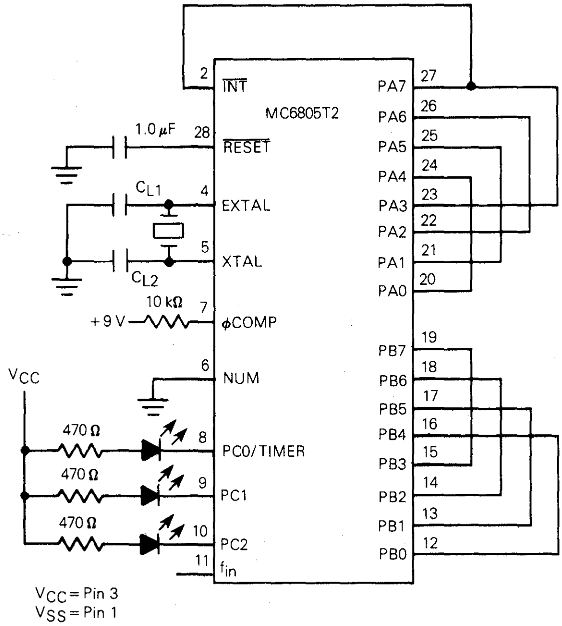


FIGURE 10 — POWER AND RESET TIMING

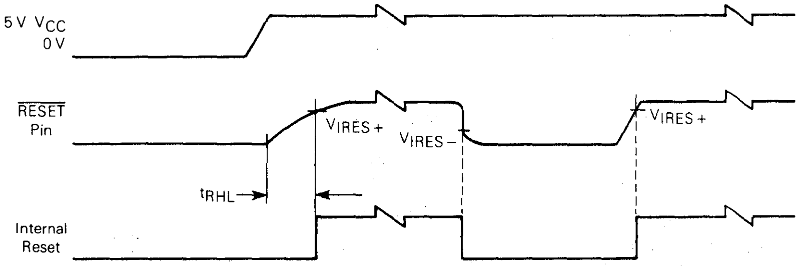


FIGURE 11 — POWER-UP RESET DELAY CIRCUIT

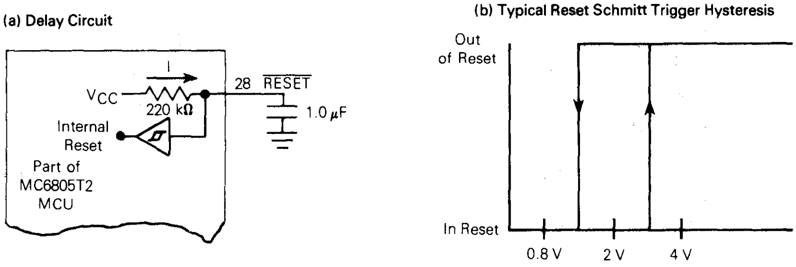


FIGURE 12 — CRYSTAL OSCILLATOR

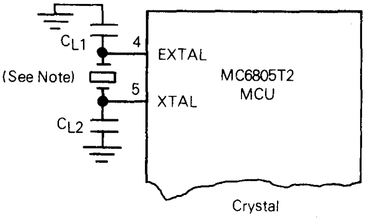
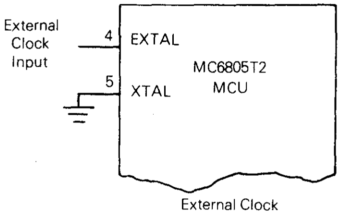


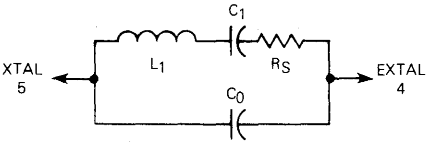
FIGURE 13 — EXTERNAL OSCILLATOR



NOTE: The recommended value of both C_{L1} and C_{L2} with a 4.0 megahertz crystal is 27 picofarads maximum, including system distributed capacitance. For crystal frequencies other than 4 megahertz, the total capacitance on each pin should be scaled as the inverse of the frequency ratio. For example, with a 2-megahertz crystal, use approximately 50 picofarads for both C_{L1} and C_{L2}. The exact value depends on the motional-arm parameters of the crystal used.

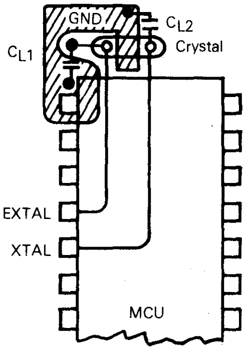
FIGURE 14 — CRYSTAL MOTIONAL ARM PARAMETERS AND SUGGESTED PC BOARD LAYOUT

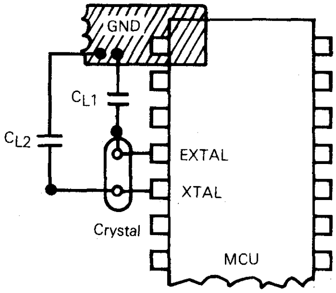
(a) Recommended Crystal Motional-Arm Parameters



AT — Cut Parallel Resonance Crystal
$C_0 = 7$ pF Max.
FREQ = 4.0 MHz @ C_{L1} and $C_{L2} = 24$ pF
$R_S = 50$ ohms Max.

(b) Suggested PC Board Layouts





NOTE: Keep crystal leads and circuit connections as short as possible.

FIGURE 15 — RESET AND INTERRUPT PROCESSING FLOWCHART

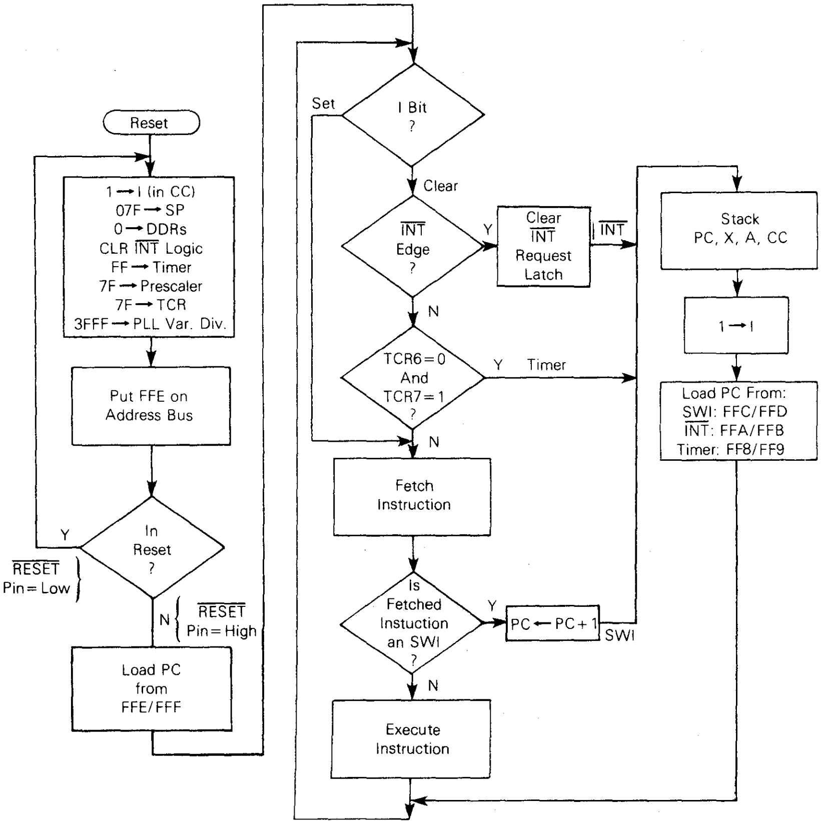


TABLE 1 — INTERRUPT/INSTRUCTION EXECUTION PRIORITY AND VECTOR ADDRESS

(a) I Bit Set

Interrupt/Instruction	Priority	Vector Address
RESET	1	$FFE-$FFF
SWI (or Other Instruction)	2	$FFC-$FFD

NOTE: INT and Timer Interrupts are not tested when the I bit is set; therefore, they are not shown.

(b) I Bit Clear

Interrupt/Instruction	Priority	Vector Address
RESET	1	$FFE-$FFF
INT	2	$FFA-$FFB
Timer	3	$FF8-$FF9
SWI (or Other Instruction)	4	$FFC-$FFD

INPUT/OUTPUT

There are 19 input/output pins. (The INT pin may also be polled with branch instructions to provide an additional input pin.) All pins (Ports A, B, and C) are programmable as either inputs or outputs under software control of the corresponding data direction registers (DDR). The port I/O programming is accomplished by writing the corresponding bit in the port DDR to a logic 1 for output or a logic 0 for input. On reset all DDRs are initialized to a logic 0 state to put the ports in the input mode. The port output registers are not initialized on reset and should be initialized before changing the DDR bits to avoid undefined levels. When programmed as outputs, all I/O pins read the latched output data, regardless of the logic levels at the output pin due to output loading; see Figure 17. When Port B is programmed for outputs, it is capable of sinking 10 milliamperes and sourcing one milliampere on each pin.

FIGURE 16 — TYPICAL INTERRUPT CIRCUITS

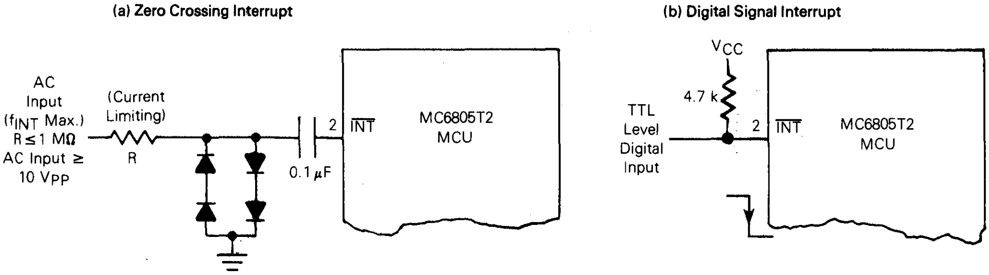
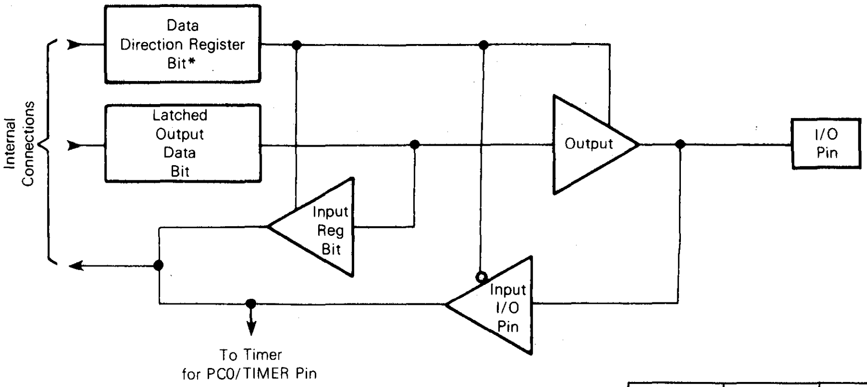


FIGURE 17 — TYPICAL PORT I/O CIRCUITRY



Data Direction Register Bit	Output Data Bit	Output State	Input To MCU
1	0	0	0
1	1	1	1
0	X	3-State**	Pin

*DDR is a write-only register and reads as all 1s.

**Ports A (with CMOS drive disabled), B, and C are three-state ports. Port A has optional internal pullup devices to provide CMOS drive capability. See Electrical Characteristics tables for complete information.

All input/output lines are TTL compatible as both inputs and outputs. Ports B and C are CMOS compatible as inputs. Port A may be made CMOS compatible as outputs with a mask option. Figure 18 provides some examples of port connections. The address map in Figure 5 gives the address of the data registers and DDRs. The register configuration is shown in Table 2.

Caution

The corresponding DDRs for ports A, B, and C are write-only registers (registers at $004, $005, and $006). 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). It is recommended that all DDR bits in a port be written using a single-store instruction.

The latched output data bit (see Figure 17) may always be written. Therefore, any write to a port writes all of its data bits even though the port DDR is set to input. This may be used to initialize the data registers and avoid undefined outputs; however, care must be exercised when using read/modify/write instructions since the data read corresponds to the pin level if the DDR is an input (0) and corresponds to the latched output data when the DDR is an output (1).

PHASE LOCK LOOP

The PLL section consists of: a 14-bit binary variable divider, a fixed 10-stage divider, a digital phase and frequency comparator with a three-state output, and circuitry to avoid "backlash" effects in phase lock conditions.

With a suitable high-frequency prescaler and an active integrator the user can easily establish a frequency synthesizer system driving a voltage controlled oscillator, as shown in Figure 19. The equations governing the PLL are given in Figure 20.

TABLE 2 — MCU REGISTER CONFIGURATION

PORT DATA REGISTER

Port A Addr = $000
Port B Addr = $001
Port C Addr = $002

7	0

PORT DATA DIRECTION REGISTER (DDR)

(1) Write Only; reads as all 1s
(2) 1 = Output, 0 = Input. Cleared to 0 by Reset.
(3) Port A Addr = $004
Port B Addr = $005
Port C Addr = $006

7	0

TIMER DATA REGISTER (TDR)

$008	7	0
	MSB	LSB

TIMER CONTROL REGISTER (TCR)

$009	7	6	5	4	3	2	1	0
					1	1	1	1

TCR Bits 0, 1, 2, and 3 read as 1s (not used).

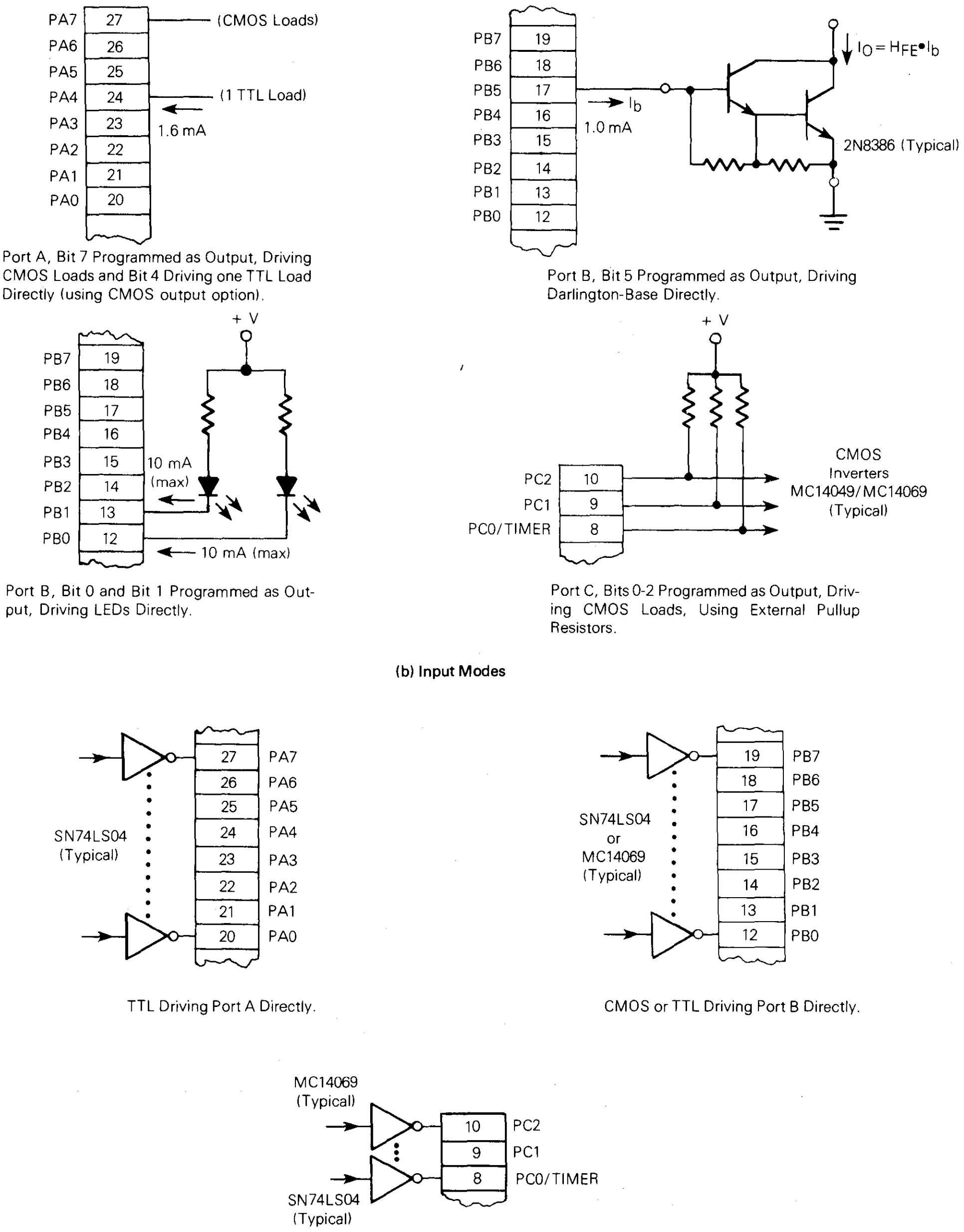
TCR4 — Disable Timer: 1 = Timer Stopped. 0 = Timer Allowed to Count. Cleared to 0 by Reset.

TCR5 — External Timer Source: 1 = External, 0 = Internal $\phi 2$. Cleared to 0 by Reset.

TCR6 — Timer Interrupt Mask Bit: 1 = timer interrupt masked (disabled). Set to 1 by Reset.

TCR7 — Timer Interrupt Request Status Bit: Set when TDR goes to zero, must be cleared by software. Cleared to 0 by Reset.

FIGURE 18 — TYPICAL PORT CONNECTIONS



(a) Output Modes

Port A, Bit 7 Programmed as Output, Driving CMOS Loads and Bit 4 Driving one TTL Load Directly (using CMOS output option).

Port B, Bit 5 Programmed as Output, Driving Darlington-Base Directly.

Port B, Bit 0 and Bit 1 Programmed as Output, Driving LEDs Directly.

Port C, Bits 0-2 Programmed as Output, Driving CMOS Loads, Using External Pullup Resistors.

(b) Input Modes

TTL Driving Port A Directly.

CMOS or TTL Driving Port B Directly.

CMOS and TTL Driving Port C Directly.

VARIABLE DIVIDER

The variable divider is a 14-bit binary down counter which communicates with the CPU via two read/write registers located at address \$00A, for the LS byte, and \$00B, for the MS byte. The upper two bits in register \$00B, always read as logic 1s. When the variable divider count has reached zero a preset pulse, f_{VAR}, is generated.

The f_{VAR} signal is applied to the phase comparator circuit together with the f_{REF} signal. The phase/frequency difference, between the two signals, results in an error signal output (ϕCOMP, pin 7) which controls the VCO frequency. In addition, the f_{VAR} pulse is also used to reload the 14-bit variable divider latch as shown in Figure 21.

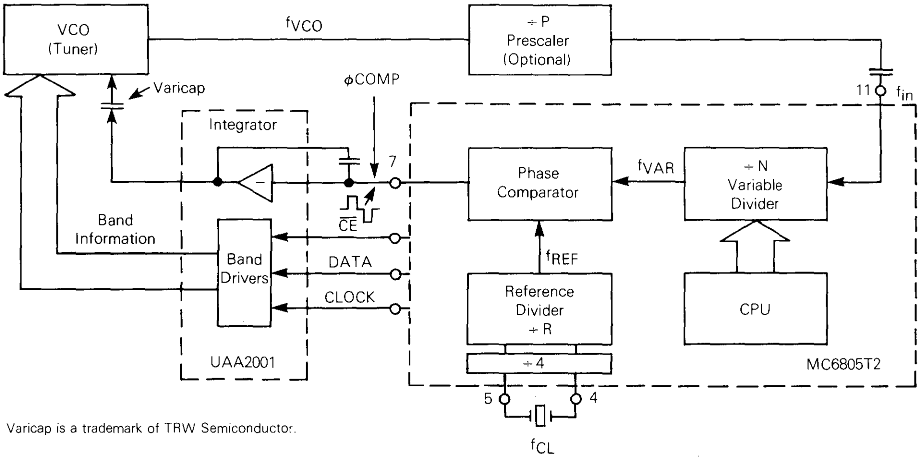
Data transfers from registers \$00A and \$00B to the latch occur outside the preset time and only during a write operation performed on register \$00A. For example; a 6-bit data transfer to register \$00B is only transferred to the variable divider if followed by a write operation to register \$00A. For correct operation of the variable divider, the absolute value of the four lower significant bits of the 14-bit binary code (loaded into the 14-bit latch) must be less than or equal to the absolute value of the upper 10 bits. Figure 22 shows a typical manipulation of the 14-bit data to the registers.

The use of the 14-bit latch synchronizes the data transfer between two asynchronous systems, namely, the CPU and the variable divider.

At power-up reset both the variable divider and the contents of the PLL registers are set to logic 1s.

The variable frequency input pin, f_{in}, is self biased requiring an ac coupled signal with a minimum swing of 0.5 V. The input frequency range of f_{in} allows the device, together with a suitable prescaler, to cover a given frequency spectrum. For example, with a ÷64 prescaler the entire television frequency spectrum can be covered.

FIGURE 19 — PHASE LOCK LOOP AN AN RF FREQUENCY SYNTHESIZER



Varicap is a trademark of TRW Semiconductor.

FIGURE 20 — PRINCIPAL PLL EQUATIONS

For a system in lock:

$f_{VAR} = f_{REF}$

$f_{VAR} = f_{in} \div N$

$f_{in} = f_{VCO} \div P$

$F_{VCO} = f_{REF} \times P \times N$

Where: P = Prescaler division ratio
N = Variable Divider division ratio

Minimum frequency step =

$$\frac{\Delta f_{VCO}}{\Delta N} = f_{REF} \times P$$

Example for determining minimum frequency step:

$f_{CL} = 4.00$ MHz = Crystal frequency

$f_{CL/4} = 1.00$ MHz = Internal Oscillator frequency

$R = 2^{10}$ = Reference Divider ratio

$P = 64$ = Prescaler division ratio

$$f_{REF} = \frac{f_{CL}}{4 \times R} = \frac{4 \times 10^6}{4 \times 1024} = 976.5 \text{ Hz}$$

$$\frac{\Delta f_{VCO}}{\Delta N} = 976.5 \times 64 = 62.5 \text{ kHz}$$

FIGURE 21 — MC6805T2 PLL BLOCK DIAGRAM

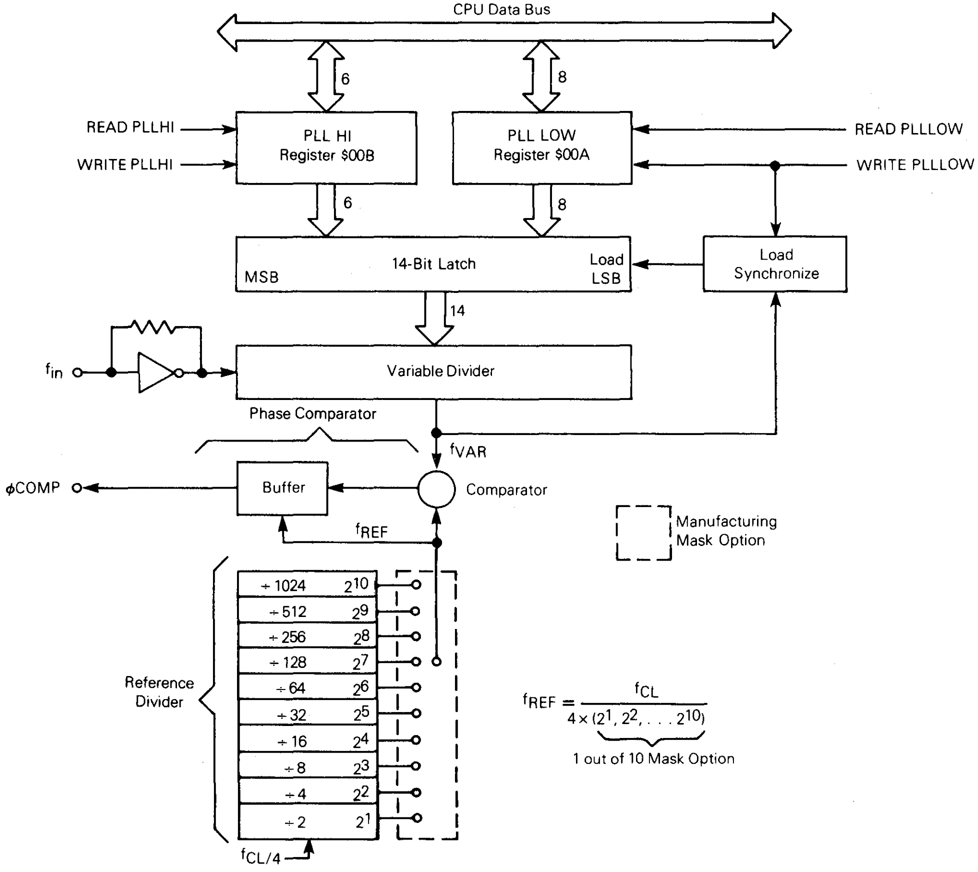


FIGURE 22 — TYPICAL FINE TUNE EXAMPLE

```
FTPLU    LDA     PLLLOW
         INCA            check if LS byte=$FF (Reg $00A)
         BNE     TT1     if not increment only LS byte
         INC     PLLHI   increment MSB (Reg $00B) before LSB
TT1      INC     PLLLOW
         •
         •
         •
FTMIN    TST     PLLLOW  check if LS byte=$00
         BNE     TT2     if not decrement only LS byte
         DEC     PLLHI   decrement MSB before LSB
TT2      DEC     PLLLOW
         •
         •
         •
```

REFERENCE DIVIDER

This 10-stage binary counter generates a reference frequency, f_{REF}, which is compared to the output of the variable divider. The reference divider is mask programable, thus, allowing the user a choice of reference frequency, see Figure 21.

PHASE COMPARATOR

The phase comparator compares the frequency and phase of f_{VAR} and f_{REF}, and according to their phase relationship generates a three-level output (1, 0, Hi-Z), φCOMP, as shown in Figures 23 and 24. The output waveform is then integrated, amplified, and the resultant dc voltage is applied to the voltage controlled oscillator varicap.

In practice a linear characteristic around the steady-state region can not be achieved due to internal propagation delays. Thus, phase comparators exhibit non-linear characteristics and for systems which lock in phase, this results in a "backlash" effect – creating sidebands and FM distortion. To avoid this effect a very short pulse is injected periodically into the system. The loop, in turn, attempts to cancel this interference and in doing so brings the phase comparator to its linear zero, as shown in Figures 25 and 26.

A typical application, for a TV frequency synthesizer, is illustrated in Figure 27.

FIGURE 23 – PHASE COMPARATOR STATE DIAGRAM

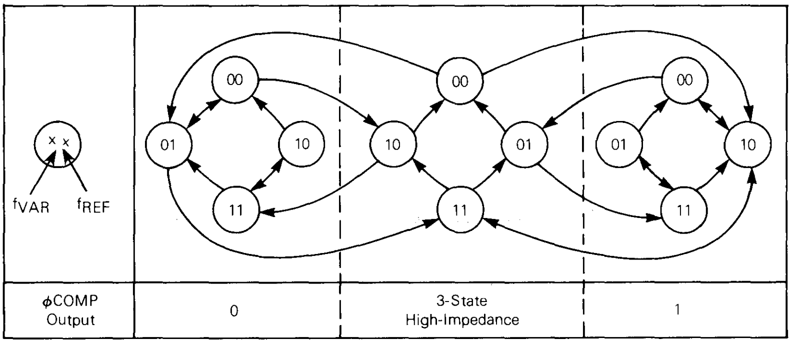


FIGURE 24 – PHASE COMPARATOR OUTPUT WAVEFORM

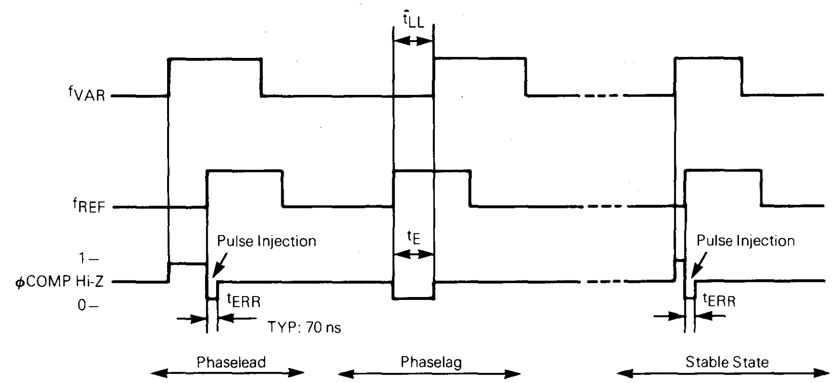


FIGURE 25 — PHASE COMPARATOR CHARACTERISTICS

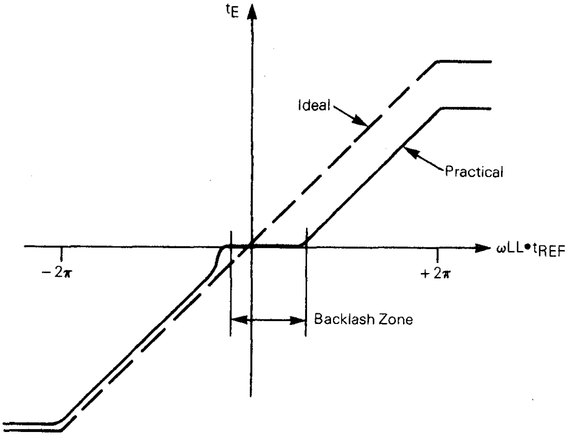


FIGURE 26 — PHASE COMPARATOR WITH PULSE INJECTION

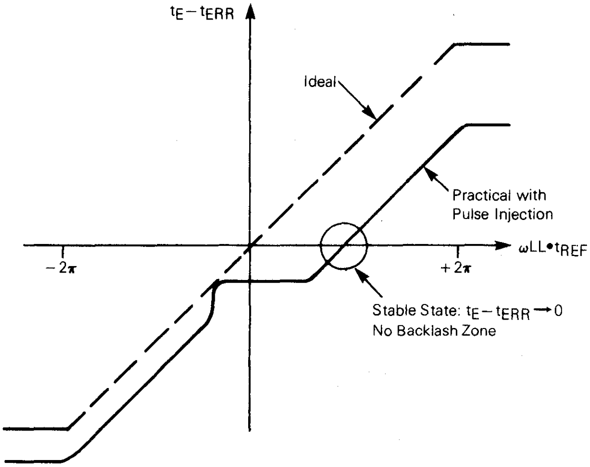
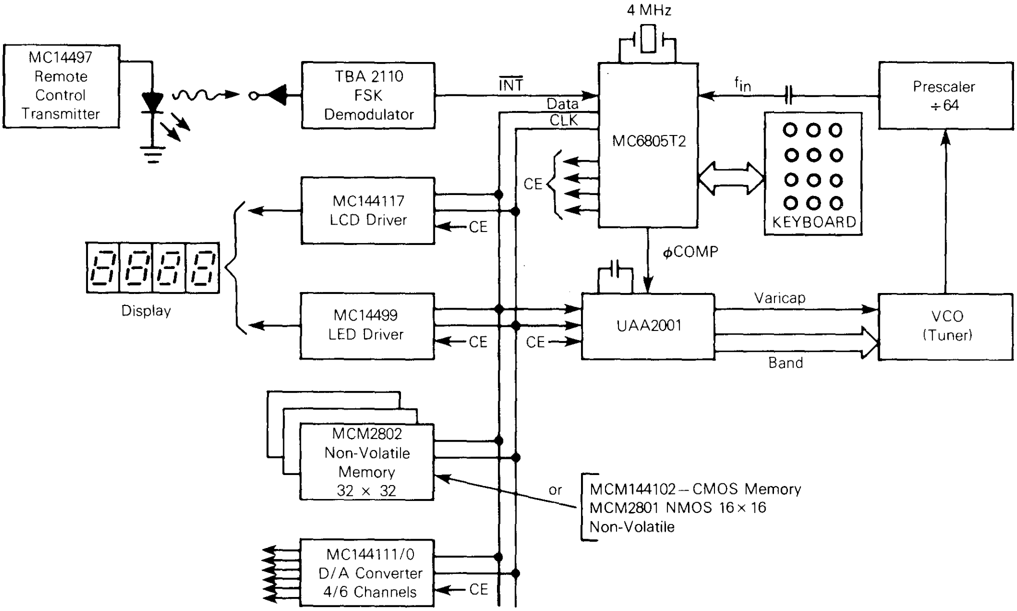


FIGURE 27 — A TYPICAL TV SYNTHESIZER APPLICATION

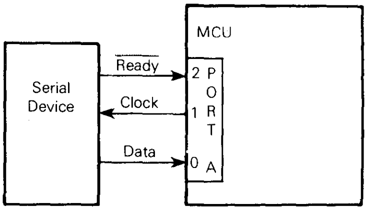


BIT MANIPULATION

The MCU has the ability to set or clear any single random-access memory or input/output bit (except the data direction registers, see Caution under **INPUT/OUTPUT**), with a single instruction (BSET, BCLR). Any bit in page zero including ROM, except the DDRs, can be tested, using the BRSET and BRCLR instructions, and the program branches as a result of its state. The carry bit equals the value of the bit referenced by BRSET and BRCLR. A rotate instruction may then be used to accumulate serial input data in a RAM location or register. The capability to work with any bit in RAM, ROM, or I/O allows the user to have individual flags in RAM or to handle I/O bits as control lines.

The coding example in Figure 28 illustrates the usefulness of the bit manipulation and test instructions. Assume that the MCU is to communicate with an external serial device. The external device has a data ready signal, a data output line, and a clock line to clock data one bit at a time, LSB first, out of the device. The MCU waits until the data is ready, clocks the external device, picks up the data in the carry flag (C bit), clears the clock line and finally accumulates the data bits in a RAM location.

FIGURE 28 — BIT MANIPULATION EXAMPLE



```
                      •
                      •
                      •
SELF      BRSET       2, PORTA, SELF
                      •
                      •
                      •
          BSET        1, PORTA
          BRCLR       0, PORTA, CONT
CONT      BCLR        1, PORTA
          ROR         RAMLOC
                      •
                      •
```

ADDRESSING MODES

The MC6805T2 has 10 addressing modes which are explained briefly in the following paragraphs. For additional details and graphical illustrations, refer to the M6805 Family Users Manual.

The term "effective address" (EA) is used in describing the address modes. EA 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 contained in the byte immediately following the opcode. The immediate addressing mode is used to access constants which do not change during program execution (e.g., 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 lowest 256 bytes in memory (page zero) with a single 2-byte instruction. The address area includes the on-chip RAM and I/O registers and 128 bytes of ROM. Direct addressing is an effective use of both memory and time.

EXTENDED

In the extended addressing mode, the effective address of the argument is contained in the two bytes following the opcode. Instructions using extended addressing are capable of referencing arguments anywhere in memory with a single 3-byte instruction. When using the Motoıola assembler, the programmer need not specify whether an instruction uses direct or extended addressing. The assembler automatically selects the shortest form of the instruction.

RELATIVE

The relative addressing mode 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 from − 126 to + 129 from the opcode address. The programmer need not worry about calculating the correct offset when using the Motorola assembler, since it calculates the proper offset and checks to see if it is within the span of the branch.

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 (page zero). These instructions are only one byte long. This mode is often used to move a pointer through a table or to hold the address of a frequently referenced RAM or I/O location.

INDEXED, 8-BIT OFFSET

In the indexed, 8-bit offset addressing mode the effective address is the sum of the contents of the unsigned 8-bit index register (X) and the unsigned byte following the opcode. This addressing mode is useful in selecting the kth element in an n element table. With this 2-byte instruction, k would typically be in X with the address of the beginning of the table in the instruction. As such, tables may begin anywhere within the first 256 addressable locations and could extend as far as location 510 ($1FE).

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 3-byte instruction allows tables to be anywhere in memory. As with direct and extended, the Motorola assembler determines the shortest form of indexed addressing.

BIT SET/CLEAR

In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode, and the byte following the opcode specifies the page-zero address of the byte in which the specified bit is to be set or cleared. Thus, any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single 2-byte instruction. See Caution under **INPUT/OUTPUT**.

BIT TEST AND BRANCH

The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit, which is to be tested, and condition (set or clear) is included in the opcode and the address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the value of the PC if the branch condition is true. This single 3-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. 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 bit of the Condition Code Register. See Caution under **INPUT/OUTPUT**.

INHERENT

In the inherent addressing mode, all the information necessary to execute the instruction is contained in the opcode. Operations specifying only the index register or accumulator, as well as control instruction with no other arguments, are included in this mode. These instructions are one byte long.

INSTRUCTION SET

The MCU has a set of 59 basic instructions, which when combined with the 10 addressing modes produce 207 usable opcodes. 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. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to Table 3.

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 (see Caution under **INPUT/OUTPUT**). The test for negative or zero (TST) instruc tions. is included in the read/modify/write instructions though it does not perform the write. Refer to Table 4.

BRANCH INSTRUCTIONS

The branch instructions cause a branch from the program when a certain condition is met. Refer to Table 5.

BIT MANIPULATION INSTRUCTIONS

These instructions are used on any bit in the first 256 bytes of the memory (see Caution under **INPUT/OUTPUT**). One group either sets or clears. The other group performs the bit test branch operations. Refer to Table 6.

CONTROL INSTRUCTIONS

The control instructions control the MCU operations during program execution. Refer to Table 7.

ALPHABETICAL LISTING

The complete instruction set is given in alphabetical order in Table 8.

OPCODE MAP SUMMARY

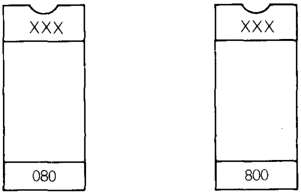
Table 9 is an opcode map for the instructions used on the MCU.

ORDERING INFORMATION

The information required when ordering a custom MCU is listed below. The ROM program may be transmitted to Motorola on EPROM(s) or on MDOS disk file.

To initiate a ROM pattern for the MCU it is necessary to first contact your local Motorola representative or distributor.

EPROMs — Two MCM2716 or one MCM2532 type EPROM(s), programmed with the customer program (positive logic sense for address and data), may be submitted for pattern generation. The EPROM(s) must be clearly marked to indicate which EPROM(s) corresponds to which address space. The recommended marking procedure for two MCM2716 EPROMs is illustrated below.



XXX = Customer ID

After the EPROM(s) are marked they should be placed in conductive IC carriers and securely packed. Do not use styrofoam.

VERIFICATION MEDIA

All original pattern media (EPROMs or Floppy Disk) 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 completed, signed, and returned to Motorola. The signed verification form constitutes the contractual agreement for creation of the customer mask. If desired, Motorola will program one blank EPROM from the data file used to create the custom mask to aid in the verification process.

ROM VERIFICATION UNITS (RVUs)

Ten MCUs containing the customer's 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 they are usually unmarked, packaged in ceramic, and tested only at room temperature and 5 volts. These RVUs are included in the mask charge and are not production parts. The RVUs thus are not guaranteed by Motorola Quality Assurance, and should be discarded after verification is completed.

FLEXIBLE DISKS

The disk media submitted must be single-sided, single-density, 8-inch, MDOS- compatible floppies. The customer must write the binary file name and company name on the disk with a felt-tip pen. The minimum MDOS system files as well as the absolute binary object file (filename, .LO type of file) from the M6805 cross assembler must be on the disk. An object file made from a memory dump using the ROLLOUT command is also acceptable. Consider submitting a source listing as well as the following files: filename, .LX(EXORciser loadable format) and filename, .SA (ASCII Source Code). These files will of course be kept confidential and are used 1) to speed up the process in-house if any problems arise, and 2) to speed up the user-to-factory interface if the user finds any software errors and needs assistance quickly from Motorola factory representatives.

MDOS is Motorola's Disk Operating System available on development systems such as EXORciser, EXORset, etc.

TABLE 3 — 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	4	C6	3	5	F6	1	4	E6	2	5	D6	3	6
Load X from Memory	LDX	AE	2	2	BE	2	4	CE	3	5	FE	1	4	EE	2	5	DE	3	6
Store A in Memory	STA	—	—	—	B7	2	5	C7	3	6	F7	1	5	E7	2	6	D7	3	7
Store X in Memory	STX	—	—	—	BF	2	5	CF	3	6	FF	1	5	EF	2	6	DF	3	7
Add Memory to A	ADD	AB	2	2	BB	2	4	CB	3	5	FB	1	4	EB	2	5	DB	3	6
Add Memory and Carry to A	ADC	A9	2	2	B9	2	4	C9	3	5	F9	1	4	E9	2	5	D9	3	6
Subtract Memory	SUB	A0	2	2	B0	2	4	C0	3	5	F0	1	4	E0	2	5	D0	3	6
Subtract Memory from A with Borrow	SBC	A2	2	2	B2	2	4	C2	3	5	F2	1	4	E2	2	5	D2	3	6
AND Memory to A	AND	A4	2	2	B4	2	4	C4	3	5	F4	1	4	E4	2	5	D4	3	6
OR Memory with A	ORA	AA	2	2	BA	2	4	CA	3	5	FA	1	4	EA	2	5	DA	3	6
Exclusive OR Memory with A	EOR	A8	2	2	B8	2	4	C8	3	5	F8	1	4	E8	2	5	D8	3	6
Arithmetic Compare A with Memory	CMP	A1	2	2	B1	2	4	C1	3	5	F1	1	4	E1	2	5	D1	3	6
Arithmetic Compare X with Memory	CPX	A3	2	2	B3	2	4	C3	3	5	F3	1	4	E3	2	5	D3	3	6
Bit Test Memory with A (Logical Compare)	BIT	A5	2	2	B5	2	4	C5	3	5	F5	1	4	E5	2	5	D5	3	6
Jump Unconditional	JMP	—	—		BC	2	3	CC	3	4	FC	1	3	EC	2	4	DC	3	5
Jump to Subroutine	JSR	—	—		BD	2	7	CD	3	8	FD	1	7	ED	2	8	DD	3	9

TABLE 4 — 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	# Bytes	# Cycles
Increment	INC	4C	1	4	5C	1	4	3C	2	6	7C	1	6	6C	2	7
Decrement	DEC	4A	1	4	5A	1	4	3A	2	6	7A	1	6	6A	2	7
Clear	CLR	4F	1	4	5F	1	4	3F	2	6	7F	1	6	6F	2	7
Complement	COM	43	1	4	53	1	4	33	2	6	73	1	6	63	2	7
Negate (2's Complement)	NEG	40	1	4	50	1	4	30	2	6	70	1	6	60	2	7
Rotate Left Thru Carry	ROL	49	1	4	59	1	4	39	2	6	79	1	6	69	2	7
Rotate Right Thru Carry	ROR	46	1	4	56	1	4	36	2	6	76	1	6	66	2	7
Logical Shift Left	LSL	48	1	4	58	1	4	38	2	6	78	1	6	68	2	7
Logical Shift Right	LSR	44	1	4	54	1	4	34	2	6	74	1	6	64	2	7
Arithmetic Shift Right	ASR	47	1	4	57	1	4	37	2	6	77	1	6	67	2	7
Test for Negative or Zero	TST	4D	1	4	5D	1	4	3D	2	6	7D	1	6	6D	2	7

TABLE 5 — BRANCH INSTRUCTIONS

		Relative Addressing Mode		
Function	Mnemonic	Op Code	# Bytes	# Cycles
Branch Always	BRA	20	2	4
Branch Never	BRN	21	2	4
Branch IFF Higher	BHI	22	2	4
Branch IFF Lower or Same	BLS	23	2	4
Branch IFF Carry Clear	BCC	24	2	4
(Branch IFF Higher or Same)	(BHS)	24	2	4
Branch IFF Carry Set	BCS	25	2	4
(Branch IFF Lower)	(BLO)	25	2	4
Branch IFF Not Equal	BNE	26	2	4
Branch IFF Equal	BEQ	27	2	4
Branch IFF Half Carry Clear	BHCC	28	2	4
Branch IFF Half Carry Set	BHCS	29	2	4
Branch IFF Plus	BPL	2A	2	4
Branch IFF Minus	BMI	2B	2	4
Branch IFF Interupt Mask Bit is Clear	BMC	2C	2	4
Branch IFF Interrupt Mask Bit is Set	BMS	2D	2	4
Branch IFF Interrupt Line is Low	BIL	2E	2	4
Branch IFF Interrupt Line is High	BIH	2F	2	4
Branch to Subroutine	BSR	AD	2	8

TABLE 6 — BIT MANIPULATION INSTRUCTIONS

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

TABLE 7 — CONTROL INSTRUCTIONS

		Inherent		
Function	Mnemonic	Op Code	# 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	11
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

TABLE 8 — INSTRUCTION SET

	Addressing Modes										Condition Code				
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			∧	●	∧	∧	∧
ADD		X	X	X		X	X	X			∧	●	∧	∧	∧
AND		X	X	X		X	X	X			●	●	∧	∧	●
ASL	X		X			X	X				●	●	∧	∧	∧
ASR	X		X			X	X				●	●	∧	∧	∧
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			●	●	∧	∧	●
BLO					X						●	●	●	●	●
BLS					X						●	●	●	●	●
BMC					X						●	●	●	●	●
BMI					X						●	●	●	●	●
BMS					X						●	●	●	●	●
BNE					X						●	●	●	●	●
BPL					X						●	●	●	●	●
BRA					X						●	●	●	●	●
BRN					X						●	●	●	●	●
BRCLR										X	●	●	●	●	∧
BRSET										X	●	●	●	●	∧
BSET									X		●	●	●	●	●
BSR					X						●	●	●	●	●
CLL	X										●	●	●	●	0
CLI	X										●	0	●	●	●
CLR	X		X			X	X				●	●	0	1	●
CMP		X	X	X		X	X	X			●	●	∧	∧	∧
COM	X		X			X	X				●	●	∧	∧	1
CPX		X	X	X		X	X	X			●	●	∧	∧	∧
DEC	X		X			X	X				●	●	∧	∧	●
EOR		X	X	X		X	X	X			●	●	∧	∧	●
INC	X		X			X	X				●	●	∧	∧	●
JMP			X	X		X	X	X			●	●	●	●	●
JSR			X	X		X	X	X			●	●	●	●	●
LDA		X	X	X		X	X	X			●	●	∧	∧	●
LDX		X	X	X		X	X	X			●	●	∧	∧	●
LSL	X		X			X	X				●	●	∧	∧	∧
LSR	X		X			X	X				●	●	0	∧	∧
NEQ	X		X			X	X				●	●	∧	∧	∧
NOP	X										●	●	●	●	●
ORA		X	X	X		X	X	X			●	●	∧	∧	●
ROL	X		X			X	X				●	●	∧	∧	∧
RSP	X										●	●	●	●	●

Condition Code Symbols:

H	Half Carry (From Bit 3)	C	Carry/Borrow
I	Interrupt Mask	∧	Test and Set if True, Cleared Otherwise
N	Negative (Sign Bit)	●	Not Affected
Z	Zero	0	Cleared

TABLE 8 — INSTRUCTION SET (Continued)

	Addressing Modes										Condition Code				
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
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	●
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										●	●	●	●	●

Condition Code Symbols:

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

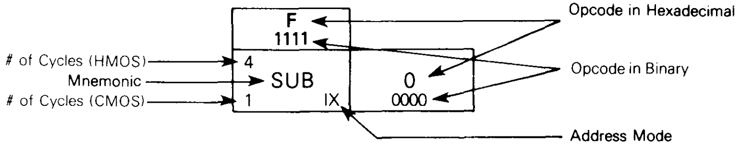
TABLE 9 — M6805 HMOS FAMILY OPCODE MAP

	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	10 BRSET0 3 BTB	7 BSET0 2 BSC	4 BRA 2 REL	6 NEG 2 DIR	4 NEG 1 INH	4 NEG 1 INH	7 NEG 2 IX1	6 NEG 1 IX	9 RTI 1 INH		2 SUB 2 IMM	4 SUB 2 DIR	5 SUB 3 EXT	6 SUB 3 IX2	5 SUB 2 IX1	4 SUB 1 IX	0 0000
1 0001	10 BRCLR0 3 BTB	7 BCLR0 2 BSC	4 BRN 2 REL						6 RTS 1 INH		2 CMP 2 IMM	4 CMP 2 DIR	5 CMP 3 EXT	6 CMP 3 IX2	5 CMP 2 IX1	4 CMP 1 IX	1 0001
2 0010	10 BRSET1 3 BTB	7 BSET1 2 BSC	4 BHI 2 REL								2 SBC 2 IMM	4 SBC 2 DIR	5 SBC 3 EXT	6 SBC 3 IX2	5 SBC 2 IX1	4 SBC 1 IX	2 0010
3 0011	10 BRCLR1 3 BTB	7 BCLR1 2 BSC	4 BLS 2 REL	6 COM 2 DIR	4 COMA 1 INH	4 COMX 1 INH	7 COM 2 IX1	6 COM 1 IX	11 SWI 1 INH		2 CPX 2 IMM	4 CPX 2 DIR	5 CPX 3 EXT	6 CPX 3 IX2	5 CPX 2 IX1	4 CPX 1 IX	3 0011
4 0100	10 BRSET2 3 BTB	7 BSET2 2 BSC	4 BCC 2 REL	6 LSR 2 DTR	4 LSRA 1 INH	4 LSRX 1 INH	7 LSR 2 IX1	6 LSR 1 IX			2 AND 2 IMM	4 AND 2 DIR	5 AND 3 EXT	6 AND 3 IX2	5 AND 2 IX1	4 AND 1 IX	4 0100
5 0101	10 BRCLR2 3 BTB	7 BCLR2 2 BSC	4 BCS 2 REL								2 BIT 2 IMM	4 BIT 2 DIR	5 BIT 3 EXT	6 BIT 3 IX2	5 BIT 2 IX1	4 BIT 1 IX	5 0101
6 0110	10 BRSET3 3 BTB	7 BSET3 2 BSC	4 BNE 2 REL	6 ROR 2 DIR	4 RORA 1 INH	4 RORX 1 INH	7 ROR 2 IX1	6 ROR 1 IX			2 LDA 2 IMM	4 LDA 2 DIR	5 LDA 3 EXT	6 LDA 3 IX2	5 LDA 2 IX1	4 LDA 1 IX	6 0110
7 0111	10 BRCLR3 3 BTB	7 BCLR3 2 BSC	4 BEQ 2 REL	6 ASR 2 DIR	4 ASRA 1 INH	4 ASRX 1 INH	7 ASR 2 IX1	6 ASR 1 IX		2 TAX 1 INH		5 STA 2 DIR	6 STA 3 EXT	7 STA 3 IX2	6 STA 2 IX1	5 STA 1 IX	7 0111
8 1000	10 BRSET4 3 BTB	7 BSET4 2 BSC	4 BHCC 2 REL	6 LSL 2 DIR	4 LSLA 1 INH	4 LSLX 1 INH	7 LSL 2 IX1	6 LSL 1 IX		2 CLC 1 INH	2 EOR 2 IMM	4 EOR 2 DIR	5 EOR 3 EXT	6 EOR 3 IX2	5 EOR 2 IX1	4 EOR 1 IX	8 1000
9 1001	10 BRCLR4 3 BTB	7 BCLR4 2 BSC	4 BHCS 2 REL	6 ROL 2 DIR	4 ROLA 1 INH	4 ROLX 1 INH	7 ROL 2 IX1	6 ROL 1 IX		2 SEC 1 INH	2 ADC 2 IMM	4 ADC 2 DIR	5 ADC 3 EXT	6 ADC 3 IX2	5 ADC 2 IX1	4 ADC 1 IX	9 1001
A 1010	10 BRSET5 3 BTB	7 BSET5 2 BSC	4 BPL 2 REL	6 DEC 2 DIR	4 DECA 1 INH	4 DECX 1 INH	7 DEC 2 IX1	6 DEC 1 IX		2 CLI 1 INH	2 ORA 2 IMM	4 ORA 2 DIR	5 ORA 3 EXT	6 ORA 3 IX2	5 ORA 2 IX1	4 ORA 1 IX	A 1010
B 1011	10 BRCLR5 3 BTB	7 BCLR5 2 BSC	4 BMI 2 REL							2 SEI 1 INH	2 ADD 2 IMM	4 ADD 2 DIR	5 ADD 3 EXT	6 ADD 3 IX2	5 ADD 2 IX1	4 ADD 1 IX	B 1011
C 1100	10 BRSET6 3 BTB	7 BSET6 2 BSC	4 BMC 2 REL	6 INC 2 DIR	4 INCA 1 INH	4 INCX 1 INH	7 INC 2 IX1	6 INC 1 IX		2 RSP 1 INH		3 JMP 2 DIR	4 JMP 3 EXT	5 JMP 3 IX2	4 JMP 2 IX1	3 JMP 1 IX	C 1100
D 1101	10 BRCLR6 3 BTB	7 BCLR6 2 BSC	4 BMS 2 REL	6 TST 2 DIR	4 TSTA 1 INH	4 TSTX 1 INH	7 TST 2 IX1	6 TST 1 IX		2 NOP 1 INH	8 BSR 2 REL	7 JSR 2 DIR	8 JSR 3 EXT	9 JSR 3 IX2	8 JSR 2 IX1	7 JSR 1 IX	D 1101
E 1110	10 BRSET7 3 BTB	7 BSET7 2 BSC	4 BIL 2 REL								2 LDX 2 IMM	4 LDX 2 DIR	5 LDX 3 EXT	6 LDX 3 IX2	5 LDX 2 IX1	4 LDX 1 IX	E 1110
F 1111	10 BRCLR7 3 BTB	7 BCLR7 2 BSC	4 BIH 2 REL	6 CLR 2 DIR	4 CLRA 1 INH	4 CLRX 1 INH	7 CLR 2 IX1	6 CLR 1 IX		2 TXA 1 INH		5 STX 2 DIR	6 STX 3 EXT	7 STX 3 IX2	6 STX 2 IX1	5 STX 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
IX	Indexed (No Offset)
IX1	Indexed, 1 Byte (8-Bit) Offset
IX2	Indexed, 2 Byte (16-Bit) Offset

LEGEND



MC6805T2 MCU CUSTOM ORDERING INFORMATION

Date __________ Customer PO Number __________

Customer Company __________

Motorola Part Numbers

MC __________

Address __________

SC __________

City __________ State __________ Zip __________

Country __________

Phone __________ Extension __________

Customer Contact Person __________

Customer Part Number __________

OPTION LIST

Select the options for your MCU from the following list. A manufacturing mask will be generated from this information.

Timer Clock Source
- □ Internal $\phi 2$ clock (gated by PC0/TIMER)
- □ PC0/TIMER input pin (positive transition)

Port A Output Drive
- □ CMOS and TTL
- □ TTL Only
- □ Open Drain

Timer Prescaler
- □ 2^0 (divide by 1)
- □ 2^1 (divide by 2)
- □ 2^2 (divide by 4)
- □ 2^3 (divide by 8)
- □ 2^4 (divide by 16)
- □ 2^5 (divide by 32)
- □ 2^6 (divide by 64)
- □ 2^7 (divide by 128)

PLL Reference Divider
- □ 2^1 (divide by 2)
- □ 2^2 (divide by 4)
- □ 2^3 (divide by 8)
- □ 2^4 (divide by 16)
- □ 2^5 (divide by 32)
- □ 2^6 (divide by 64)
- □ 2^7 (divide by 128)
- □ 2^8 (divide by 256)
- □ 2^9 (divide by 512)
- □ 2^{10} (divide by 1024)

Pattern Media (All other media requires prior factory approval.

□ EPROMS (two MCM2716s or one MCM2532) □ Floppy Disk

□ Other __________

Clock Freq. __________

Temp. Range __________ □ 0° to +70°C (Standard) □ −40° to +85°C*

*Requires prior factory approval

Marking Information (12 Characters Maximum)

Title __________

Signature __________