THOMSON COMPONENTS MOSTEK

# EF6802 • EF6808

## MICROPROCESSOR WITH CLOCK AND OPTIONAL RAM

T-49-17-06

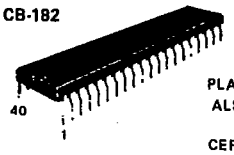
The EF6802 is a monolithic 8-bit microprocessor that contains all the registers and accumulators of the present EF6800 plus an internal clock oscillator and driver on the same chip. In addition, the EF6802 has 128 bytes of on-board RAM located at hex addresses $0000 to $007F. The first 32 bytes of RAM, at hex addresses $0000 to $001F, may be retained in a low power mode by utilizing $V_{CC}$ standby ; thus, facilitating memory retention during a power-down situation.
The EF6802 is completely software compatible with the EF6800 as well as the entire EF6800 family of parts. Hence, the EF6802 is expandable to 64K words.
The EF6808 is identical to the EF6802 without on-board RAM.

- On-Chip Clock Circuit
- 128 x 8 Bit On-Chip RAM
- 32 Bytes of RAM are Retainable
- Software-Compatible with the EF6800
- Expandable to 64K Words
- Standard TTL-Compatible Inputs and Outputs
- 8-Bit Word Size
- 16-Bit Memory Addressing
- Interrupt Capability
- Three available versions : EF6802/08 (1.0 MHz), EF68A02/08 (1.5 MHz), EF68B02/08 (2.0 MHz).

**MOS**
(N-CHANNEL, SILICON-GATE, DEPLETION LOAD)
MICROPROCESSOR WITH CLOCK AND OPTIONAL RAM

**CASES**

CB-182




P SUFFIX
PLASTIC PACKAGE
ALSO AVAILABLE
C SUFFIX
CERAMIC PACKAGE

CB-521

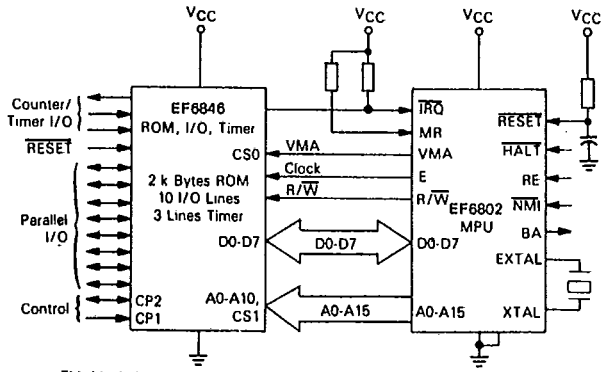
FN SUFFIX
PLCC 44

CB-708

E SUFFIX
LCCC 44

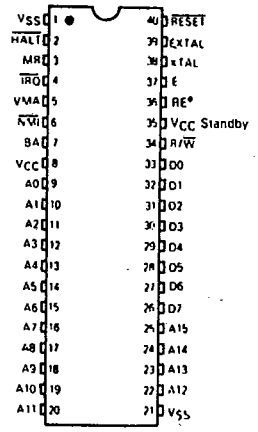
Hi-Rel versions available - See chapter 9

**TYPICAL MICROCOMPUTER**



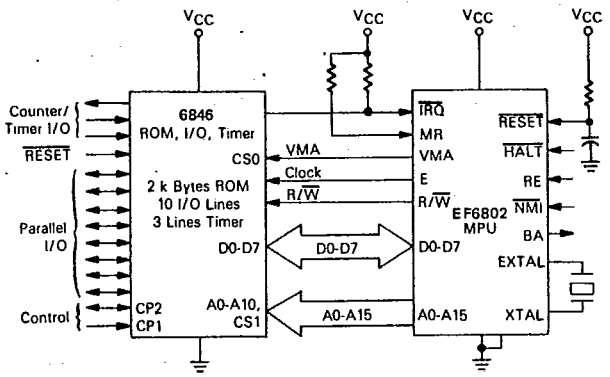

This block diagram shows a typical cost effective microcomputer. The MPU is the center of the microcoputer system and is shown in a minimum system interfacing with a ROM combination chip. It is not intended that this system be limited to this function but that it be expandable with other parts in the 6800 Microcomputer family.

**PIN ASSIGNMENT**




*Pin 36 must be tied to ground for the EF6808

TYPICAL MICROCOMPUTER




This block diagram shows a typical cost effective microcomputer. The MPU is the center of the microcomputer system and is shown in a minimum system interfacing with a ROM combination chip. It is not intended that this system be limited to this function but that it be expandable with other parts in the 6800 Microcomputer family.

## 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<br>EF6802, EF680A02, EF680B02<br>EF6802, EF68A02, EF68B02 : V suffix<br>EF6802, EF68A02 : M suffix<br>EF6808, EF68A08, EF68B08 | $T_A$ | 0 to +70<br>−40 to +85<br>−55 to +125<br>0 to +70 | °C |
| Storage Temperature Range | $T_{stg}$ | −55 to +150 | °C |

This input 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. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either $V_{SS}$ or $V_{CC}$).

## THERMAL CHARACTERISTICS

| Characteristic | Symbol | Value | Unit |
|---|---|---|---|
| Average Thermal Resistance (Junction to Ambient)<br>Plastic<br>Ceramic<br>PLCC | $\theta_{JA}$ | <br>100<br>50<br>100 | °C/W |

## POWER CONSIDERATIONS

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

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

Where:

$T_A$ = Ambient Temperature, °C

$\theta_{JA}$ = 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 $P_{PORT} \ll P_{INT}$ 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:

$$P_D = K + (T_J + 273°C) \quad (2)$$

Solving equations 1 and 2 for K gives:

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

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 $T_A$.

T-49-17-06

**DC ELECTRICAL CHARACTERISTICS** ($V_{CC}$ = 5.0 Vdc ±5%, $V_{SS}$ = 0, $T_A$ = 0 to 70°C, unless otherwise noted)

| Characteristic | | Symbol | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| Input High Voltage | EXTAL | $V_{IH}$ | $V_{SS}$+2.0 | – | $V_{CC}$ | V |
| | RESET | | $V_{SS}$+4.0 | – | $V_{CC}$ | |
| Input Low Voltage | Logic, EXTAL, RESET | $V_{IL}$ | $V_{SS}$−0.3 | – | $V_{SS}$+0.8 | V |
| Input Leakage Current ($V_{in}$ = 0 to 5.25 V, $V_{CC}$ = max) | Logic | $I_{in}$ | – | 1.0 | 2.5 | μA |
| Output High Voltage | | | | | | |
| ($I_{Load}$ = −205 μA, $V_{CC}$ = min) | D0-D7 | $V_{OH}$ | $V_{SS}$+2.4 | – | – | V |
| ($I_{Load}$ = −145 μA, $V_{CC}$ = min) | A0-A15, R/$\overline{W}$, VMA, E | | $V_{SS}$+2.4 | – | – | |
| ($I_{Load}$ = −100 μA, $V_{CC}$ = min) | BA | | $V_{SS}$+2.4 | – | – | |
| Output Low Voltage ($I_{Load}$ = 1.6 mA, $V_{CC}$ = min) | | $V_{OL}$ | – | – | $V_{SS}$+0.4 | V |
| Internal Power Dissipation (Measured at $T_A$ = 0°C) | | $P_{INT}$ | – | 0.750 | 1.0 | W |
| $V_{CC}$ Standby | Power Down | $V_{SBB}$ | 4.0 | – | 5.25 | V |
| | Power Up | $V_{SB}$ | 4.75 | – | 5.25 | |
| Standby Current | | $I_{SBB}$ | – | – | 8.0 | mA |
| Capacitance # ($V_{in}$ = 0, $T_A$ = 25°C, f = 1.0 MHz) | D0-D7 | $C_{in}$ | – | 10 | 12.5 | pF |
| | Logic Inputs, EXTAL | | – | 6.5 | 10 | |
| | A0-A15, R/$\overline{W}$, VMA | $C_{out}$ | – | – | 12 | pF |

*In power-down mode, maximum power dissipation is less than 42 mW.
#Capacitances are periodically sampled rather than 100% tested.

**CONTROL TIMING** ($V_{CC}$ = 5.0 V ±5%, $V_{SS}$ = 0, $T_A$ = $T_L$ to $T_H$, unless otherwise noted)

| Characteristics | Symbol | EF6802 EF6808 | | EF68A02 EF68A08 | | EF68B02 EF68B08 | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | Min | Max | |
| Frequency of Operation | $f_o$ | 0.1 | 1.0 | 0.1 | 1.5 | 0.1 | 2.0 | MHz |
| Crystal Frequency | $f_{XTAL}$ | 1.0 | 4.0 | 1.0 | 6.0 | 1.0 | 8.0 | MHz |
| External Oscillator Frequency | $4xf_o$ | 0.4 | 4.0 | 0.4 | 6.0 | 0.4 | 8.0 | MHz |
| Crystal Oscillator Start Up Time | $t_{rc}$ | 100 | – | 100 | – | 100 | – | ms |
| Processor Controls (HALT, MR, RE, RESET, IRQ NMI) | | | | | | | | |
| Processor Control Setup Time | $t_{PCS}$ | 200 | – | 140 | – | 110 | – | ns |
| Processor Control Rise and Fall Time (Does Not Apply to RESET) | $t_{PCr}$, $t_{PCf}$ | – | 100 | – | 100 | – | 100 | ns |

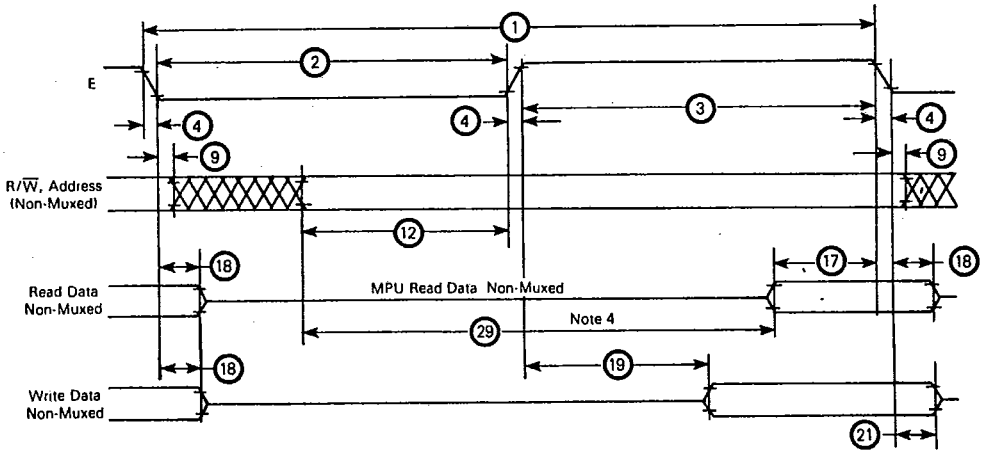
## BUS TIMING CHARACTERISTICS

87D 09233 D T-49-17-06

| Ident. Number | Characteristic | Symbol | EF6802 EF6808 | | EF68A02 EF68A08 | | EF68B02 EF68B08 | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | Min | Max | |
| 1 | Cycle Time | $t_{cyc}$ | 1.0 | 10 | 0.667 | 10 | 0.5 | 10 | µs |
| 2 | Pulse Width, E Low | $PW_{EL}$ | 450 | 5000 | 280 | 5000 | 210 | 5000 | ns |
| 3 | Pulse Width, E High | $PW_{EH}$ | 450 | 9500 | 280 | 9700 | 220 | 9700 | ns |
| 4 | Clock Rise and Fall Time | $t_r$, $t_f$ | – | 25 | – | 25 | – | 25 | ns |
| 9 | Address Hold Time* | $t_{AH}$ | 20 | – | 20 | – | 20 | – | ns |
| 12 | Non-Muxed Address Valid Time to E (See Note 5) | $t_{AV1}$<br>$t_{AV2}$ | 160<br>– | –<br>270 | 100<br>– | –<br>– | 50<br>– | –<br>– | ns |
| 17 | Read Data Setup Time | $t_{DSR}$ | 100 | – | 70 | – | 60 | – | ns |
| 18 | Read Data Hold Time | $t_{DHR}$ | 10 | – | 10 | – | 10 | – | ns |
| 19 | Write Data Delay Time | $t_{DDW}$ | – | 225 | – | 170 | – | 160 | ns |
| 21 | Write Data Hold Time* | $t_{DHW}$ | 30 | – | 20 | – | 20 | – | ns |
| 29 | Usable Access Time (See Note 4) | $t_{ACC}$ | 535 | – | 335 | – | 235 | – | ns |

* Address and data hold times are periodically tested rather than 100% tested.

FIGURE 2 – BUS TIMING

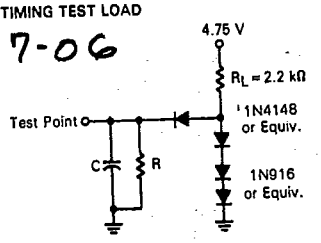



NOTES:

1. Voltage levels shown are $V_L \leq 0.4$ V, $V_H \geq 2.4$ V, unless otherwise specified.
2. Measurement points shown are 0.8 V and 2.0 V, unless otherwise noted.
3. All electricals shown for the EF6802 apply to the EF6808, unless otherwise noted.
4. Usable access time is computed by: 12+3+4−17.
5. If programs are not executed from on-board RAM, TAV1 applies. If programs are to be stored and executed from on-board RAM, TAV2 applies. For normal data storage in the on-board RAM, this extended delay does not apply. Programs cannot be executed from on-board RAM when using A and B parts (EF68A02, EF68A08, EF68B02, EF68B08). On-board RAM can be used for data storage with all parts.
6. All electrical and control characteristics are referenced from: $T_L = 0°C$ minimum and $T_H = 70°C$ maximum.

FIGURE 3 – BUS TIMING TEST LOAD

87D 09234 DT-49-17-06




C = 130 pF for D0-D7, E
= 90 pF for A0-A15, R/$\overline{W}$, and VMA
= 30 pF for BA
R = 11.7 kΩ for D0-D7, E
= 16.5 kΩ for A0-A15, R/$\overline{W}$, and VMA
= 24 kΩ for BA

FIGURE 4 – TYPICAL DATA BUS OUTPUT DELAY versus CAPACITIVE LOADING

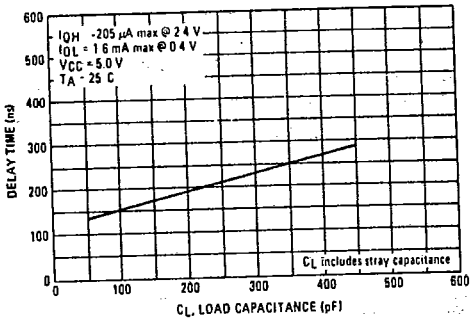



FIGURE 5 – TYPICAL READ/WRITE, VMA AND ADDRESS OUTPUT DELAY versus CAPACITIVE LOADING

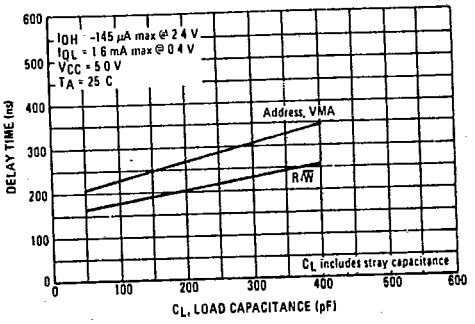
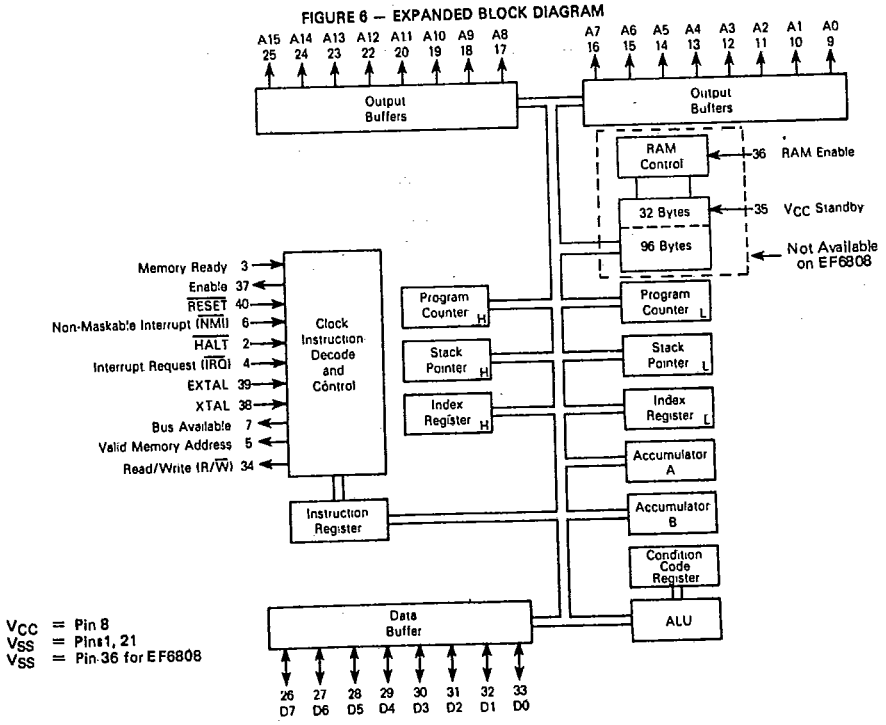



FIGURE 6 – EXPANDED BLOCK DIAGRAM



## MPU REGISTERS

87D 09235 DT-49-17-06

A general block diagram of the EF6802 is shown in Figure 6. As shown, the number and configuration of the registers are the same as for the EF6800. The 128x8-bit RAM* has been added to the basic MPU. The first 32 bytes can be retained during power-up and power-down conditions via the RE signal.

The EF6808 is identical to the EF6802 except for on-board RAM. Since the EF6808 does not have on-board RAM pin 36 must be tied to ground allowing the processor to utilize up to 64K bytes of external memory.

The MPU has three 16-bit registers and three 8-bit registers available for use by the programmer (Figure 7).

### PROGRAM COUNTER

The program counter is a two byte (16-bit) register that points to the current program address.

### STACK POINTER

The stack pointer is a two byte register that contains the address of the next available location in an external push-down/pop-up stack. This stack is normally a random access read/write memory that may have any location (address) that is convenient. In those applications that require storage of information in the stack when power is lost, the stack must be non-volatile.

### INDEX REGISTER

The index register is a two byte register that is used to store data or a 16-bit memory address for the indexed mode of memory addressing.

### ACCUMULATORS

The MPU contains two 8-bit accumulators that are used to hold operands and results from an arithmetic logic unit (ALU).

### CONDITION CODE REGISTER

The condition code register indicates the results of an Arithmetic Logic Unit operation: Negative (N), Zero (Z), Overflow (V), Carry from bit 7 (C), and Half Carry from bit 3 (H). These bits of the Condition Code Register are used as testable conditions for the conditional branch instructions. Bit 4 is the interrupt mask bit (I). The unused bits of the Condition Code Register (b6 and b7) are ones.

Figure 8 shows the order of saving the microprocessor status within the stack.

*If programs are not executed from on-board RAM, TAV1 applies. If programs are to be stored and executed from on-board RAM, TAV2 applies. For normal data storage in the on-board RAM, this extended delay does not apply. Programs cannot be executed from on-board RAM when using A and B parts (EF68A02, EF68A08, EF68B02, and EF68B08). On-board RAM can be used for data storage with all parts.

FIGURE 7 — PROGRAMMING MODEL OF THE MICROPROCESSING UNIT

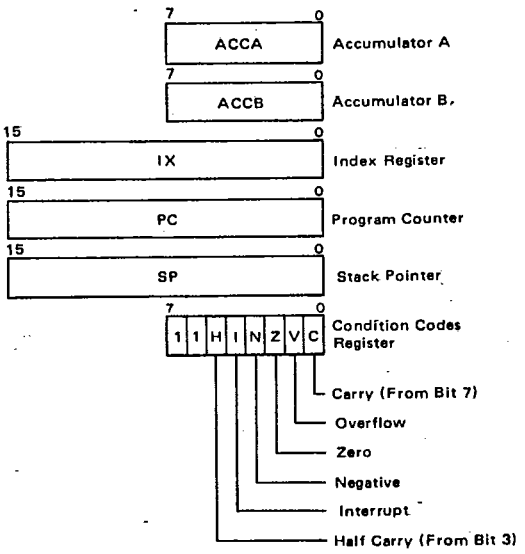
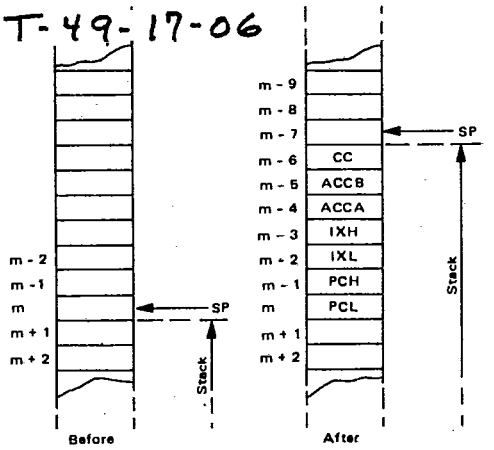


FIGURE 8 – SAVING THE STATUS OF THE MICROPROCESSOR IN THE STACK

87D 09236 D T-49-17-06

SP = Stack Pointer
CC = Condition Codes (Also called the Processor Status Byte)
ACCB = Accumulator B
ACCA = Accumulator A
IXH = Index Register, Higher Order 8 Bits
IXL = Index Register, Lower Order 8 Bits
PCH = Program Counter, Higher Order 8 Bits
PCL = Program Counter, Lower Order 8 Bits




## MPU SIGNAL DESCRIPTION

Proper operation of the MPU requires that certain control and timing signals be provided to accomplish specific functions and that other signal lines be monitored to determine the state of the processor. These control and timing signals are similar to those of the EF6800 except that TSC, DBE, $\phi$1, $\phi$2 input, and two unused pins have been eliminated, and the following signal and timing lines have been added:

RAM Enable (RE)
Crystal Connections EXTAL and XTAL
Memory Ready (MR)
$V_{CC}$ Standby
Enable $\phi$2 Output (E)

The following is a summary of the MPU signals:

### ADDRESS BUS (A0-A15)

Sixteen pins are used for the address bus. The outputs are capable of driving one standard TTL load and 90 pF. These lines do not have three-state capability.

### DATA BUS (D0-D7)

Eight pins are used for the data bus. It is bidirectional, transferring data to and from the memory and peripheral devices. It also has three-state output buffers capable of driving one standard TTL load and 130 pF.

Data bus will be in the output mode when the internal RAM is accessed and RE will be high. This prohibits external data entering the MPU. It should be noted that the internal RAM is fully decoded from $0000 to $007F. External RAM at $0000 to $007F must be disabled when internal RAM is accessed.

### $\overline{\text{HALT}}$

When this input is in the low state, all activity in the machine will be halted. This input is level sensitive. In the $\overline{\text{HALT}}$ mode, the machine will stop at the end of an instruction, bus available will be at a high state, valid memory address will be at a low state. The address bus will display the address of the next instruction.

To ensure single instruction operation, transition of the $\overline{\text{HALT}}$ line must occur $t_{PCS}$ before the falling edge of E and the $\overline{\text{HALT}}$ line must go high for one clock cycle.

$\overline{\text{HALT}}$ should be tied high if not used. This is good engineering design practice in general and necessary to ensure proper operation of the part.

### READ/WRITE (R/$\overline{\text{W}}$)

This TTL-compatible output signals the peripherals and memory devices whether the MPU is in a read (high) or write (low) state. The normal standby state of this signal is read (high). When the processor is halted, it will be in the read state. This output is capable of driving one standard TTL load and 90 pF.

### VALID MEMORY ADDRESS (VMA)

This output indicates to peripheral devices that there is a valid address on the address bus. In normal operation, this signal should be utilized for enabling peripheral interfaces such as the PIA and ACIA. This signal is not three-state. One standard TTL load and 90 pF may be directly driven by this active high signal.

**BUS AVAILABLE (BA)** – The bus available signal will normally be in the low state; when activated, it will go to the high state indicating that the microprocessor has stopped and that the address bus is available (but not in a three-state condition). This will occur if the $\overline{\text{HALT}}$ line is in the low state or the processor is in the WAIT state as a result of the execution of a WAIT instruction. At such time, all three-state output drivers will go to their off-state and other outputs to their normally inactive level. The processor is removed from the

WAIT state by the occurrence of a maskable (mask bit I=0) or nonmaskable interrupt. This output is capable of driving one standard TTL load and 30 pF.

### INTERRUPT REQUEST ($\overline{IRQ}$) T-49-17-06

A low level on this input requests that an interrupt sequence be generated within the machine. The processor will wait until it completes the current instruction that is being excuted before it recognizes the request. At that time, if the interrupt mask bit in the condition code register is not set, the machine will begin an interrupt sequence. The index register, program counter, accumulators, and condition code register are stored away on the stack. Next the MPU will respond to the interrupt request by setting the interrupt mask bit high so that no further interrupts may occur. At the end of the cycle, a 16-bit vectoring address which is located in memory locations $FFF8 and $FFF9 is loaded which causes the MPU to branch to an interrupt routine in memory.

The $\overline{HALT}$ line must be in the high state for interrupts to be serviced. Interrupts will be latched internally while $\overline{HALT}$ is low.

A nominal 3 kΩ pullup resistor to $V_{CC}$ should be used for wire-OR end optimum control of interrupts. $\overline{IRQ}$ may be tied directly to $V_{CC}$ if not used.

### $\overline{RESET}$

This input is used to reset and start the MPU from a power-down condition, resulting from a power failure or an initial start-up of the processor. When this line is low, the MPU is inactive and the information in the registers will be lost. If a high level is detected on the input, this will signal the MPU to begin the restart sequence. This will start execution of a routine to initialize the processor from its reset condition. All the higher order address lines will be forced high. For the restart, the last two ($FFFE, $FFFF) locations in memory will be used to load the program that is addressed by the program counter. During the restart routine, the interrupt mask bit is set and must be reset before the MPU can be interrupted by $\overline{IRQ}$. Power-up and reset timing and power-down sequences are shown in Figures 9 and 10, respectively.

$\overline{RESET}$, when brought low, must be held low at least three clock cycles. This allows adequate time to respond internally to the reset. This is independent of the $t_{rc}$ power-up reset that is required.

When $\overline{RESET}$ is released it *must* go through the low-to-high threshold without bouncing, oscillating, or otherwise causing an erroneous reset (less than three clock cycles). This may cause improper MPU operation until the next valid reset.

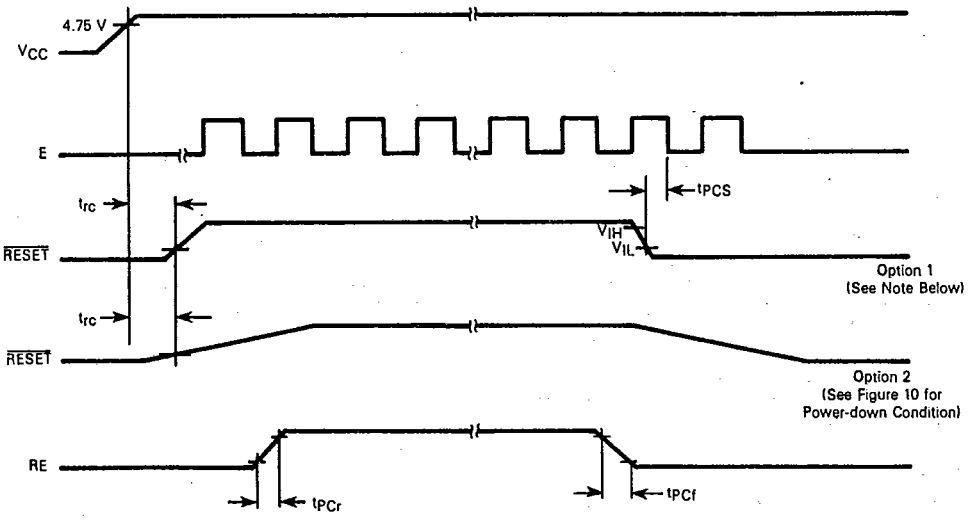
### NON-MASKABLE INTERRUPT ($\overline{NMI}$)

A low-going edge on this input requests that a non-maskable interrupt sequence be generated within the processor. As with the interrupt request signal, the processor will complete the current instruction that is being executed before it recognizes the $\overline{NMI}$ signal. The interrupt mask bit in the condition code register has no effect on $\overline{NMI}$.

The index register, program counter, accumulators, and condition code registers are stored away on the stack. At the end of the cycle, a 16-bit vectoring address which is located in memory locations $FFFC and $FFFD is loaded causing the MPU to branch to an interrupt service routine in memory.

A nominal 3 kΩ pullup resistor to $V_{CC}$ should be used for wire-OR and optimum control of interrupts. $\overline{NMI}$ may be tied

FIGURE 9 — POWER-UP AND RESET TIMING




NOTE: If option 1 is chosen, $\overline{RESET}$ and RE pins can be tied together.

directly to $V_{CC}$ if not used.

Inputs $\overline{IRQ}$ and $\overline{NMI}$ are hardware interrupt lines that are sampled when E is high and will start the interrupt routine on a low E following the completion of an instruction.

Figure 11 is a flowchart describing the major decision paths and interrupt vectors of the microprocessor. Table 1 gives the memory map for interrupt vectors.

TABLE 1 — MEMORY MAP FOR INTERRUPT VECTORS

| Vector | | Description |
|---|---|---|
| MS | LS | |
| $FFFE | $FFFF | Restart |
| $FFFC | $FFFD | Non-Maskable Interrupt |
| $FFFA | $FFFB | Software Interrupt |
| $FFF8 | $FFF9 | Interrupt Request |

FIGURE 10 — POWER-DOWN SEQUENCE

87D 09238 D T-49-17-06

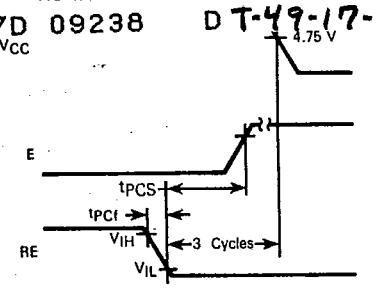



FIGURE 11 — MPU FLOWCHART

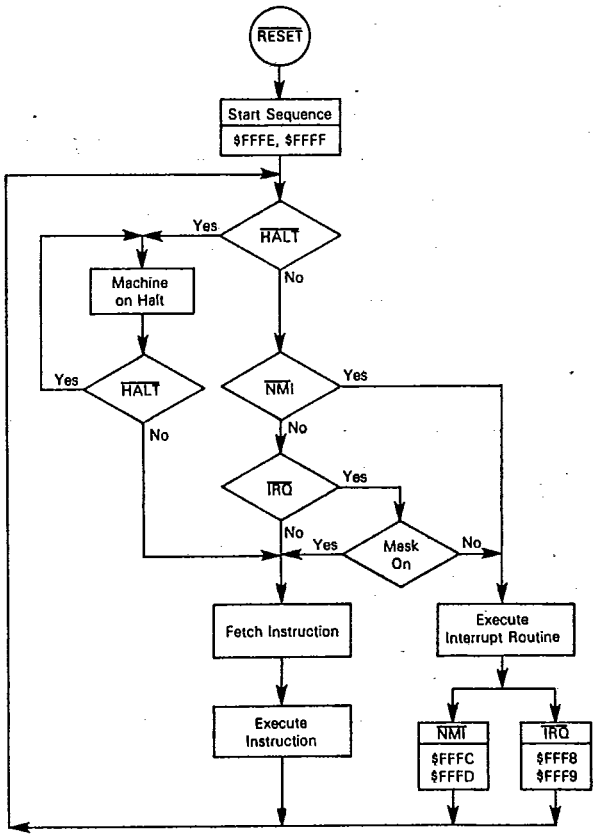
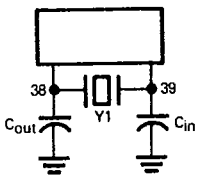


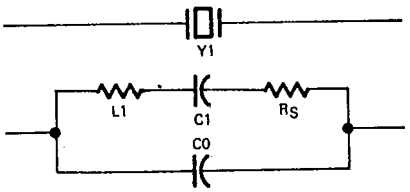
FIGURE 12 – CRYSTAL SPECIFICATIONS

87D 09239 D T-49-17-06




| Y1 | Cin | Cout |
|---|---|---|
| 3.58 MHz | 27 pF | 27 pF |
| 4 MHz | 27 pF | 27 pF |
| 6 MHz | 20 pF | 20 pF |
| 8 MHz | 18 pF | 18 pF |

Crystal Loading




Nominal Crystal Parameters*

| | 3.58 MHz | 4.0 MHz | 6.0 MHz | 8.0 MHz |
|---|---|---|---|---|
| RS | 60 Ω | 50 Ω | 30-50 Ω | 20-40 Ω |
| C0 | 3.5 pF | 6.5 pF | 4-6 pF | 4-6 pF |
| C1 | 0.015 pF | 0.025 pF | 0.01-0.02 pF | 0.01-0.02 pF |
| Q | >40K | >30K | >20K | >20K |

*These are representative AT-cut parallel resonance crystal parameters only. Crystals of other types of cuts may also be used.

Figure 13 – SUGGESTED PC BOARD LAYOUT

Example of Board Design Using the Crystal Oscillator

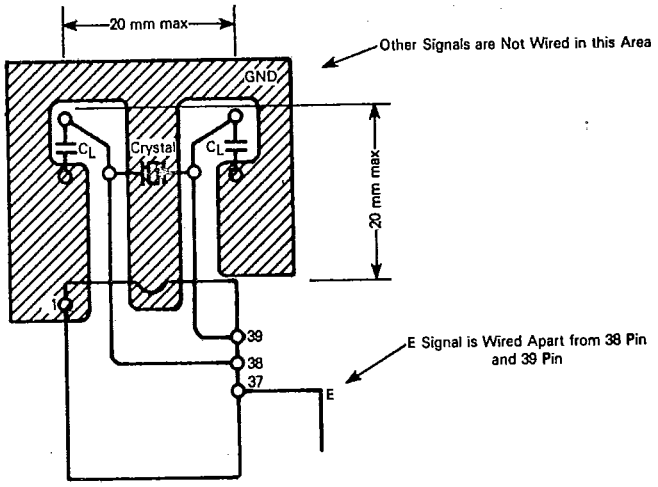


FIGURE 14 – MEMORY READY SYNCHRONIZATION

87D 09240 D T-49-17-06

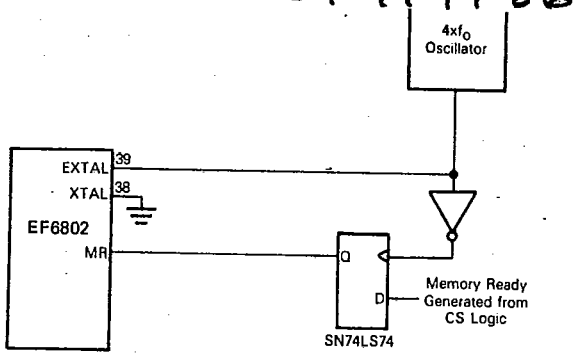
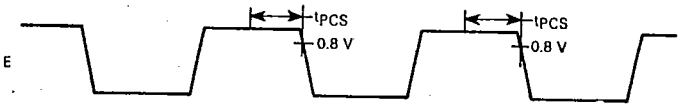


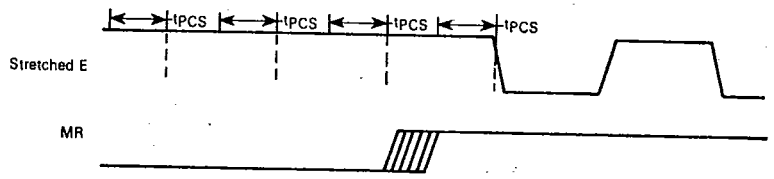

FIGURE 15 – MR NEGATIVE SETUP TIME REQUIREMENT

E Clock Stretch




The E clock will be stretched at end of E high of the cycle during which MR negative meets the $t_{PCS}$ setup time. The $t_{PCS}$ setup time is referenced to the fall of E. If the $t_{PCS}$ setup time is not met, E will be stretched at the end of the next E-high ½ cycle. E will be stretched in integral multiples of ½ cycles.

Resuming E Clocking




The E clock will resume normal operation at the end of the ½ cycle during which MR assertion meets the $t_{PCS}$ setup time. The $t_{PCS}$ setup time is referenced to transitions of E were it not stretched. If $t_{PCS}$ setup time is not met, E will fall at the second possible transition time after MR is asserted. There is no direct means of determining when the $t_{PCS}$ references occur, unless the synchronizing circuit of Figure 14 is used.

### RAM ENABLE (RE – EF6802 ONLY)

A TTL-compatible RAM enable input controls the on-chip RAM of the EF6802. When placed in the high state, the on-chip memory is enabled to respond to the MPU controls. In the low state, RAM is disabled. This pin may also be utilized to disable reading and writing the on-chip RAM during a power-down situation. RAM Enable must be low three cycles before $V_{CC}$ goes below 4.75 V during power-down. RAM enable must be tied low on the EF6808. RE should be tied to the correct high or low state if not used.

### EXTAL AND XTAL

These inputs are used for the internal oscillator that may be crystal controlled. These connections are for a parallel resonant fundamental crystal (see Figure 12). (AT-cut.) A divide-by-four circuit has been added so a 4 MHz crystal may be used in lieu of a 1 MHz crystal for a more cost-effective system. An example of the crystal circuit layout is shown in Figure 13. Pin 39 may be driven externally by a TTL input signal four times the required E clock frequency. Pin 38 is to be grounded.

An RC network is not directly usable as a frequency source on pins 38 and 39. An RC network type TTL or CMOS oscillator will work well as long as the TTL or CMOS output drives the on-chip oscillator.

LC networks are not recommended to be used in place of the crystal.

If an external clock is used, it may not be halted for more than $t_{PW\phi L}$. The EF6802 and EF6808 are dynamic parts except for the internal RAM, and require the external clock to retain information.

### MEMORY READY (MR)

MR is a TTL-compatible input signal controlling the stretching of E. Use of MR requires synchronization with the $4xf_o$ signal, as shown in Figure 14. When MR is high, E will be in normal operation. When MR is low, E will be stretched integral numbers of half periods, thus allowing interface to slow memories. Memory Ready timing is shown in Figure 15.

MR should be tied high (connected directly to $V_{CC}$) if not used. This is necessary to ensure proper operation of the part. A maximum stretch is $t_{cyc}$.

### ENABLE (E)

This pin supplies the clock for the MPU and the rest of the system. This is e single-phase, TTL-compatible clock. This clock may be conditioned by a memory read signal. This is equivalent to $\phi 2$ on the EF6800. This output is capable of driving one standard TTL load and 130 pF.

### $V_{CC}$ STANDBY (EF6802 ONLY)

This pin supplies the dc voltage to the first 32 bytes of RAM as well as the RAM Enable (RE) control logic. Thus, retention of data in this portion of the RAM on a power-up, power-down, or standby condition is guaranteed. Maximum current drain at $V_{SB}$ maximum is $I_{SBB}$.

## MPU INSTRUCTION SET

87D 09241 D T-49-17-06

The instruction set has 72 different instructions. Included are binary and decimal arithmetic, logical, shift, rotate, load, store, conditional or unconditional branch, interrupt and stack manipulation instructions (Tables 2 through 6). The instruction set is the same as that for the EF6800.

## MPU ADDRESSING MODES

There are seven address modes that can be used by a programmer, with the addressing mode a function of both the type of instruction and the coding within the instruction. A summary of the addressing modes for a particular instruction can be found in Table 7 along with the associated instruction execution time that is given in machine cycles. With a bus frequency of 1 MHz, these times would be microseconds.

### ACCUMULATOR (ACCX) ADDRESSING

In accumulator only addressing, either accumulator A or accumulator B is specified. These are one-byte instructions.

### IMMEDIATE ADDRESSING

In immediate addressing, the operand is contained in the second byte of the instruction except LDS and LDX which have the operand in the second and third bytes of the instruction. The MPU addresses this location when it fetches the immediate instruction for execution. These are two- or three-byte instructions.

### DIRECT ADDRESSING

In direct addressing, the address of the operand is contained in the second byte of the instruction. Direct addressing allows the user to directly address the lowest 256 bytes in the machine, i.e., locations zero through 255. Enhanced execution times are achieved by storing data in these locations. In most configurations, it should be a random-access memory. These are two-byte instructions.

### EXTENDED ADDRESSING

In extended addressing, the address contained in the second byte of the instruction is used as the higher eight bits of the address of the operand. The third byte of the instruction is used as the lower eight bits of the address for the operand. This is an absolute address in memory. These are three-byte instructions.

### INDEXED ADDRESSING

In indexed addressing, the address contained in the second byte of the instruction is added to the index register's lowest eight bits in the MPU. The carry is then added to the higher order eight bits of the index register. This result is then used to address memory. The modified address is held in a temporary address register so there is no change to the index register. These are two-byte instructions.

T-49-17-06

### IMPLIED ADDRESSING

In the implied addressing mode, the instruction gives the address (i.e., stack pointer, index register, etc.). These are one-byte instructions.

### RELATIVE ADDRESSING

In relative addressing, the address contained in the second byte of the instruction is added to the program counter's lowest eight bits plus two. The carry or borrow is then added to the high eight bits. This allows the user to address data within a range of − 125 to + 129 bytes of the present instruction. These are two-byte instructions.

TABLE 2 — MICROPROCESSOR INSTRUCTION SET — ALPHABETIC SEQUENCE

| | | | | | |
|---|---|---|---|---|---|
| ABA | Add Accumulators | CLR | Clear | PUL | Pull Data |
| ADC | Add with Carry | CLV | Clear Overflow | ROL | Rotate Left |
| ADD | Add | CMP | Compare | ROR | Rotate Right |
| AND | Logical And | COM | Complement | RTI | Return from Interrupt |
| ASL | Arithmetic Shift Left | CPX | Compare Index Register | RTS | Return from Subroutine |
| ASR | Arithmetic Shift Right | DAA | Decimal Adjust | SBA | Subtract Accumulators |
| BCC | Branch if Carry Clear | DEC | Decrement | SBC | Subtract with Carry |
| BCS | Branch if Carry Set | DES | Decrement Stack Pointer | SEC | Set Carry |
| BEQ | Branch if Equal to Zero | DEX | Decrement Index Register | SEI | Set Interrupt Mask |
| BGE | Branch if Greater or Equal Zero | EOR | Exclusive OR | SEV | Set Overflow |
| BGT | Branch if Greater than Zero | INC | Increment | STA | Store Accumulator |
| BHI | Branch if Higher | INS | Increment Stack Pointer | STS | Store Stack Register |
| BIT | Bit Test | INX | Increment Index Register | STX | Store Index Register |
| BLE | Branch if Less or Equal | JMP | Jump | SUB | Subtract |
| BLS | Branch if Lower or Same | JSR | Jump to Subroutine | SWI | Software Interrupt |
| BLT | Branch if Less than Zero | LDA | Load Accumulator | TAB | Transfer Accumulators |
| BMI | Branch if Minus | LDS | Load Stack Pointer | TAP | Transfer Accumulators to Condition Code Reg. |
| BNE | Branch if Not Equal to Zero | LDX | Load Index Register | TBA | Transfer Accumulators |
| BPL | Branch if Plus | LSR | Logical Shift Right | TPA | Transfer Condition Code Reg. to Accumulator |
| BRA | Branch Always | NEG | Negate | TST | Test |
| BSR | Branch to Subroutine | NOP | No Operation | TSX | Transfer Stack Pointer to Index Register |
| BVC | Branch if Overflow Clear | ORA | Inclusive OR Accumulator | TXS | Transfer Index Register to Stack Pointer |
| BVS | Branch if Overflow Set | PSH | Push Data | WAI | Wait for Interrupt |
| CBA | Compare Accumulators | | | | |
| CLC | Clear Carry | | | | |
| CLI | Clear Interrupt Mask | | | | |

87D 09243 D T-49-17-06

## TABLE 3 — ACCUMULATOR AND MEMORY INSTRUCTIONS

| OPERATIONS | MNEMONIC | IMMED OP | IMMED ~ | IMMED = | DIRECT OP | DIRECT ~ | DIRECT = | INDEX OP | INDEX ~ | INDEX = | EXTND OP | EXTND ~ | EXTND = | IMPLIED OP | IMPLIED ~ | IMPLIED = | BOOLEAN/ARITHMETIC OPERATION (All register labels refer to contents) | 5 H | 4 I | 3 N | 2 Z | 1 V | 0 C |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Add | ADDA | 8B | 2 | 2 | 9B | 3 | 2 | AB | 5 | 2 | BB | 4 | 3 | | | | A + M → A | ↕ | • | ↕ | ↕ | ↕ | ↕ |
| | ADDB | CB | 2 | 2 | DB | 3 | 2 | EB | 5 | 2 | FB | 4 | 3 | | | | B + M → B | ↕ | • | ↕ | ↕ | ↕ | ↕ |
| Add Acmltrs | ABA | | | | | | | | | | | | | 1B | 2 | 1 | A + B → A | ↕ | • | ↕ | ↕ | ↕ | ↕ |
| Add with Carry | ADCA | 89 | 2 | 2 | 99 | 3 | 2 | A9 | 5 | 2 | B9 | 4 | 3 | | | | A + M + C → A | ↕ | • | ↕ | ↕ | ↕ | ↕ |
| | ADCB | C9 | 2 | 2 | D9 | 3 | 2 | E9 | 5 | 2 | F9 | 4 | 3 | | | | B + M + C → B | ↕ | • | ↕ | ↕ | ↕ | ↕ |
| And | ANDA | 84 | 2 | 2 | 94 | 3 | 2 | A4 | 5 | 2 | B4 | 4 | 3 | | | | A · M → A | • | • | ↕ | ↕ | R | • |
| | ANDB | C4 | 2 | 2 | D4 | 3 | 2 | E4 | 5 | 2 | F4 | 4 | 3 | | | | B · M → B | • | • | ↕ | ↕ | R | • |
| Bit Test | BITA | 85 | 2 | 2 | 95 | 3 | 2 | A5 | 5 | 2 | B5 | 4 | 3 | | | | A · M | • | • | ↕ | ↕ | R | • |
| | BITB | C5 | 2 | 2 | D5 | 3 | 2 | E5 | 5 | 2 | F5 | 4 | 3 | | | | B · M | • | • | ↕ | ↕ | R | • |
| Clear | CLR | | | | | | | 6F | 7 | 2 | 7F | 6 | 3 | | | | 00 → M | • | • | R | S | R | R |
| | CLRA | | | | | | | | | | | | | 4F | 2 | 1 | 00 → A | • | • | R | S | R | R |
| | CLRB | | | | | | | | | | | | | 5F | 2 | 1 | 00 → B | • | • | R | S | R | R |
| Compare | CMPA | 81 | 2 | 2 | 91 | 3 | 2 | A1 | 5 | 2 | B1 | 4 | 3 | | | | A − M | • | • | ↕ | ↕ | ↕ | ↕ |
| | CMPB | C1 | 2 | 2 | D1 | 3 | 2 | E1 | 5 | 2 | F1 | 4 | 3 | | | | B − M | • | • | ↕ | ↕ | ↕ | ↕ |
| Compare Acmltrs | CBA | | | | | | | | | | | | | 11 | 2 | 1 | A − B | • | • | ↕ | ↕ | ↕ | ↕ |
| Complement, 1's | COM | | | | | | | 63 | 7 | 2 | 73 | 6 | 3 | | | | M → M | • | • | ↕ | ↕ | R | S |
| | COMA | | | | | | | | | | | | | 43 | 2 | 1 | A → A | • | • | ↕ | ↕ | R | S |
| | COMB | | | | | | | | | | | | | 53 | 2 | 1 | B → B | • | • | ↕ | ↕ | R | S |
| Complement, 2's (Negate) | NEG | | | | | | | 60 | 7 | 2 | 70 | 6 | 3 | | | | 00 − M → M | • | • | ↕ | ↕ | ① | ② |
| | NEGA | | | | | | | | | | | | | 40 | 2 | 1 | 00 − A → A | • | • | ↕ | ↕ | ① | ② |
| | NEGB | | | | | | | | | | | | | 50 | 2 | 1 | 00 − B → B | • | • | ↕ | ↕ | ① | ② |
| Decimal Adjust, A | DAA | | | | | | | | | | | | | 19 | 2 | 1 | Converts Binary Add of BCD Characters into BCD Format | • | • | ↕ | ↕ | ↕ | ③ |
| Decrement | DEC | | | | | | | 6A | 7 | 2 | 7A | 6 | 3 | | | | M − 1 → M | • | • | ↕ | ↕ | ④ | • |
| | DECA | | | | | | | | | | | | | 4A | 2 | 1 | A − 1 → A | • | • | ↕ | ↕ | ④ | • |
| | DECB | | | | | | | | | | | | | 5A | 2 | 1 | B − 1 → B | • | • | ↕ | ↕ | ④ | • |
| Exclusive OR | EORA | 88 | 2 | 2 | 98 | 3 | 2 | A8 | 5 | 2 | B8 | 4 | 3 | | | | A⊕M → A | • | • | ↕ | ↕ | R | • |
| | EORB | C8 | 2 | 2 | D8 | 3 | 2 | E8 | 5 | 2 | F8 | 4 | 3 | | | | B⊕M → B | • | • | ↕ | ↕ | R | • |
| Increment | INC | | | | | | | 6C | 7 | 2 | 7C | 6 | 3 | | | | M + 1 → M | • | • | ↕ | ↕ | ⑤ | • |
| | INCA | | | | | | | | | | | | | 4C | 2 | 1 | A + 1 → A | • | • | ↕ | ↕ | ⑤ | • |
| | INCB | | | | | | | | | | | | | 5C | 2 | 1 | B + 1 → B | • | • | ↕ | ↕ | ⑤ | • |
| Load Acmltr | LDAA | 86 | 2 | 2 | 96 | 3 | 2 | A6 | 5 | 2 | B6 | 4 | 3 | | | | M → A | • | • | ↕ | ↕ | R | • |
| | LDAB | C6 | 2 | 2 | D6 | 3 | 2 | E6 | 5 | 2 | F6 | 4 | 3 | | | | M → B | • | • | ↕ | ↕ | R | • |
| Or, Inclusive | ORAA | 8A | 2 | 2 | 9A | 3 | 2 | AA | 5 | 2 | BA | 4 | 3 | | | | A + M → A | • | • | ↕ | ↕ | R | • |
| | ORAB | CA | 2 | 2 | DA | 3 | 2 | EA | 5 | 2 | FA | 4 | 3 | | | | B + M → B | • | • | ↕ | ↕ | R | • |
| Push Data | PSHA | | | | | | | | | | | | | 36 | 4 | 1 | A → $M_{SP}$, SP − 1 → SP | • | • | • | • | • | • |
| | PSHB | | | | | | | | | | | | | 37 | 4 | 1 | B → $M_{SP}$, SP − 1 → SP | • | • | • | • | • | • |
| Pull Data | PULA | | | | | | | | | | | | | 32 | 4 | 1 | SP + 1 → SP, $M_{SP}$ → A | • | • | • | • | • | • |
| | PULB | | | | | | | | | | | | | 33 | 4 | 1 | SP + 1 → SP, $M_{SP}$ → B | • | • | • | • | • | • |
| Rotate Left | ROL | | | | | | | 69 | 7 | 2 | 79 | 6 | 3 | | | | M | • | • | ↕ | ↕ | ⑥ | ↕ |
| | ROLA | | | | | | | | | | | | | 49 | 2 | 1 | A | • | • | ↕ | ↕ | ⑥ | ↕ |
| | ROLB | | | | | | | | | | | | | 59 | 2 | 1 | B — C b7 b0 | • | • | ↕ | ↕ | ⑥ | ↕ |
| Rotate Right | ROR | | | | | | | 66 | 7 | 2 | 76 | 6 | 3 | | | | M | • | • | ↕ | ↕ | ⑥ | ↕ |
| | RORA | | | | | | | | | | | | | 46 | 2 | 1 | A | • | • | ↕ | ↕ | ⑥ | ↕ |
| | RORB | | | | | | | | | | | | | 56 | 2 | 1 | B — C b7 b0 | • | • | ↕ | ↕ | ⑥ | ↕ |
| Shift Left, Arithmetic | ASL | | | | | | | 68 | 7 | 2 | 78 | 6 | 3 | | | | M | • | • | ↕ | ↕ | ⑥ | ↕ |
| | ASLA | | | | | | | | | | | | | 48 | 2 | 1 | A — C b7 b0 — 0 | • | • | ↕ | ↕ | ⑥ | ↕ |
| | ASLB | | | | | | | | | | | | | 58 | 2 | 1 | B | • | • | ↕ | ↕ | ⑥ | ↕ |
| Shift Right, Arithmetic | ASR | | | | | | | 67 | 7 | 2 | 77 | 6 | 3 | | | | M | • | • | ↕ | ↕ | ⑥ | ↕ |
| | ASRA | | | | | | | | | | | | | 47 | 2 | 1 | A — C b7 b0 | • | • | ↕ | ↕ | ⑥ | ↕ |
| | ASRB | | | | | | | | | | | | | 57 | 2 | 1 | B | • | • | ↕ | ↕ | ⑥ | ↕ |
| Shift Right, Logic | LSR | | | | | | | 64 | 7 | 2 | 74 | 6 | 3 | | | | M | • | • | R | ↕ | ⑥ | ↕ |
| | LSRA | | | | | | | | | | | | | 44 | 2 | 1 | A 0 — b7 b0 — C | • | • | R | ↕ | ⑥ | ↕ |
| | LSRB | | | | | | | | | | | | | 54 | 2 | 1 | B | • | • | R | ↕ | ⑥ | ↕ |
| Store Acmltr | STAA | | | | 97 | 4 | 2 | A7 | 6 | 2 | B7 | 5 | 3 | | | | A → M | • | • | ↕ | ↕ | R | • |
| | STAB | | | | D7 | 4 | 2 | E7 | 6 | 2 | F7 | 5 | 3 | | | | B → M | • | • | ↕ | ↕ | R | • |
| Subtract | SUBA | 80 | 2 | 2 | 90 | 3 | 2 | A0 | 5 | 2 | B0 | 4 | 3 | | | | A − M → A | • | • | ↕ | ↕ | ↕ | ↕ |
| | SUBB | C0 | 2 | 2 | D0 | 3 | 2 | E0 | 5 | 2 | F0 | 4 | 3 | | | | B − M → B | • | • | ↕ | ↕ | ↕ | ↕ |
| Subtract Acmltrs | SBA | | | | | | | | | | | | | 10 | 2 | 1 | A − B → A | • | • | ↕ | ↕ | ↕ | ↕ |
| Subtr with Carry | SBCA | 82 | 2 | 2 | 92 | 3 | 2 | A2 | 5 | 2 | B2 | 4 | 3 | | | | A − M − C → A | • | • | ↕ | ↕ | ↕ | ↕ |
| | SBCB | C2 | 2 | 2 | D2 | 3 | 2 | E2 | 5 | 2 | F2 | 4 | 3 | | | | B − M − C → B | • | • | ↕ | ↕ | ↕ | ↕ |
| Transfer Acmltrs | TAB | | | | | | | | | | | | | 16 | 2 | 1 | A → B | • | • | ↕ | ↕ | R | • |
| | TBA | | | | | | | | | | | | | 17 | 2 | 1 | B → A | • | • | ↕ | ↕ | R | • |
| Test, Zero or Minus | TST | | | | | | | 6D | 7 | 2 | 7D | 6 | 3 | | | | M − 00 | • | • | ↕ | ↕ | R | R |
| | TSTA | | | | | | | | | | | | | 4D | 2 | 1 | A − 00 | • | • | ↕ | ↕ | R | R |
| | TSTB | | | | | | | | | | | | | 5D | 2 | 1 | B − 00 | • | • | ↕ | ↕ | R | R |
| | | | | | | | | | | | | | | | | | | H | I | N | Z | V | C |

**LEGEND:**

- OP Operation Code (Hexadecimal).
- ~ Number of MPU Cycles.
- = Number of Program Bytes.
- + Arithmetic Plus.
- − Arithmetic Minus.
- · Boolean AND.
- $M_{SP}$ Contents of memory location pointed to be Stack Pointer.
- + Boolean Inclusive OR.
- ⊕ Boolean Exclusive OR.
- M̄ Complement of M.
- → Transfer Into.
- 0 Bit = Zero.
- 00 Byte = Zero.

Note – Accumulator addressing mode instructions are included in the column for IMPLIED addressing

**CONDITION CODE SYMBOLS:**

- H Half-carry from bit 3.
- I Interrupt mask
- N Negative (sign bit)
- Z Zero (byte)
- V Overflow, 2's complement
- C Carry from bit 7
- R Reset Always
- S Set Always
- ↕ Test and set if true, cleared otherwise
- • Not Affected

87D 09244 DT-49-17-06

## TABLE 4 — INDEX REGISTER AND STACK MANIPULATION INSTRUCTIONS

| POINTER OPERATIONS | MNEMONIC | IMMED OP | IMMED ~ | IMMED # | DIRECT OP | DIRECT ~ | DIRECT # | INDEX OP | INDEX ~ | INDEX # | EXTND OP | EXTND ~ | EXTND # | IMPLIED OP | IMPLIED ~ | IMPLIED # | BOOLEAN/ARITHMETIC OPERATION | 5 H | 4 I | 3 N | 2 Z | 1 V | 0 C |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Compare Index Reg | CPX | 8C | 3 | 3 | 9C | 4 | 2 | AC | 6 | 2 | BC | 5 | 3 | | | | $X_H - M, X_L - (M+1)$ | • | • | ⑦ | ↕ | ⑧ | • |
| Decrement Index Reg | DEX | | | | | | | | | | | | | 09 | 4 | 1 | $X - 1 \rightarrow X$ | • | • | • | ↕ | • | • |
| Decrement Stack Pntr | DES | | | | | | | | | | | | | 34 | 4 | 1 | $SP - 1 \rightarrow SP$ | • | • | • | • | • | • |
| Increment Index Reg | INX | | | | | | | | | | | | | 08 | 4 | 1 | $X + 1 \rightarrow X$ | • | • | • | ↕ | • | • |
| Increment Stack Pntr | INS | | | | | | | | | | | | | 31 | 4 | 1 | $SP + 1 \rightarrow SP$ | • | • | • | • | • | • |
| Load Index Reg | LDX | CE | 3 | 3 | DE | 4 | 2 | EE | 6 | 2 | FE | 5 | 3 | | | | $M \rightarrow X_H, (M+1) \rightarrow X_L$ | • | • | ⑨ | ↕ | R | • |
| Load Stack Pntr | LDS | 8E | 3 | 3 | 9E | 4 | 2 | AE | 6 | 2 | BE | 5 | 3 | | | | $M \rightarrow SP_H, (M+1) \rightarrow SP_L$ | • | • | ⑨ | ↕ | R | • |
| Store Index Reg | STX | | | | DF | 5 | 2 | EF | 7 | 2 | FF | 6 | 3 | | | | $X_H \rightarrow M, X_L \rightarrow (M+1)$ | • | • | ⑨ | ↕ | R | • |
| Store Stack Pntr | STS | | | | 9F | 5 | 2 | AF | 7 | 2 | BF | 6 | 3 | | | | $SP_H \rightarrow M, SP_L \rightarrow (M+1)$ | • | • | ⑨ | ↕ | R | • |
| Indx Reg → Stack Pntr | TXS | | | | | | | | | | | | | 35 | 4 | 1 | $X - 1 \rightarrow SP$ | • | • | • | • | • | • |
| Stack Pntr → Indx Reg | TSX | | | | | | | | | | | | | 30 | 4 | 1 | $SP + 1 \rightarrow X$ | • | • | • | • | • | • |

## TABLE 5 — JUMP AND BRANCH INSTRUCTIONS

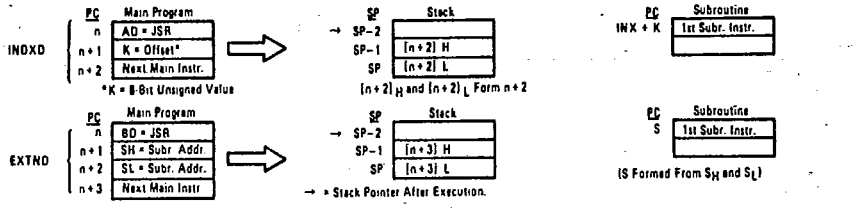
| OPERATIONS | MNEMONIC | RELATIVE OP | RELATIVE ~ | RELATIVE # | INDEX OP | INDEX ~ | INDEX # | EXTND OP | EXTND ~ | EXTND # | IMPLIED OP | IMPLIED ~ | IMPLIED # | BRANCH TEST | 5 H | 4 I | 3 N | 2 Z | 1 V | 0 C |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Branch Always | BRA | 20 | 4 | 2 | | | | | | | | | | None | • | • | • | • | • | • |
| Branch If Carry Clear | BCC | 24 | 4 | 2 | | | | | | | | | | C = 0 | • | • | • | • | • | • |
| Branch If Carry Set | BCS | 25 | 4 | 2 | | | | | | | | | | C = 1 | • | • | • | • | • | • |
| Branch If = Zero | BEQ | 27 | 4 | 2 | | | | | | | | | | Z = 1 | • | • | • | • | • | • |
| Branch If ≥ Zero | BGE | 2C | 4 | 2 | | | | | | | | | | $N \oplus V = 0$ | • | • | • | • | • | • |
| Branch If > Zero | BGT | 2E | 4 | 2 | | | | | | | | | | $Z + (N \oplus V) = 0$ | • | • | • | • | • | • |
| Branch If Higher | BHI | 22 | 4 | 2 | | | | | | | | | | C + Z = 0 | • | • | • | • | • | • |
| Branch If ≤ Zero | BLE | 2F | 4 | 2 | | | | | | | | | | $Z + (N \oplus V) = 1$ | • | • | • | • | • | • |
| Branch If Lower Or Same | BLS | 23 | 4 | 2 | | | | | | | | | | C + Z = 1 | • | • | • | • | • | • |
| Branch If < Zero | BLT | 2D | 4 | 2 | | | | | | | | | | $N \oplus V = 1$ | • | • | • | • | • | • |
| Branch If Minus | BMI | 2B | 4 | 2 | | | | | | | | | | N = 1 | • | • | • | • | • | • |
| Branch If Not Equal Zero | BNE | 26 | 4 | 2 | | | | | | | | | | Z = 0 | • | • | • | • | • | • |
| Branch If Overflow Clear | BVC | 28 | 4 | 2 | | | | | | | | | | V = 0 | • | • | • | • | • | • |
| Branch If Overflow Set | BVS | 29 | 4 | 2 | | | | | | | | | | V = 1 | • | • | • | • | • | • |
| Branch If Plus | BPL | 2A | 4 | 2 | | | | | | | | | | N = 0 | • | • | • | • | • | • |
| Branch To Subroutine | BSR | 8D | 8 | 2 | | | | | | | | | | | • | • | • | • | • | • |
| Jump | JMP | | | | 6E | 4 | 2 | 7E | 3 | 3 | | | | See Special Operations (Figure 16) | • | • | • | • | • | • |
| Jump To Subroutine | JSR | | | | AD | 8 | 2 | BD | 9 | 3 | | | | | • | • | • | • | • | • |
| No Operation | NOP | | | | | | | | | | 01 | 2 | 1 | Advances Prog. Cntr. Only | • | • | • | • | • | • |
| Return From Interrupt | RTI | | | | | | | | | | 3B | 10 | 1 | ⑩ | | | | | | |
| Return From Subroutine | RTS | | | | | | | | | | 39 | 5 | 1 | | • | • | • | • | • | • |
| Software Interrupt | SWI | | | | | | | | | | 3F | 12 | 1 | See Special Operations (Figure 16) | • | ⑪ | • | • | • | • |
| Wait for Interrupt | WAI | | | | | | | | | | 3E | 9 | 1 | | • | • | • | • | • | • |

FIGURE 16 — SPECIAL OPERATIONS

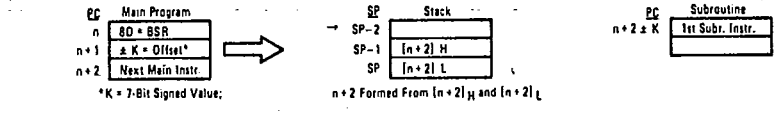
## SPECIAL OPERATIONS

87D 09245 DT-49-17-06

### JSR, JUMP TO SUBROUTINE:

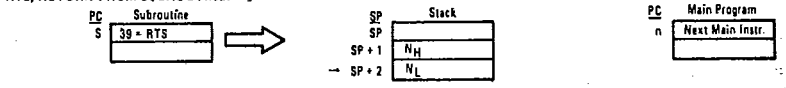



### BSR, BRANCH TO SUBROUTINE:

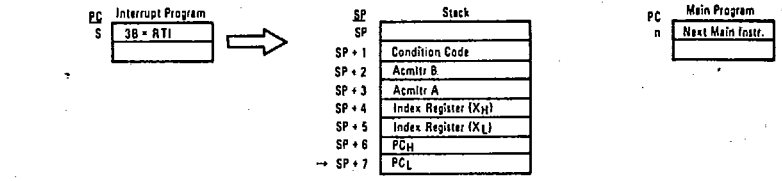



### JMP, JUMP:


### RTS, RETURN FROM SUBROUTINE:




### RTI, RETURN FROM INTERRUPT:




## TABLE 6 — CONDITION CODE REGISTER MANIPULATION INSTRUCTIONS

| OPERATIONS | MNEMONIC | IMPLIED OP | ~ | # | BOOLEAN OPERATION | COND. CODE REG. 5 H | 4 I | 3 N | 2 Z | 1 V | 0 C |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Clear Carry | CLC | 0C | 2 | 1 | 0→C | • | • | • | • | • | R |
| Clear Interrupt Mask | CLI | 0E | 2 | 1 | 0→I | • | R | • | • | • | • |
| Clear Overflow | CLV | 0A | 2 | 1 | 0→V | • | • | • | • | R | • |
| Set Carry | SEC | 0D | 2 | 1 | 1→C | • | • | • | • | • | S |
| Set Interrupt Mask | SEI | 0F | 2 | 1 | 1→I | • | S | • | • | • | • |
| Set Overflow | SEV | 0B | 2 | 1 | 1→V | • | • | • | • | S | • |
| Acmltr A → CCR | TAP | 06 | 2 | 1 | A→CCR | (12) | | | | | |
| CCR → Acmltr A | TPA | 07 | 2 | 1 | CCR→A | • | • | • | • | • | • |

**CONDITION CODE REGISTER NOTES:** (Bit set if test is true and cleared otherwise)

1 (Bit V) Test: Result = 10000000?
2 (Bit C) Test: Result ≠ 00000000?
3 (Bit C) Test: Decimal value of most significant BCD Character greater than nine? (Not cleared if previously set.)
4 (Bit V) Test: Operand = 10000000 prior to execution?
5 (Bit V) Test: Operand = 01111111 prior to execution?
6 (Bit V) Test: Set equal to result of N⊕C after shift has occurred.
7 (Bit N) Test: Sign bit of most significant (MS) byte = 1?
8 (Bit V) Test: 2's complement overflow from subtraction of MS bytes?
9 (Bit N) Test: Result less than zero? (Bit 15 = 1)
10 (All) Load Condition Code Register from Stack. (See Special Operations)
11 (Bit I) Set when interrupt occurs. If previously set, a Non-Maskable Interrupt is required to exit the wait state.
12 (All) Set according to the contents of Accumulator A.

TABLE 7 – INSTRUCTION ADDRESSING MODES AND ASSOCIATED EXECUTION TIMES
(Times in Machine Cycle)

87D 09246 D T-49-17-06

| | (Dual Operand) | ACCX | Immediate | Direct | Extended | Indexed | Implied | Relative |
|---|---|---|---|---|---|---|---|---|
| ABA | | • | • | • | • | • | 2 | • |
| ADC | x | • | 2 | 3 | 4 | 5 | • | • |
| ADD | x | • | 2 | 3 | 4 | 5 | • | • |
| AND | x | • | 2 | 3 | 4 | 5 | • | • |
| ASL | | 2 | • | • | 6 | 7 | • | • |
| ASR | | 2 | • | • | 6 | 7 | • | • |
| BCC | | • | • | • | • | • | • | 4 |
| BCS | | • | • | • | • | • | • | 4 |
| BEA | | • | • | • | • | • | • | 4 |
| BGE | | • | • | • | • | • | • | 4 |
| BGT | | • | • | • | • | • | • | 4 |
| BHI | | • | • | • | • | • | • | 4 |
| BIT | x | • | 2 | 3 | 4 | 5 | • | • |
| BLE | | • | • | • | • | • | • | 4 |
| BLS | | • | • | • | • | • | • | 4 |
| BLT | | • | • | • | • | • | • | 4 |
| BMI | | • | • | • | • | • | • | 4 |
| BNE | | • | • | • | • | • | • | 4 |
| BPL | | • | • | • | • | • | • | 4 |
| BRA | | • | • | • | • | • | • | 4 |
| BSR | | • | • | • | • | • | • | 8 |
| BVC | | • | • | • | • | • | • | 4 |
| BVS | | • | • | • | • | • | • | 4 |
| CBA | | • | • | • | • | • | 2 | • |
| CLC | | • | • | • | • | • | 2 | • |
| CLI | | • | • | • | • | • | 2 | • |
| CLR | | 2 | • | • | 6 | 7 | • | • |
| CLV | | • | • | • | • | • | 2 | • |
| CMP | x | • | 2 | 3 | 4 | 5 | • | • |
| COM | | 2 | • | • | 6 | 7 | • | • |
| CPX | | • | 3 | 4 | 5 | 6 | • | • |
| DAA | | • | • | • | • | • | 2 | • |
| DEC | | 2 | • | • | 6 | 7 | • | • |
| DES | | • | • | • | • | • | 4 | • |
| DEX | | • | • | • | • | • | 4 | • |
| EOR | x | • | 2 | 3 | 4 | 5 | • | • |

| | (Dual Operand) | ACCX | Immediate | Direct | Extended | Indexed | Implied |
|---|---|---|---|---|---|---|---|
| INC | | 2 | • | • | 6 | 7 | • |
| INS | | • | • | • | • | • | 4 |
| INX | | • | • | • | • | • | 4 |
| JMP | | • | • | • | 3 | 4 | • |
| JSR | | • | • | • | 9 | 8 | • |
| LDA | x | • | 2 | 3 | 4 | 5 | • |
| LDS | | • | 3 | 4 | 5 | 6 | • |
| LDX | | • | 3 | 4 | 5 | 6 | • |
| LSR | | 2 | • | • | 6 | 7 | • |
| NEG | | 2 | • | • | 6 | 7 | • |
| NOP | | • | • | • | • | • | 2 |
| ORA | x | • | 2 | 3 | 4 | 5 | • |
| PSH | | • | • | • | • | • | 4 |
| PUL | | • | • | • | • | • | 4 |
| ROL | | 2 | • | • | 6 | 7 | • |
| ROR | | 2 | • | • | 6 | 7 | • |
| RTI | | • | • | • | • | • | 10 |
| RTS | | • | • | • | • | • | 5 |
| SBA | | • | • | • | • | • | 2 |
| SBC | x | • | 2 | 3 | 4 | 5 | • |
| SEC | | • | • | • | • | • | 2 |
| SEI | | • | • | • | • | • | 2 |
| SEV | | • | • | • | • | • | 2 |
| STA | x | • | • | 4 | 5 | 6 | • |
| STS | | • | • | 5 | 6 | 7 | • |
| STX | | • | • | 5 | 6 | 7 | • |
| SUB | x | • | 2 | 3 | 4 | 5 | • |
| SWI | | • | • | • | • | • | 12 |
| TAB | | • | • | • | • | • | 2 |
| TAP | | • | • | • | • | • | 2 |
| TBA | | • | • | • | • | • | 2 |
| TPA | | • | • | • | • | • | 2 |
| TST | | 2 | • | • | 6 | 7 | • |
| TSX | | • | • | • | • | • | 4 |
| TSX | | • | • | • | • | • | 4 |
| WAI | | • | • | • | • | • | 9 |

NOTE: Interrupt time is 12 cycles from the end of the instruction being executed, except following a WAI instruction. Then it is 4 cycles.

## SUMMARY OF CYCLE-BY-CYCLE OPERATION

87D 09247 D T-49-17-06

Table 8 provides a detailed description of the information present on the address bus, data bus, valid memory address line (VMA), and the read/write line (R/$\overline{W}$) during each cycle for each instruction.

This information is useful in comparing actual with expected results during debug of both software and hardware as the control program is executed. The information is categorized in groups according to addressing modes and number of cycles per instruction. (In general, instructions with the same addressing mode and number of cycles execute in the same manner; exceptions are indicated in the table.)

TABLE 8 — OPERATIONS SUMMARY

| Address Mode and Instructions | Cycles | Cycle # | VMA Line | Address Bus | R/$\overline{W}$ Line | Data Bus |
|---|---|---|---|---|---|---|
| **IMMEDIATE** | | | | | | |
| ADC EOR<br>ADD LDA<br>AND ORA<br>BIT SBC<br>CMP SUB | 2 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Operand Data |
| CPX<br>LDS<br>LDX | 3 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Operand Data (High Order Byte) |
| | | 3 | 1 | Op Code Address + 2 | 1 | Operand Data (Low Order Byte) |
| **DIRECT** | | | | | | |
| ADC EOR<br>ADD LDA<br>AND ORA<br>BIT SBC<br>CMP SUB | 3 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Address of Operand |
| | | 3 | 1 | Address of Operand | 1 | Operand Data |
| CPX<br>LDS<br>LDX | 4 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Address of Operand |
| | | 3 | 1 | Address of Operand | 1 | Operand Data (High Order Byte) |
| | | 4 | 1 | Operand Address + 1 | 1 | Operand Data (Low Order Byte) |
| STA | 4 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Destination Address |
| | | 3 | 0 | Destination Address | 1 | Irrelevant Data (Note 1) |
| | | 4 | 1 | Destination Address | 0 | Data from Accumulator |
| STS<br>STX | 5 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Address of Operand |
| | | 3 | 0 | Address of Operand | 1 | Irrelevant Data (Note 1) |
| | | 4 | 1 | Address of Operand | 0 | Register Data (High Order Byte) |
| | | 5 | 1 | Address of Operand + 1 | 0 | Register Data (Low Order Byte) |
| **INDEXED** | | | | | | |
| JMP | 4 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Offset |
| | | 3 | 0 | Index Register | 1 | Irrelevant Data (Note 1) |
| | | 4 | 0 | Index Register Plus Offset (w/o Carry) | 1 | Irrelevant Data (Note 1) |
| ADC EOR<br>ADD LDA<br>AND ORA<br>BIT SBC<br>CMP SUB | 5 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Offset |
| | | 3 | 0 | Index Register | 1 | Irrelevant Data (Note 1) |
| | | 4 | 0 | Index Register Plus Offset (w/o Carry) | 1 | Irrelevant Data (Note 1) |
| | | 5 | 1 | Index Register Plus Offset | 1 | Operand Data |
| CPX<br>LDS<br>LDX | 6 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Offset |
| | | 3 | 0 | Index Register | 1 | Irrelevant Data (Note 1) |
| | | 4 | 0 | Index Register Plus Offset (w/o Carry) | 1 | Irrelevant Data (Note 1) |
| | | 5 | 1 | Index Register Plus Offset | 1 | Operand Data (High Order Byte) |
| | | 6 | 1 | Index Register Plus Offset + 1 | 1 | Operand Data (Low Order Byte) |

TABLE 8 — OPERATIONS SUMMARY (CONTINUED)

87D 09248 D T-49-17-06

| Address Mode and Instructions | Cycles | Cycle # | VMA Line | Address Bus | Line | Data Bus |
|---|---|---|---|---|---|---|
| **INDEXED (Continued)** | | | | | | |
| STA | 6 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Offset |
| | | 3 | 0 | Index Register | 1 | Irrelevant Data (Note 1) |
| | | 4 | 0 | Index Register Plus Offset (w/o Carry) | 1 | Irrelevant Data (Note 1) |
| | | 5 | 0 | Index Register Plus Offset | 1 | Irrelevant Data (Note 1) |
| | | 6 | 1 | Index Register Plus Offset | 0 | Operand Data |
| ASL LSR<br>ASR NEG<br>CLR ROL<br>COM ROR<br>DEC TST<br>INC | 7 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Offset |
| | | 3 | 0 | Index Register | 1 | Irrelevant Data (Note 1) |
| | | 4 | 0 | Index Register Plus Offset (w/o Carry) | 1 | Irrelevant Data (Note 1) |
| | | 5 | 1 | Index Register Plus Offset | 1 | Current Operand Data |
| | | 6 | 0 | Index Register Plus Offset | 1 | Irrelevant Data (Note 1) |
| | | 7 | 1/0 (Note 3) | Index Register Plus Offset | 0 | New Operand Data (Note 3) |
| STS<br>STX | 7 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Offset |
| | | 3 | 0 | Index Register | 1 | Irrelevant Data (Note 1) |
| | | 4 | 0 | Index Register Plus Offset (w/o Carry) | 1 | Irrelevant Data (Note 1) |
| | | 5 | 0 | Index Register Plus Offset | 1 | Irrelevant Data (Note 1) |
| | | 6 | 1 | Index Register Plus Offset | 0 | Operand Data (High Order Byte) |
| | | 7 | 1 | Index Register Plus Offset + 1 | 0 | Operand Data (Low Order Byte) |
| JSR | 8 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Offset |
| | | 3 | 0 | Index Register | 1 | Irrelevant Data (Note 1) |
| | | 4 | 1 | Stack Pointer | 0 | Return Address (Low Order Byte) |
| | | 5 | 1 | Stack Pointer — 1 | 0 | Return Address (High Order Byte) |
| | | 6 | 0 | Stack Pointer — 2 | 1 | Irrelevant Data (Note 1) |
| | | 7 | 0 | Index Register | 1 | Irrelevant Data (Note 1) |
| | | 8 | 0 | Index Register Plus Offset (w/o Carry) | 1 | Irrelevant Data (Note 1) |
| **EXTENDED** | | | | | | |
| JMP | 3 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Jump Address (High Order Byte) |
| | | 3 | 1 | Op Code Address + 2 | 1 | Jump Address (Low Order Byte) |
| ADC EOR<br>ADD LDA<br>AND ORA<br>BIT SBC<br>CMP SUB | 4 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Address of Operand (High Order Byte) |
| | | 3 | 1 | Op Code Address + 2 | 1 | Address of Operand (Low Order Byte) |
| | | 4 | 1 | Address of Operand | 1 | Operand Data |
| CPX<br>LDS<br>LDX | 5 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Address of Operand (High Order Byte) |
| | | 3 | 1 | Op Code Address + 2 | 1 | Address of Operand (Low Order Byte) |
| | | 4 | 1 | Address of Operand | 1 | Operand Data (High Order Byte) |
| | | 5 | 1 | Address of Operand + 1 | 1 | Operand Data (Low Order Byte) |
| STA A<br>STA B | 5 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Destination Address (High Order Byte) |
| | | 3 | 1 | Op Code Address + 2 | 1 | Destination Address (Low Order Byte) |
| | | 4 | 0 | Operand Destination Address | 1 | Irrelevant Data (Note 1) |
| | | 5 | 1 | Operand Destination Address | 0 | Data from Accumulator |
| ASL LSR<br>ASR NEG<br>CLR ROL<br>COM ROR<br>DEC TST<br>INC | 6 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Address of Operand (High Order Byte) |
| | | 3 | 1 | Op Code Address + 2 | 1 | Address of Operand (Low Order Byte) |
| | | 4 | 1 | Address of Operand | 1 | Current Operand Data |
| | | 5 | 0 | Address of Operand | 1 | Irrelevant Data (Note 1) |
| | | 6 | 1/0 (Note 3) | Address of Operand | 0 | New Operand Data (Note 3) |

TABLE 8 — OPERATIONS SUMMARY (CONTINUED)

| Address Mode and Instructions | Cycles | Cycle # | VMA Line | Address Bus | R/W Line | Data Bus |
|---|---|---|---|---|---|---|
| **EXTENDED (Continued)** | | | | | | |
| STS<br>STX | 6 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Address of Operand (High Order Byte) |
| | | 3 | 1 | Op Code Address + 2 | 1 | Address of Operand (Low Order Byte) |
| | | 4 | 0 | Address of Operand | 1 | Irrelevant Data (Note 1) |
| | | 5 | 1 | Address of Operand | 0 | Operand Data (High Order Byte) |
| | | 6 | 1 | Address of Operand + 1 | 0 | Operand Data (Low Order Byte) |
| JSR | 9 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Address of Subroutine (High Order Byte) |
| | | 3 | 1 | Op Code Address + 2 | 1 | Address of Subroutine (Low Order Byte) |
| | | 4 | 1 | Subroutine Starting Address | 1 | Op Code of Next Instruction |
| | | 5 | 1 | Stack Pointer | 0 | Return Address (Low Order Byte) |
| | | 6 | 1 | Stack Pointer — 1 | 0 | Return Address (High Order Byte) |
| | | 7 | 0 | Stack Pointer — 2 | 1 | Irrelevant Data (Note 1) |
| | | 8 | 0 | Op Code Address + 2 | 1 | Irrelevant Data (Note 1) |
| | | 9 | 1 | Op Code Address + 2 | 1 | Address of Subroutine (Low Order Byte) |
| **INHERENT** | | | | | | |
| ABA DAA SEC<br>ASL DEC SEI<br>ASR INC SEV<br>CBA LSR TAB<br>CLC NEG TAP<br>CLI NOP T8A<br>CLR ROL TPA<br>CLV ROR TST<br>COM SBA | 2 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Op Code of Next Instruction |
| DES<br>DEX<br>INS<br>INX | 4 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Op Code of Next Instruction |
| | | 3 | 0 | Previous Register Contents | 1 | Irrelevant Data (Note 1) |
| | | 4 | 0 | New Register Contents | 1 | Irrelevant Data (Note 1) |
| PSH | 4 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Op Code of Next Instruction |
| | | 3 | 1 | Stack Pointer | 0 | Accumulator Data |
| | | 4 | 0 | Stack Pointer — 1 | 1 | Accumulator Data |
| PUL | 4 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Op Code of Next Instruction |
| | | 3 | 0 | Stack Pointer | 1 | Irrelevant Data (Note 1) |
| | | 4 | 1 | Stack Pointer + 1 | 1 | Operand Data from Stack |
| TSX | 4 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Op Code of Next Instruction |
| | | 3 | 0 | Stack Pointer | 1 | Irrelevant Data (Note 1) |
| | | 4 | 0 | New Index Register | 1 | Irrelevant Data (Note 1) |
| TXS | 4 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Op Code of Next Instruction |
| | | 3 | 0 | Index Register | 1 | Irrelevant Data |
| | | 4 | 0 | New Stack Pointer | 1 | Irrelevant Data |
| RTS | 5 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Irrelevant Data (Note 2) |
| | | 3 | 0 | Stack Pointer | 1 | Irrelevant Data (Note 1) |
| | | 4 | 1 | Stack Pointer + 1 | 1 | Address of Next Instruction (High Order Byte) |
| | | 5 | 1 | Stack Pointer + 2 | 1 | Address of Next Instruction (Low Order Byte) |

TABLE 8 — OPERATIONS SUMMARY (CONCLUDED) T-49-17-06

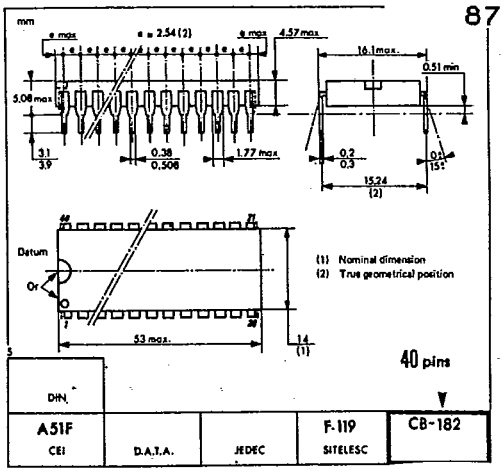
| Address Mode and Instructions | Cycles | Cycle # | VMA Line | Address Bus | R/W Line | Data Bus |
|---|---|---|---|---|---|---|
| **INHERENT (Continued)** | | | | | | |
| WAI | 9 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Op Code of Next Instruction |
| | | 3 | 1 | Stack Pointer | 0 | Return Address (Low Order Byte) |
| | | 4 | 1 | Stack Pointer – 1 | 0 | Return Address (High Order Byte) |
| | | 5 | 1 | Stack Pointer – 2 | 0 | Index Register (Low Order Byte) |
| | | 6 | 1 | Stack Pointer – 3 | 0 | Index Register (High Order Byte) |
| | | 7 | 1 | Stack Pointer – 4 | 0 | Contents of Accumulator A |
| | | 8 | 1 | Stack Pointer – 5 | 0 | Contents of Accumulator B |
| | | 9 | 1 | Stack Pointer – 6 | 1 | Contents of Cond. Code Register |
| RTI | 10 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Irrelevant Data (Note 2) |
| | | 3 | 0 | Stack Pointer | 1 | Irrelevant Data (Note 1) |
| | | 4 | 1 | Stack Pointer + 1 | 1 | Contents of Cond. Code Register from Stack |
| | | 5 | 1 | Stack Pointer + 2 | 1 | Contents of Accumulator B from Stack |
| | | 6 | 1 | Stack Pointer + 3 | 1 | Contents of Accumulator A from Stack |
| | | 7 | 1 | Stack Pointer + 4 | 1 | Index Register from Stack (High Order Byte) |
| | | 8 | 1 | Stack Pointer + 5 | 1 | Index Register from Stack (Low Order Byte) |
| | | 9 | 1 | Stack Pointer + 6 | 1 | Next Instruction Address from Stack (High Order Byte) |
| | | 10 | 1 | Stack Pointer + 7 | 1 | Next Instruction Address from Stack (Low Order Byte) |
| SWI | 12 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Irrelevant Data (Note 1) |
| | | 3 | 1 | Stack Pointer | 0 | Return Address (Low Order Byte) |
| | | 4 | 1 | Stack Pointer – 1 | 0 | Return Address (High Order Byte) |
| | | 5 | 1 | Stack Pointer – 2 | 0 | Index Register (Low Order Byte) |
| | | 6 | 1 | Stack Pointer – 3 | 0 | Index Register (High Order Byte) |
| | | 7 | 1 | Stack Pointer – 4 | 0 | Contents of Accumulator A |
| | | 8 | 1 | Stack Pointer – 5 | 0 | Contents of Accumulator B |
| | | 9 | 1 | Stack Pointer – 6 | 0 | Contents of Cond. Code Register |
| | | 10 | 0 | Stack Pointer – 7 | 1 | Irrelevant Data (Note 1) |
| | | 11 | 1 | Vector Address FFFA (Hex) | 1 | Address of Subroutine (High Order Byte) |
| | | 12 | 1 | Vector Address FFFB (Hex) | 1 | Address of Subroutine (Low Order Byte) |
| **RELATIVE** | | | | | | |
| BCC BHI BNE<br>BCS BLE BPL<br>BEQ BLS BRA<br>BGE BLT BVC<br>BGT BMI BVS | 4 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Branch Offset |
| | | 3 | 0 | Op Code Address + 2 | 1 | Irrelevant Data (Note 1) |
| | | 4 | 0 | Branch Address | 1 | Irrelevant Data (Note 1) |
| BSR | 8 | 1 | 1 | Op Code Address | 1 | Op Code |
| | | 2 | 1 | Op Code Address + 1 | 1 | Branch Offset |
| | | 3 | 0 | Return Address of Main Program | 1 | Irrelevant Data (Note 1) |
| | | 4 | 1 | Stack Pointer | 0 | Return Address (Low Order Byte) |
| | | 5 | 1 | Stack Pointer – 1 | 0 | Return Address (High Order Byte) |
| | | 6 | 0 | Stack Pointer – 2 | 1 | Irrelevant Data (Note 1) |
| | | 7 | 0 | Return Address of Main Program | 1 | Irrelevant Data (Note 1) |
| | | 8 | 0 | Subroutine Address (Note 4) | 1 | Irrelevant Data (Note 1) |

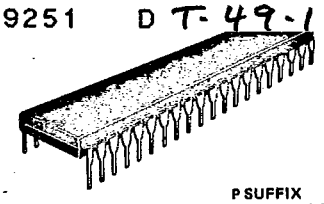
NOTES:
1. If device which is addressed during this cycle uses VMA, then the Data Bus will go to the high-impedance three-state condition. Depending on bus capacitance, data from the previous cycle may be retained on the Data Bus.
2. Data is ignored by the MPU.
3. For TST, VMA = 0 and Operand data does not change.
4. MS Byte of Address Bus = MS Byte of Address of BSR instruction and LS Byte of Address Bus = LS Byte of Sub-Routine Address.

## PHYSICAL DIMENSIONS

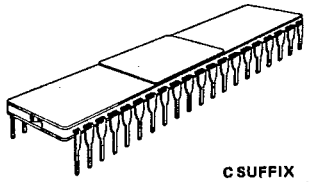
CB-182

87D 09251 D T-49-17-06



P SUFFIX
PLASTIC PACKAGE



C SUFFIX
CERAMIC PACKAGE

## ORDERING INFORMATION

EF68A02 | C | M | B/B

- EF68A02: Device
- C: Package
- M: Oper. temp.
- B/B: Screening level

The table below horizontally shows all available suffix combinations for package, operating temperature and screening level. Other possibilities on request.

| DEVICE | PACKAGE | | | | | OPER. TEMP | | | SCREENING LEVEL | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | C | J | P | E | FN | L* | V | M | Std | D | G/B | B/B |
| EF6802/06 (1.0 MHz) | ● | | ● | | ● | ● | | | ● | | | |
| | ● | | ● | | | | ● | | ● | | | |
| | | | | ● | | | | ● | ● | | ● | ● |
| EF68A02/A06 (1.5 MHz) | ● | | ● | | | ● | | | ● | | | |
| | ● | | ● | | | | ● | | ● | | | |
| | ● | | | ● | | | | ● | ● | | ● | ● |
| EF68B02/B06 (2.0 MHz) | ● | | ● | | | ● | | | ● | | | |
| | ● | | | | | | ● | | ● | | ● | |

Examples : EF6802C, EF6802CV, EF6802EM, EF6802EM G/B

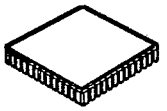
Package : C : Ceramic DIL, J : Cerdip DIL, P : Plastic DIL, E : LCCC, FN : PLCC.
Oper. temp. : L* : 0°C to +70°C, V : −40°C to +85°C, M : −55°C to +125°C, * : may be omitted.
Screening level : Std : (no-end suffix), D : NFC 96883 level D,
G/B : NFC 96883 level G, B/B : NFC 96883 level B and MIL-STD-883C level B.

87D 09252 D T-49-17-06

**CB-521**

**FN SUFFIX**
**PLCC 44**

**CB-708**



**E SUFFIX**
**LCCC 44**