## MICROPROCESSOR WITH CLOCK AND OPTIONAL RAM

The MC6802 is a monolithic 8-bit microprocessor that contains all the registers and accumulators of the present MC6800 plus an internal clock oscillator and driver on the same chip. In addition, the MC6802 has 128 bytes of on-board RAM located at hex addresses $\$ 0000$ to $\$ 007 \mathrm{~F}$. The first 32 bytes of RAM, at hex addresses $\$ 0000$ to $\$ 001 \mathrm{~F}$, may be retained in a low power mode by utilizing $V_{C C}$ standby; thus, facilitating memory retention during a power-down situation.

The MC6802 is completely software compatible with the MC6800 as well as the entire M6800 family of parts. Hence, the MC6802 is expandable to 64 K words.

The MC6802NS is identical to the MC6802 without standby RAM feature. The MC6808 is identical to the MC6802 without on-board RAM.

- On-Chip Clock Circuit
- $128 \times 8$ Bit On-Chip RAM
- 32 Bytes of RAM are Retainable
- Software-Compatible with the MC6800
- Expandable to 64 K Words
- Standard TTL-Compatible Inputs and Outputs
- 8-Bit Word Size
- 16-Bit Memory Addressing
- Interrupt Capability

PART NUMBER DESIGNATION BY SPEED



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 M6800 Microcomputer family.

## MC6802 MC6808 MC6808



PIN ASSIGNMENT

*Pin 35 must be tied to 5 V on the 6802NS
**Pin 36 must be tied to ground for the 6808

MAXIMUM RATINGS

| Rating | Symbol | Value | Unit |
| :---: | :---: | :---: | :---: |
| Supply Voltage, , ${ }^{\text {a }}$, | VCC | -0.3 to +70 | V |
| Input Voltage, $\quad$, | $V_{\text {in, }}$ | -03 to +70 | V |
| Operating Temperature Range | TTA | 0 to +70 | ${ }^{2}$ |
| Storage Temperature Range, , , , , | T stg | -55 to +150 | C |

THERMAL CHARACTERISTICS

| Characteristic | Symbol | Value | Unit |
| :--- | :---: | :---: | :---: |
| Average Thermal Resistance (Junction to Ambient) |  |  |  |
| Plastic |  | 100, | ${ }^{\circ} \mathrm{C} / \mathrm{W}$ |
| Ceramic |  | $\theta_{J A}$ | 50, |

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 highimpedance circuit, Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level le.g., either VSS or VCCl .

## POWER CONSIDERATIONS

The average chip-junction temperature, $T_{J}$, in ${ }^{\circ} \mathrm{C}$ can be obtained from:

$$
\begin{equation*}
T J=T_{A}+\left(P D^{\bullet} \theta J A\right) \tag{1}
\end{equation*}
$$

Where:

$$
\begin{aligned}
& T_{A}=\text { Ambient Temperature, }{ }^{\circ} \mathrm{C} \\
& \theta \mathrm{JA}=\text { Package Thermal Resistance, Junction-to-Ambient, }{ }^{\circ} \mathrm{C} / \mathrm{W} \\
& P D=P I N T+P P O R T \\
& \text { PINT }=I C C \times V C C, \text { Watts } ~=~ C h i p ~ I n t e r n a l ~ P o w e r ~ \\
& \text { PPORT = Port Power Dissipation, Watts - User Determined }
\end{aligned}
$$

For most applications PPORT \&PINT and can be neglected. PPORT may become significant if the device is configured to drive Darlington bases or sink LED loads.
An approximate relationship between PD and TU ( if $P P O R T$ is neglected) is:

$$
\begin{equation*}
P_{D}=K-\left(T J+273^{\circ} \mathrm{C}\right) \tag{2}
\end{equation*}
$$

Solving equations 1 and 2 for $K$ gives:

$$
\begin{equation*}
K=P D \cdot\left(T A+273^{\circ} \mathrm{C}\right)+\theta J A^{\bullet} \cdot \mathrm{PD}^{2} \tag{3}
\end{equation*}
$$

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 TA. 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 $T^{A}$.

OPERATING TEMPERATURE RANGE

| Device | Speed | Symbol | Value | Unit |
| :---: | :---: | :---: | :---: | :---: |
| $\begin{aligned} & \text { MC6802P,L } \\ & \text { MC6802CP,CL } \end{aligned}$ | $\begin{aligned} & \text { (1.0 MHz) } \\ & (1.0 \mathrm{MHz}) \\ & \hline \end{aligned}$ | TA | $\begin{gathered} 0 \text { to }+70 \\ -40 \text { to }+85 \end{gathered}$ | ${ }^{\circ} \mathrm{C}$ |
| MC68A02P,L MC68A02CP,CL | $\begin{aligned} & \text { (1.5 MHz) } \\ & \text { (1.5 MHz) } \end{aligned}$ | TA | $\begin{gathered} 0 \text { to }+70 \\ -40 \text { to }+85 \end{gathered}$ | ${ }^{\circ} \mathrm{C}$ |
| $\begin{aligned} & \text { MC68B02P,L } \\ & \text { MC68B02CP,CL } \end{aligned}$ | $\begin{aligned} & (2.0 \mathrm{MHz}) \\ & (2.0 \mathrm{MHz}) \end{aligned}$ | ${ }^{1}$ A | $\begin{gathered} 0 \text { to }+70 \\ -40 \text { to }+85 \end{gathered}$ | ${ }^{\circ} \mathrm{C}$ |
| MC6802NSP,L | $(1.0 \mathrm{MHz})$ | TA | 0 to +70 | ${ }^{\circ} \mathrm{C}$ |
| MC6808P,L MC68A08P,L MC68B08P,L | $\begin{aligned} & \hline(1.0 \mathrm{MHz}) \\ & (1.5 \mathrm{MHz}) \\ & (2.0 \mathrm{MHz}) \\ & \hline \end{aligned}$ | TA | $0 \text { to }+70$ | ${ }^{\circ} \mathrm{C}$ |

DC ELECTRICAL CHARACTERISTICS $\left(V_{C C}=5.0 \mathrm{Vdc} \pm 5 \%, V_{S S}=0, T_{A}=0\right.$ to $70^{\circ} \mathrm{C}$, unless otherwise noted)

*In power-down mode, maximura powerdissipation is less than 42 mW .
\#Capacitances are periodically sampled rather than $100 \%$ tested.

CONTROL TIMING ${ }^{W} C C=5.0 \mathrm{~V} \pm 5 \%, V_{S S}=0, T_{A}=T_{L}$ to $T_{H}$, unless otherwise noted)

| + ${ }^{\text {characteristics }}$ | Symbol | MC6 | $\begin{aligned} & \hline 02 \mathrm{NS}, \\ & 3808 \end{aligned}$ | MC6 | $\begin{aligned} & 3 \mathrm{~A} 02 \\ & 3 \mathrm{~A} \end{aligned}$ |  | $\begin{aligned} & 3 \mathrm{BO2} \\ & 3 \mathrm{BO} \end{aligned}$ | Unit |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  | Min | Max | Min | Max | Min | Max |  |
| Frequency of Operation | $\mathrm{f}_{0}$ | 0.1 | 1.0 | 0.1 | 1.5 | 0.1 | 2.0 | MHz |
| Crystâ Frequency | fxtal | 1.0 | 4.0 | 1.0 | 6.0 | 1.0 | 8.0 | MHz |
| External Oscillator Frequency | $4 \mathrm{xfo}_{0}$ | 0.4 | 4.0 | 0.4 | 6.0 | 0.4 | 8.0 | MHz |
| Crystal Oscillator Start Up Time | trc | 100 | - | 100 | - | 100 | - | ms |
| Processor Controls (HALT, MR, RE, $\overline{\mathrm{RESET}}$, $\overline{\mathrm{R} \mathrm{C}}$ NMM) |  |  |  |  |  |  |  |  |
| - Processor Control Setup Time | ${ }^{\text {tPCS }}$ | 200 | - | 140 | - | 110 | - |  |
| Processor Control Rise and Fall Time (Does Not Apply to RESET) | ${ }^{\mathrm{tPCR}}$, tpCf | - | 100 | - | 100 | - | 100 | ns |

## BUS TIMING CHARACTERISTICS

| Ident. <br> Number | Characteristic | Symbol | MC6802NSMC6802MC6808 |  | $\begin{aligned} & \text { MC68A02 } \\ & \text { MC68A08 } \end{aligned}$ |  | $\begin{aligned} & \text { MC68B02 } \\ & \text { MC68B08 } \end{aligned}$ |  | Unit |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  | Min | Max | Min | Max | Min | Max |  |
| 1 | Cycle Time | teyc | 1.0 | 10 | 0.667 | 10 | 0.5 | 10 | $\mu \mathrm{S}$ |
| 2 | Pulse Width, E Low, | PWEL | 450 | 5000 | 280 | 5000 | 210 | 5000 | ns |
| 3 | Pulse Width, EHigh | PWEH | 450 | 9500 | 280 | 9700 | 220 | 9700 | ns |
| 4 | Clock Rise and Fall Time, , , , , , , , \% | $\mathrm{tr}_{\mathrm{r}, \mathrm{tf}}$ | - | 25 | - | 25. | - | 20 | ns |
| 9 | Address Hold Time | tAH | 20 | - | 20 | - | 20 |  | ns |
| 12 | Non-Muxed Address Valid Time to E (See Note 5) | tAV1 | 160 | $270$ | 100 | - | 50 |  | ns |
| 17 | Read Data Setup Time, , , , , | tDSR | 100 | - | 70 | - | 60 | , | ns |
| 18 | Read Data Hold Time | tDHR | 10 | - | 10 |  | 10 | - | ns |
| 19 | Write Data Delay Time | todw | - | 225 | - | 170 | U | 160 | ns |
| 21 | Write Data Hold Time | tDHW | 30 | - | 20 |  | 20 | - | ns |
| 29 | Usable Access Time (See Note 4 ) | tACC | 605 | - | 310 | - | 235 | - | ns |

FIGURE 2 - BUS TIMING


NOTES:

1. Voltage levels shown are $V_{L} \leq 0.4 \mathrm{~V}, V_{H} \geq 2.4 \mathrm{~V}$, unless otherwise specified.
2. Measurement points shown are 0.8 V and 20 V , unless otherwise noted.
3. All electricals shown for the MC6802 apply to the MC6802NS and MC6808, unless otherwise noted.
4. Usable access time is computed by $12+3+4-17$.

5月tprograms 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 (MC68A02, MC68A08, MC68B02, MC68B08) On-board RAM can be used for data storage with all parts.

FIGURE 3 - BUS TIMING TEST LOAD


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

$V_{C C}=\operatorname{Pin} 8$
$V_{C C}=\operatorname{Pin} 35$ for MC6802NS
$V_{S S}=$ Pins 1, 21
$V_{S S}=\operatorname{Pin} 36$ for MC6808


MOTOROLA Semiconductor Products Inc.

## MPU REGISTERS

A general block diagram of the MC6802 is shown in Figure 6. As shown, the number and configuration of the registers are the same as for the MC6800. The $128 \times 8$-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 MC6802NS is identical to the MC6802 except for the standby feature on the first 32 bytes of RAM. The standby feature does not exist on the MC6802NS and thus pin 35 must be tied to 5 V .

The MC6808 is identical to the MC6802 except for on board RAM. Since the MC6808 does not have on-board RAM pin 36 must be tied to ground allowing the processor to utilize up to 64 K 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 pushdown/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 (1). 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.

[^0]FIGURE 7 - PROGRAMMING MODEL OF THE MICROPROCESSING UNIT


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

```
SP = Stack Pointer
\(\mathrm{CC}=\) Condition Codes (Also called the Processor Status Byte) \(A C C B=\) Accumulator \(B\) \(A C C A=\) Accumulator \(A\)
IXH = Index Register, Higher Order 8 Bits
\(1 \times L=\) 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 MC6800 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)
VCC 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 (DO-D7)

Eight pins are used for the data bus. It is bidirectional, transferring data to and from the memory and peripheral deviees, 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 $\$ 007 \mathrm{~F}$. External RAM at $\$ 0000$ to $\$ 007 \mathrm{~F}$ must be disabled when internal RAM is accessed.

## HALT

When this input is in the low state, all activity in the machine will be halted. This input is level sensitive. In the HALT mode, the machine will stop at the end of an instruc-
tion, 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 HALT line must occur tpCS before the falling edge of $E$ and the HALT line must go high for one clock cycle.
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/ $\bar{W}$ )

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 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 $1=0$ ) or nonmaskable interrupt. This output is capable of driving one standard TTL load and 30 pF .

## INTERRUPT REQUEST (IRQ)

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 SFFF8 and SFFF9 is loaded which causes the MPU to branch to an interrupt routine in memory.

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

A nominal $3 \mathrm{k} \Omega$ pullup resistor to VCC should be used for wire-OR and optimum control of interrupts. TRQ may be tied directly to VCC if not used.

## RESET

This input is used to reset and start the MPU from a power-down condition, resulting from a power failure or an initial startup 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 execu-
tion 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, SFFFF) 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 $\overline{R Q}$. Power-up and reset timing and powerdown sequences are shown in Figures 9 and 10, respectively.

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 $t_{r c}$ power-up feset that is required.

When RESET is released it must go through the low-tohigh threshold without bouncing, oscillating, or otherwise causing an erroneous reset (less than three clock cycles). This may cause improper MPU opetation until the next valid reset.

## NON-MASKABLE INTERRUPT (NMI)

A low-going edge on this input requests that a nonmaskable 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 NMI signal. The interrupt mask bit in the condition code register has no effect on $\overline{N M M}$.

The index register, program counter, accumulators, and condifion 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 $3 \mathrm{k} \Omega$ pullup resistor to $V_{C C}$ should be used for wire-OR and optimum control of interrupts. NMI may be tied


NOTE If option 1 is chosen, RESET and RE pins can be tied together:
directly to $V_{C C}$ if not used
Inputs $\overline{\mathrm{RO}}$ and $\overline{\mathrm{NMI}}$ 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 | Restart |
| \$FFFE | \$FFFF | R |
| \$FFFC | \$FFFD | Non-Maskable Interrupt |
| \$FFFA | \$FFFB | Software Interrupt |
| \$FFF8 | \$FFF9 | Interrupt Request |

FIGURE 10 - POWER-DOWN SEQUENCE



FIGURE 12 - CRYSTAL SPECIFICATIONS


| Y 1 | $\mathrm{C}_{\mathrm{in}}$ | $\mathrm{C}_{\text {out }}$ |
| :---: | :---: | :---: |
| 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 |



Nominal Crystal Parameters*

|  | 3.58 MHz | 4.0 MHz | 6.0 MHz | 8.0 MHz |
| :---: | :---: | :---: | :---: | :---: |
| $\mathrm{R}_{\mathrm{S}}$ | $60 \Omega$ | $50 \Omega$ | $30-50 \Omega$ | $20-40 \Omega$ |
| C 0 | 3.5 pF | 6.5 pF | 46 pF | 4.6 pF |
| Cl | 0.015 pF | 0.025 pF | 0.010 .02 pF | $0.01-0.02 \mathrm{pF}$ |
| O | $>40 \mathrm{~K}$ | $>30 \mathrm{~K}$ | $>20 \mathrm{~K}$ | $>20 \mathrm{~K}$ |

*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



FIGURE 15 - MR NEGATIVE SETUP TIME REQUIREMENT

> E Clock Streteh


The E clock will be stretched at end of $E$ high of the cycle during which MR negative meets the tPCS setup time. The tPCS setup time is referenced to the fall of E . If the tPCS setup time is not met, E will be stretched at the end of the next E -high $1 / 2$ cycle. E will be stretched in integral multiples of $1 / 2$ cycles.


The E clock will resume normal operation at the end of the $1 / 2$ cycle during which MR assertion meets the tPCS setup time. The tpCS setup time is referenced to transitions of $E$ were it not stretched. If tPCS 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 tpCS references occur, unless the synchronizing circuit of Figure 14 is used.

## RAM ENABLE (RE - MC6802 + MC6802NS ONLY)

A TTL-compatible RAM enable input controls the on-chip RAM of the MC6802. When placed in the high state, the onchip 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 VCC goes below $4.75 \vee$ during power-down. RAM enable must be tied low on the MC6808. 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 tPW $\phi$ L The MC6802, MC6808 and MC6802NS 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 stret ching of $E$. Use of MR requires synchronization with the $4 \times f_{0}$ signal, as shown in Figure 14. When MR is high, Ewill be in normal operation. When MR is low, E will be stretohed integral numbers of half periods, thus allowing interface to slow memories. Memory Ready timing is shown in Figure 15.
MR should be tied high lconnected directly to Vccl if not used. This is necessary to ensure proper operation of the part. A maximum stretch is toyc.

## ENABLE (E)

This pin supplies the clock for the MPU and the rest of the system. This is a single-phase, TTL-compatible clock. This clock may be conditiored by a memory read signal. This is equivalent to $\phi 2$ on the MC6800. This output is capable of driving one standard TTL load and 130 pF .

## VCC STANDBY (MC6B02 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 $V_{S B}$ maximum is ISBB. For the MC6802NS this pin must be connected to $V \mathrm{CC}$.

## MPU INSTRUCTION SET

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 MC6800.

## 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.

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

TABLE 3 - ACCUMULATOR AND MEMORY INSTRUCTIONS


[^1]CONDITION CODE SYMBOLS
H Half-cary from bit 3 ;
Interrupt mask
Negalive (sign bit)
Zero (byte)
Overllow, 2's complement
Carry from bit 7
Reset Always
Set Always:
Test and set if trie, cleared otherwise

- Not Affected

TABLE 4 - INDEX REGISTER AND STACK MANIPULATION INSTRUCTIONS

| POINTER OPERATIONS | MNEMONIC | IMMED |  |  | DIRECT |  |  | INDEX |  |  | EXTND |  |  | IMPLIED |  |  | BOOLEAN/ARITHMETIC OPERATION |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  | OP | $\sim$ | $\pm$ | OP | $\sim$ | $=$ | DP | $\sim$ | $=$ | OP | $\sim$ | $=$ | OP | $\sim$ | $=$ |  | H | 1 | N | 2 |  | $v$ | C |
| Compare Index Heg. | CPX | 8 C | 3 | 3 | 9C | 4 | 2 | AC | 6 | 2 | EC | 5 | 3 |  |  |  | $X_{H}-M, X_{L}-(M+1)$ | - | - |  |  |  |  | - |
| Decrement Index $\mathrm{Reg}^{\text {g }}$ | DEX |  |  |  |  |  |  |  |  |  |  |  |  | 09 | 4 | 1 | $x^{+1}-1 \rightarrow x^{\text {a }}$ | - | - | - |  |  |  | - |
| Decrement Stack Pntr | DES |  |  |  |  |  |  |  |  |  |  |  |  | 34 | 4 | 1 | SP-1 ${ }^{\text {P }}$ S $P$ | - | - |  |  |  |  | - |
| Increment Index Reg | INX |  |  |  |  |  |  |  |  |  |  |  |  | 08 | 4 | 1 | $x+1 \rightarrow x$ | - |  |  |  |  |  | - |
| Increment Stack Potr | INS |  |  |  |  |  |  |  |  |  |  |  |  | 31 | 4 | 1 | SP $+1 \rightarrow \mathrm{SP}$ | - |  | - |  |  |  | - |
| 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 | 8 E | 3 | 3. | 9 E | 4 | 2 | AE | 6 | 2 | BE | 5 | 3 |  |  |  |  | - |  | (9) |  |  | R | - |
| Store Index Reg | STX |  |  |  | DF | 5 | 2 | EF | 7 | 2 | FF | 6 | 3 |  |  |  | $X_{H} \cdots M, X_{L} \rightarrow(M+1)$ | - |  | (9) |  |  | R | - |
| Store Stack Pntr | STS |  |  |  | 9 F | 5 | 2 | AF | 7 | 2 | BF | 6 | 3 |  |  |  | $S P_{H} \rightarrow M, S P_{L} \rightarrow(M+1)$ | - |  | (9) |  |  | R | - |
| Indx Reg $\rightarrow$ Stack Pritr | TXS |  |  |  |  |  |  |  |  |  |  |  |  | 35 | 4 | 1 | $\mathrm{X}-1 \rightarrow \mathrm{SP} \quad$ - | - | - | - |  |  | - | - |
| Stack Pntr $\rightarrow$ Indx Reg | TSX |  |  |  |  |  |  |  |  |  |  |  |  | 30 | 4 | 1 | SP + $1 \rightarrow \mathrm{X}$ | - |  | - |  |  | - |  |

## TABLE 5 - JUMP AND BRANCH INSTRUCTIONS

| OPERATIONS | mnemonic | relative |  |  | imatex |  |  | ExtNo |  |  | IMPLIED |  |  |  | BRANCH TEST | cond. Code reg. |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  |  |  |  | 5 | 4 |  |  |  | 2 | 1 | 0 |  |
|  |  | 0 P | - | \# |  |  |  | op | $\sim$ | \# |  |  |  | OP |  | $\sim$ | \# | OP | $\sim$ | $\#$ |  | H | 1 | $z$ | $\checkmark$ | c |
| Branch Always | BRA | 20 | 4 | 2 |  |  |  |  |  |  |  |  |  |  | None | - |  | - | - | $\bullet$ |
| Branch If Carry Clear | 8cc | 24 | 4 | 2 |  |  |  |  |  |  |  |  |  |  | $\mathrm{c}=0$ | - | - | - | - | - |
| Branch If Carry Set | BCS | 25 | 4 | 2 |  |  |  |  |  |  |  | - |  |  | $\mathrm{c}=1$ | - | - | - | - | - |
| Branch If $=$ Zero | beg | 27 | 4 | 2 |  |  |  |  |  |  |  | - |  |  | Z $=1$ | - | - | - | - | - |
| Branch If $\geqslant$ Zero | bge | 2 C | 4 | 2 |  |  |  |  |  |  |  |  |  |  | $N \oplus \mathrm{~V}=0$ | - | - | - | - | - |
| Branch If $>$ Zero | bGt | 2 E | 4 | 2 |  |  |  |  |  |  |  |  |  |  | $\mathrm{Z}+(\mathrm{N} \oplus \mathrm{V})=0$ | - | - | - | - | - |
| Branch If Higher | BHI | 22 | 4 | 2 |  |  |  |  |  |  |  |  |  |  | $\mathrm{C}+\mathrm{z}=0$ | - | - | - | - | - |
| Branch If $\leqslant$ Zero | bLE | 2 F | 4 | 2 |  |  |  |  |  |  |  |  |  |  | $z+(N \oplus V)=1$ | - | - | - | - | - |
| Branch If Lower Or Same | BLS | 23 | 4 | 2 |  |  | * |  |  |  |  |  |  |  | $c+z=1$ | - | - | - | - | - |
| Branch If $<$ Zero | blt | 20 | 4 |  |  |  |  |  |  |  |  |  |  |  | $N \oplus\left(\begin{array}{l}\text { c }\end{array}\right.$ | - |  | - | - | - |
| Branch If Mirus | BMI | 2 B | 4 | 2 |  |  |  |  |  |  |  |  |  |  | $\mathrm{N}=1$ | - | - | - | - | - |
| Branch If Not Equal Zero | bne | 26 | 4 | 2 |  |  |  |  |  |  |  |  |  |  | $z=0$ | - |  |  | - | - |
| Branch If Overflow Clear | bVC | 28 | 4 |  |  | - |  |  |  |  |  |  |  |  | $v=0$ | - |  | - | - | - |
| Branch If Overitow Set | BVS | 29 | 4 | 2 | - | b |  |  |  |  |  |  |  |  | $v=1$ | - | - | - | - | - |
| Branch if Plus | BPL | 2 A |  |  |  |  |  |  |  |  |  |  |  |  | $N=0$ | - | - | - | - | - |
| Branch To Subroutine | BSR |  |  | 2 |  |  |  |  |  |  |  |  |  |  |  | - | - | - | - | - |
| Jump | JMP. |  | = | * | 6 E | 4 | $2$ | $T E$ | $3$ | 3 |  |  |  |  | See Special Operations |  | - | - | - | - |
| Jump To Subroutine | JSR |  | 2 |  | AD | 8 | 2 | Bо | 9 | 3 |  |  |  |  | (Figure 16) |  |  |  | - | - |
| No Operation | NOP |  |  |  |  |  |  |  |  |  |  | 2 | 1 |  | Advances Prog. Contr. Only |  |  | - |  | - |
| Return Fram Interrupl |  |  |  |  |  |  |  |  |  |  | 3 B | 10 | 1 |  |  |  |  |  |  |  |
| Return From Subroutine | RIS |  |  |  |  |  |  |  |  |  | 39 | 5 | 1 |  |  |  |  |  |  | - |
| Soltware Interrupt | swn |  |  |  |  |  |  |  |  |  | 3 F | 12 | 1 |  | See Special Uperations | - | - | - | - | - |
| Wait for Interrupt | wal |  |  |  |  |  |  |  |  |  |  | 9 |  |  | (Figure 16) |  |  | - | - |  |

SPECIAL OPERATIONS JSR, JUMP TO SUBROUTINE:


BSR, BAANCH TO SUBROUTTINE:

$$
\begin{aligned}
& n+2 \text { Formed From }[n+2]_{H} \text { and }[n+2]_{L}
\end{aligned}
$$

JMP, JUMP



RTS, RETURN FROM SUBROUTINE:


RTI, RETURN FROM INTERRUPT:


TABLE 6 - CONDITION CODE REGISTER MANIPULATION INSTRUCTIONS

| OPERATIONS | MNEMONIC | IMPLIED. |  |  |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  |  |  | BOOLEAN OPERATION | 5 | 4 | 3 | 2 | 1 | 0 |
|  |  | OP | $\sim$ | $=$ |  | H | 1 | N | 2 | $v$ | C |
| Clear Carry | CLC | OC. | 2 | 1 | $0 \sim \mathrm{C}$ |  | $\bullet$ | $\bullet$ | $\bullet$ | - | R |
| Clear Interrupt Mask | CL | OE | 2 | 1 | , 0-1 | $\bullet$ | R | - | $\bullet$ | - | $\bullet$ |
| Clear Overflow | CLV | OA | 2 | 1 | $0 \rightarrow \mathrm{~V}$ | $\bullet$ | - | $\bullet$ | - | R | $\bullet$ |
| Sat Carry | SEC | 0 O | 2 | 1 | $1-\mathrm{C}$ | - | - | - | - | - | S |
| Set Interrupt Mask | SEI. | 0 F | 2 | 1 | $1 \rightarrow 1$ | $\bullet$ | S | - | $\bullet$ | $\bullet$ | $\bullet$ |
| Set Overflow | SEV | OB | 2 | 1 | $1 \rightarrow \mathrm{~V}$ | $\bullet$ | - | - | $\bullet$ | S | $\bullet$ |
| Acmitr A $\rightarrow$ CCR | TAP | 06 | 2 | 1 | $A \rightarrow C C R$ |  |  |  |  |  |  |
| $C C R \rightarrow$ Acmltr $A$ | TPA | 07 | 2 | 1 | $\mathrm{CCR} \rightarrow \mathrm{A}$ | - | $\bullet$ | - | $\bullet$ | $\bullet$ | - |

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


MOTOROLA Semiconductor Products Inc.
(Times in Machine Cycle)
 (Dual Operand)



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

## SUMMARY OF CYCLE-BY-CYCLE OPERATION

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/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 cycies 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 <br> and Instructions | Cycles | Cycle | VMA |  |  |  |  |
| ---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |


| ADC EOR <br> ADD LDA <br> AND ORA <br> BIT, SBC <br> CMP SUB | 2 | 1 | 1 | Op Code Address <br> Op Code Address +1 | $\frac{1}{1}$ | Op Code Operand Data |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| $\begin{aligned} & \text { CPX } \\ & \text { LDS } \end{aligned}$ | 3 | 1 2 3 | 1 | Op Code Address <br> Op Code Address + 1 <br> Op Code Address +2 | 1 | Op Code <br> Operand Data (High Order Byte) <br> Operand Data (Low Order Byte) |

DIRECT



TABLE 8 - OPERATIONS SUMMARY (CONCLUDED)

| Address Mode and Instructions | Cycles | Cycle \# | VMA Line | Address Bus | $R / \bar{W}$ Line | Data Bus |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| INHERENT (Continued) |  |  |  |  |  |  |
| WAI | 9 | 1 2 3 4 5 6 7 8 9 | $\begin{aligned} & 1 \\ & 1 \\ & 1 \\ & 1 \\ & 1 \\ & 1 \\ & 1 \\ & 1 \\ & 1 \end{aligned}$ | Op Code Address <br> Op Code Address +1 <br> Stack. Pointer <br> Stack Pointer - 1 <br> Stack Pointer - 2 <br> Stack Pointer - 3 <br> Stack Pointer - 4 <br> Stack Pointer - 5 <br> Stack Pointer - 6 | $\begin{aligned} & 1 \\ & 1 \\ & 0 \\ & 0 \\ & 0 \\ & 0 \\ & 0 \\ & 0 \\ & 1 \end{aligned}$ | Op Code <br> Op Code of Next Instruction <br> Return Address (Low Order Byte) <br> Return Address (High Order Byte) <br> Index Register (Low Order Byte) <br> Index Register (High Order Byte) <br> Contents of Accumulator $A$ <br> Contents of Accumulator B <br> Contents of Cond. Code Register |
| RTI | $10$ | 1 2 3 4 5 6 7 8 9 10 |  | Op Code Address <br> Op Code Address + 1 <br> Stack Pointer <br> Stack Pointer + 1 <br> Stack Pointer +2 <br> Stack Pointer +3 <br> Stack Pointer +4 <br> Stack Pointer +5 <br> Stack Pointer +6 <br> Stack Pointer +7 |  | Op Code <br> Irrelevant Data (Note 2) <br> Irrelevant Data (Note 1) <br> Contents of Cond. Code Register from Stack <br> Contents of Accumulator B from Stack <br> Contents of Accumulator A from Stack <br> Index Register from Stack (High Order Byte) <br> Index Register from Stack (Low Order Byte) <br> Next Instruction Address from Stack (High Order Byte) <br> Next Instruction Address from Stack (Low Order Byte) |
| SWI | $12$ | 1 2 3 4 5 6 7 8 92 10 11 | 1 1 1 1 1 1 1 1 1 0 1 | Op Code Address <br> Op Code Address +1 <br> Stack Pointer <br> Stack Pointer- 1 <br> Stack Pointer - 2 <br> Stack Pointer - 3 <br> Stack Pointer - 4 <br> Stack Pointer - 5 <br> Stack Pointer - 6 <br> Stack Pointer - 7 <br> Vector Address FFFA (Hex) <br> Vector Address FFFB (Hex) | $\begin{aligned} & 1 \\ & 1 \\ & 0 \\ & 0 \\ & 0 \\ & 0 \\ & 0 \\ & 0 \\ & 0 \\ & 1 \\ & 1 \end{aligned}$ | Op Code <br> Irrelevant Data (Note 1) <br> Return Address (Low Order Byte) <br> Return Address (High Order Byte) <br> Index Register (Low Order Byte) <br> Index Register (High Order Byte) <br> Contents of Accumulator A <br> Cantents of Accumulator B <br> Contents of Cond. Code Register <br> Irrelevant Data (Note 1) <br> Address of Subroutine (High Order Byte) <br> Address of Subroutine (Low Order Byte) |



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.

TABLE 8 - OPERATIONS SUMMARY (CONTINUED)

| Address Mode and Instructions | Cycles | Cycle \# | VMA Line | Address Bus | $\begin{gathered} R / W \\ L \text { ine } \end{gathered}$ | Data Bus |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| EXTENDED (Continued) |  |  |  |  |  |  |
| STS <br> STX | 6 | 1 2 2 3 4 4 5 6 | $\begin{aligned} & 1 \\ & 1 \\ & 1 \\ & 0 \\ & 1 \\ & 1 \end{aligned}$ | Op Code Address <br> Op Code Address + 1 <br> Op Code Address +2 <br> Address of Operand <br> Address of Operand <br> Address of Operand +1 | $\begin{aligned} & 1 \\ & 1 \\ & 1 \\ & 0 \\ & 0 \end{aligned}$ | Op Code <br> Address of Operand (High Order Byte) <br> Address of Operand (Low Order Byte) <br> Irrelevant Data (Note 1) <br> Operand Data (High Order Byte) <br> Operand Data (Low Order Byte) |
| JSR |  | 1 2 3 3 4 5 6 7 8 9 | $\begin{gathered} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{gathered}$ | Op Code Address <br> Op Code Address + 1 <br> Op Code Address +2 <br> Subroutine Starting Address <br> Stack Pointer <br> Stack Pointer - 1 <br> Stack Pointer-2 <br> Op Code Address + 2 <br> Op Code Address +2 | $1$ | Op Code <br> Address of Subroutine (High Order Byte) <br> Address of Subroutine (Low Order Byte) <br> Op Code of Next Instruction <br> Return Address (Low Order Byte) <br> Return Address (High Order Byte) <br> Irrelevant Data (Note 1) <br> Irrelevant Data (Note 1) <br> Address of Subroutine (Low Order Byte) |

## INHERENT

|  | $2$ | $\frac{1}{2}$ | $\begin{aligned} & 1 \\ & 1 \end{aligned}$ | Op Code Address <br> Op Code Address + 1 |  | Op Code <br> Op Code of Next Instruction |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| DES <br> DEX <br> INS <br> INX | 4 | $\begin{aligned} & 1 \\ & 2 \\ & 3 \\ & 4 \end{aligned}$ | 1 1 0 0 | Op Code Address <br> Op Code Address +1 <br> Previous Register Contents <br> New Register Contents | $\frac{1}{1}$ | Op Code <br> Op Code of Next Instruction Irretevant Data (Note 1) <br> Irrelevant Data (Note 1) |
| PSH | 4 | 1 2 3 3 4 | $\begin{aligned} & 1 \\ & 1 \\ & 1 \\ & 0 \end{aligned}$ | Op Code Address <br> Op Code Address $t 1$ <br> Stack Pointer <br> Stack Pöinter - 1 | $1$ | Op Code <br> Op Code of Next Instruction <br> Accumulator Data <br> Accumulator Data |
| PUL | 4 | 1. 2 3 4. | $\left\lvert\, \begin{aligned} & 1 \\ & 1 \\ & 0 \\ & 0 \end{aligned}\right.$ | Op Code Address <br> Op Code Address +1 <br> Stack Pointer <br> Stack Pointer + 1 | $1$ | Op Code <br> Op Code of Next Instruction Irrelevant Data (Note 1) Operand Däta from Stack |
| $T S X$ |  | 1 -2 3 3 4 | $\begin{aligned} & 1 \\ & 1 \\ & 0 \\ & 0 \end{aligned}$ | Op Code Address <br> Op Code Address + 1 <br> Stack Pointer <br> New Index Register | $\frac{1}{1}$ | Op Code <br> Op Code of Next Instruction Irrelevant Data (Note 1) <br> Irrelevant Data (Note 1) |
| TXS | 4 | $\begin{gathered} 1 \\ 2 \\ 3 \\ 4 \end{gathered}$ | 1 1 1 0 0 | Op Code Address <br> Op Code Address +1 <br> Index Register <br> New Stack Pointer | $\begin{aligned} & 1 \\ & 1 \\ & 1 \end{aligned}$ | Op Code <br> Op Code of Next Instruction <br> Irrelevant Data <br> Irrelevant Data |
| RTS | $5$ |  | 1 1 0 1 1 1 | Op Code Address <br> Op Code Address +1 <br> Stack Pointer <br> Stack Pointer + 1 <br> Stack Pointer +2 | $\begin{aligned} & 1 \\ & 1 \\ & 1 \\ & 1 \end{aligned}$ | Op Code <br> Irrelevant Data (Note 2) <br> Irrelevant Data (Note 1) <br> Address of Next Instruction (High Order Bute) <br> Address of Next Instruction (Low Order Byte) |



Motorola reserves the right to make changes to any products herein to improve reliability, function or design. Motorola does not assume any liability arișing out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights nor the rights of others.

| Address Mode and Instructions | Cycles | Cycle \# | VMA Line | Address Bus | $\mathrm{B} / \mathrm{w}$ Line | Data Bus |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |


| STA | 6 | 1 <br> 2 <br> 2 <br> 3 <br> 4 <br> 5 <br> 6 | 1 1 0 0 0 1 1. | Op Code Address <br> Op Code Address +1 <br> Index Register <br> Index Register Plus Offset (w/o Carry) <br> Index Register Plus Offset <br> Index Register Plus Offset | $\begin{aligned} & 1 . \\ & 1 . \\ & 1 \\ & 1 \\ & 0 \end{aligned}$ | Op Code <br> Offset <br> Irrelevant Data (Note 1) <br> Irrelevant Data (Note 1) <br> Irrelevant Data (Note 1) <br> Operand Data |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| ASL LSR ASR NEG CLR ROL COM ROR DEC TST INC T | $7$ | 1 <br> 2 <br> 3 <br> 4 <br> 4 <br> 5 <br> 6 <br> 7 | 1 1 0 0 1 0 0 $1 / 0$ (Note $3)$ | Op Code Address <br> Op Code Address +1 <br> Index Register <br> Index Register Plus Offset (w/o Carry) <br> IndexRegister Plus Offset <br> Index Register Plus Offset <br> Index Register Plus Offset | $\begin{aligned} & 1 \\ & 1 \\ & 1 \\ & 1 \\ & 1 \end{aligned}$ | Op Code <br> Offset <br> Irrelevant Data (Note 1) <br> Irrelevant Data (Note 1) <br> Current Operand Data <br> Irrelevant Data Note 1) <br> New Operand Data (Note 3) |
| $\begin{aligned} & \text { STS } \\ & \text { STX } \end{aligned}$ | 7 | 1 2 3 4 5 6 7 | 1 1 0 0 0 0 1 1 | Op Code Address <br> Op Code Address +1 <br> Index Register <br> Index Register Plus Offset (w/o Carry) <br> Index Register Plus Offset <br> Index. Register Plus Offset <br> Index Register Plús Offset + 1 | 1 1 1 1 1 0 0 | Op Code <br> Offset <br> Irrelevant Data (Note 1) <br> Irrelevant Data (Note 1) <br> Irrelevant Data (Note 1) <br> Operand Data (High Order Byte) <br> Operand Data (Low Order Byte) |
| JSR | 8 | 1 <br> 2 <br> 2 <br> 3 <br> 4 <br> 5 <br> 6 <br> 7 <br> 7 <br> 8 | $\left(\begin{array}{c}1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0\end{array}\right.$ | Op Code Address <br> Op Code Address +1 <br> Index Register <br> Stack Pointer <br> Stack Pointer - 1 <br> Stack Pointer -2 <br> Index Fegister <br> Index Reglster Plus Offset (w/o Carry) | $\begin{aligned} & 1 \\ & 1 \\ & 0 \\ & 0 \\ & 1 \\ & 1 \\ & 1 \end{aligned}$ | Op Code <br> Offset <br> Irrelevant Data (Note 1) <br> Return Address (Low Order Byte) <br> Return Address (High Order Byte) <br> Irrelevant Data (Note 1) <br> Irrelevant Data (Note 1) <br> Irrelevant Data (Note 1) |


[^0]:    *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 (MC68A02, MC68A08, MC68B02, and MC68B08). On-board RAM can be used for data storage with all parts.

[^1]:    OP Operation Code (Hexadecimal)
    Number of MPU Cycles:
    $=$ Number of Program Bytes
    Arithmetic Plus,
    Arithmetic Minusi
    Boolear AND
    MSP Contents of memory location pointed to be Slack Pointer,
    Note - Accumulator addressing mode instruclions are ineluded in the column for IMPLIED addressing

    ## LEGEND:

