MOTOROLA

3

MC6805S2

8-BIT MICROCOMPUTER WITH ANALOG-TO-DIGITAL CONVERTER, SERIAL PERIPHERAL INTERFACE, AND THREE TIMERS

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

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Concluded)

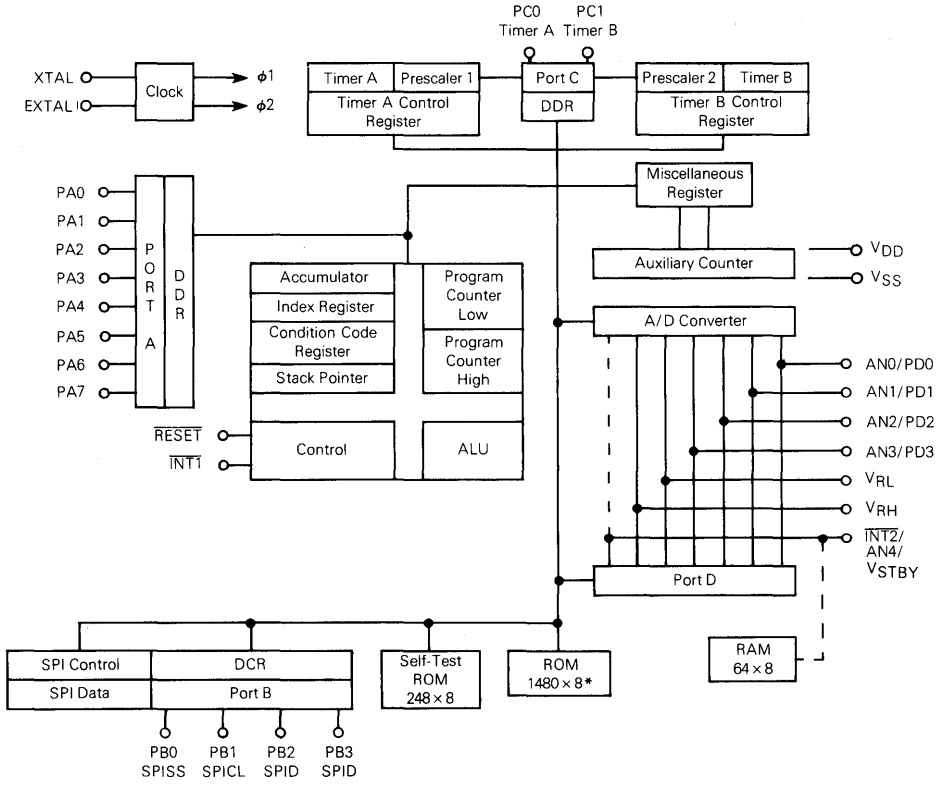
LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS
(Continued)

LIST OF TABLES

SECTION 1
INTRODUCTION

The MC6805S2 microcomputer unit (MCU) is a member of the M6805 Family of low-cost single-chip microcomputers. This 8-bit microcomputer contains a CPU, on-chip clock, ROM, RAM, I/O, 4-channel 8-bit analog-to-digital (A/D) converter, three timers, two programmable prescalers, and a serial peripheral interface (a block diagram is shown in Figure 1-1). It is designed for the user who needs an economical microcomputer with the proven capabilities of the M6800-based instruction set.



*Includes 8 bytes for interrupt vectors.

Figure 1-1. Block Diagram

1.1 HARDWARE FEATURES

The following are some of the hardware features of the MC6805S2 MCU.

- A/D Converter
  - 8-Bit Conversion, Monotonic
  - Four Multiplexed Analog Inputs
  - Ratiometric Conversion
- 21 TTL Including Eight TTL/CMOS Compatible I/O Lines
  - 14 Bidirectional (Four Lines are LED Compatible)
  - 7 Input-Only
- 1480 Bytes of User ROM
- 64 Bytes of RAM
- Self-Check Mode
- Serial Peripheral Interface (SPI)
- Zero-Crossing Detect/Interrupt
- One 8-Bit and One 16-Bit Timer
- One 7-Bit and One 15-Bit Software Programmable Prescaler
- Three Bidirectional I/O Lines with TTL or Open-Drain Interface (Software Programmable)
- Auxiliary Counter with "Watchdog" Reset Feature
- 5-Volt Single Supply

1.2 SOFTWARE FEATURES

The following are some of the software features of the MC6805S2 MCU.

- 10 Powerful Addressing Modes
- Byte Efficient Instruction Set with True Bit Manipulation, Bit Test, and Branch Instructions
- Single Instruction Memory Examine/Change
- Powerful Indexed Addressing for Tables
- Full Set of Conditional Branches
- Memory Usable as Register/Flags
- User Callable Self-Check Subroutines
- Complete Development System Support on EXORciser, EXORset, and HDS-200 Available Now

1.3 USER SELECTABLE OPTIONS

The following are user selectable options of the MC6805S2 MCU.

- Eight Bidirectional I/O Lines with TTL or TTL/CMOS Interface Option
- Crystal or Low-Cost Resistor Oscillator Option
- Low Voltage Inhibit Option
- Vectored Interrupts: Timer/SPI, Software, and External
- 16-Bytes Standby RAM Option
- Fifth A/D Channel Option

EXORciser is a registered trademark of Motorola Inc.
EXORset and HDS-200 are trademarks of Motorola Inc.

SECTION 2
SIGNAL DESCRIPTION, MEMORY, CPU, AND REGISTERS

This section provides a description of the signals, memory spaces, the central processing unit (CPU), and the various registers.

2.1 SIGNAL DESCRIPTION

The following paragraphs provide a brief description of the signals and a reference (if applicable) to other paragraphs that contain more detail about the function being performed.

2.1.1 V_{CC} and V_{SS}

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

2.1.2 $\overline{INT1}$ and $\overline{INT2}$

These pins provide the capability for asynchronously applying an external interrupt to the MCU. Refer to **5.4 INTERRUPTS** for additional information.

2.1.3 XTAL and EXTAL

These pins provide control input for the on-chip clock oscillator circuit. A crystal, a resistor, or an external signal, depending on the user selected manufacturing mask option, can be connected to these pins to provide a system clock source with various stability/cost tradeoffs. Lead length and stray capacitance on these two pins should be minimized. Refer to **5.3 INTERNAL CLOCK GENERATOR OPTIONS** for recommendations about these inputs.

2.1.4 Timer A/PC0 and Timer B/PC1

These pins allow an external input to be used to decrement the internal timer circuitry. Refer to **SECTION 3 TIMERS** for additional information about the timer circuitry.

2.1.5 $\overline{RESET}$

This pin allows resetting of the MCU at times other than the automatic resetting capability already in the MCU. The MCU can be reset by pulling $\overline{RESET}$ low. Refer to **5.2 RESETS** for additional information.

2.1.6 Input/Output Lines (PA0-PA7, PB0-PB3, PC0-PC1, and PD0-PD6)

Ports A, B, and C are programmable as either inputs or outputs under software control of the data direction registers (DDRs). Port D has up to four analog inputs or five via mask option, plus two voltage reference inputs when the analog-to-digital (A/D) converter is used (PD5/V_{RH}, PD4/V_{RL}, and an $\overline{INT2}$ input). If any analog input is used, then the voltage reference pins (PD5/V_{RH}, PD4/V_{RL}) must be used in the analog mode. Refer to **6.1 INPUT/OUTPUT, 6.6 ANALOG-TO-DIGITAL CONVERTER (A/D),** and **5.4 INTERRUPTS** for additional information. Port D can also be used as a 7-bit digital input-only port.

2.2. MEMORY

As shown in Figure 2-1, the MCU is capable of addressing 4096 bytes of memory and I/O registers with its program counter. The MC6805S2 MCU has implemented 1802 of these locations. This consists of: 1480 bytes of user ROM including eight interrupt vectors, 248 bytes of self-check ROM, 64 bytes of user RAM, seven bytes of port I/O, five timer registers, two A/D registers, a miscellaneous register, and two serial peripheral interface (SPI) registers. The user ROM has been split into three areas. The first area is memory locations $080 to $0FF, and allows the user to access these ROM locations utilizing the direct and table look-up indexed addressing modes. The main user ROM area is from $9C0 to $EFF. The last eight user ROM locations at the top of memory are for the interrupt vectors.

The MCU reserves the first 18 memory locations for I/O features, of which 17 have been implemented. These locations are used for the ports, the port DDRs, the timers, the miscellaneous register, the SPI, and the A/D. Of the 64 RAM bytes, 31 are shared with the stack area, from $061 through $07F. The stack must be used with care when data shares the stack area. The lower 16 bytes of RAM, between $40 and $4F, may be powered through the $\overline{INT2}$/PD6 pin via a user-defined mask option. Selection of this option does not exclude any of the available functions of the $\overline{INT2}$/PD6 input.

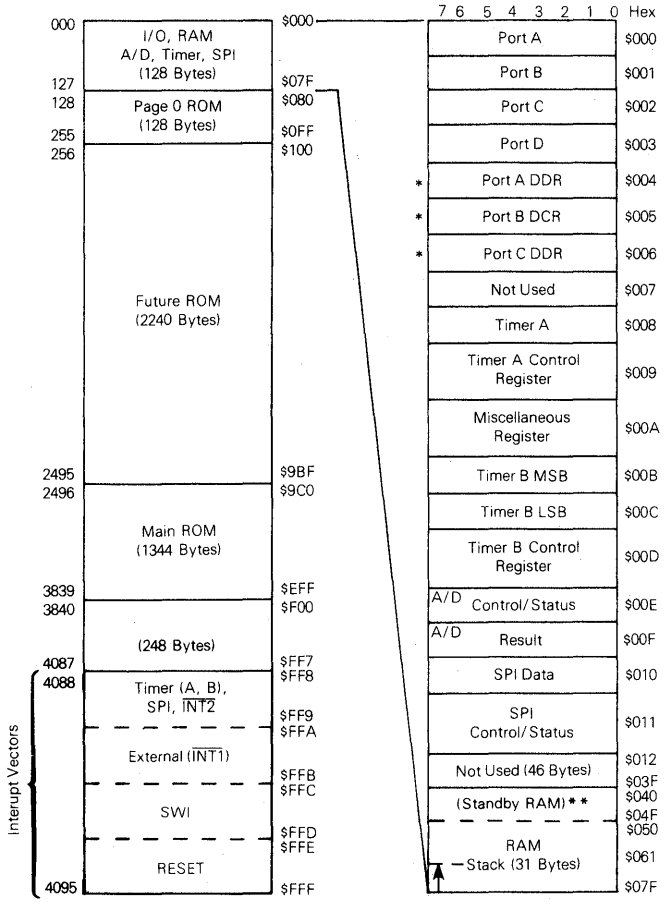
The shared stack area is used during the processing of an interrupt or subroutine calls to save the contents of the CPU state. The register contents are pushed onto the stack in the order shown in Figure 2-2. Since the stack pointer decrements during pushes, the low order byte (PCL) of the program counter is stacked first; then the high order four bits (PCH) are stacked. This ensures that the program counter is loaded correctly during pulls from the stack, since the stack pointer increments when it pulls data from the stack. A subroutine call results in only the program counter (PCL, PCH) contents being pushed onto the stack; the remaining CPU registers are not pushed.

2.3 CENTRAL PROCESSING UNIT

The CPU of the M6805 Family is implemented independently from the I/O or memory configuration. Consequently, it can be treated as an independent central processor communicating with I/O and memory via internal address, data, and control buses.

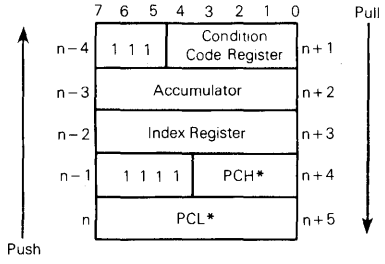
2.4 REGISTERS

The M6805 Family CPU has five registers available to the programmer. They are shown in Figure 2-3 and are explained in the following paragraphs.



*Data direction registers (DDRs) are write only; they read as $FF.
**Mask Option

Figure 2-1. Address Map



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

Figure 2-2. Interrupt Stacking Order

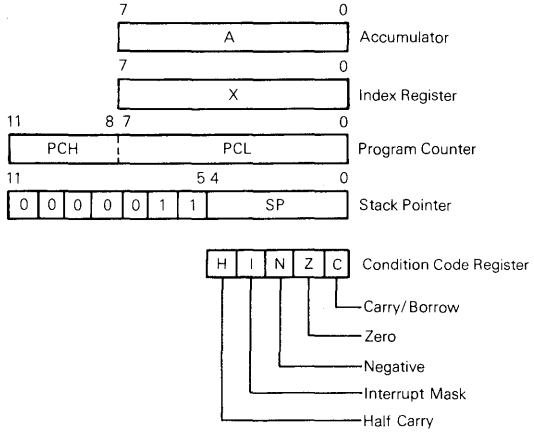


Figure 2-3. Programming Model

2.4.1 Accumulator (A)

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

2.4.2 Index Register (X)

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

2.4.3 Program Counter (PC)

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

2.4.4 Stack Pointer (SP)

The stack pointer is a 12-bit register that contains the address of the next free location on the stack. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set to location $07F. The stack pointer is then decremented as data is pushed onto the stack and incremented as data is pulled from the stack. The seven most significant bits of the stack pointer are permanently configured to 0000011. Subroutines and interrupts may be nested down to location $061 (31 bytes maximum) which allows the programmer to use up to 15 levels of subroutine calls (less if interrupts are allowed).

2.4.5 Condition Code Register (CC)

The condition code register is a 5-bit register in which four bits are used to indicate the results of the instruction just executed. These bits can be individually tested by a program and specific action taken as a result of their state. Each bit is explained in the following paragraphs. For more information concerning the condition code register refer to the *M6805 HMOS/M146805 CMOS Family Users Manual.*

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

2.4.5.2 INTERRUPT (I) — When set, this bit masks (disables) the timer (both A and B), external ($\overline{INT1}$ and $\overline{INT2}$), and the serial peripheral interface interrupts. If an interrupt occurs while this bit is set, the interrupt is latched and is processed as soon as the interrupt bit is cleared.

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

2.4.5.4 ZERO (Z) — When set, this bit indicates that the result of the last arithmetic, logic, or data manipulation was zero.

2.4.5.5 CARRY/BORROW (C) — When set, this bit indicates that a carry or borrow out of the arithmetic logic unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions plus shifts and rotates.

SECTION 3
TIMERS

The MC6805S2 has three timers and two programmable prescalers (see Figure 3-1) which are described in this section.

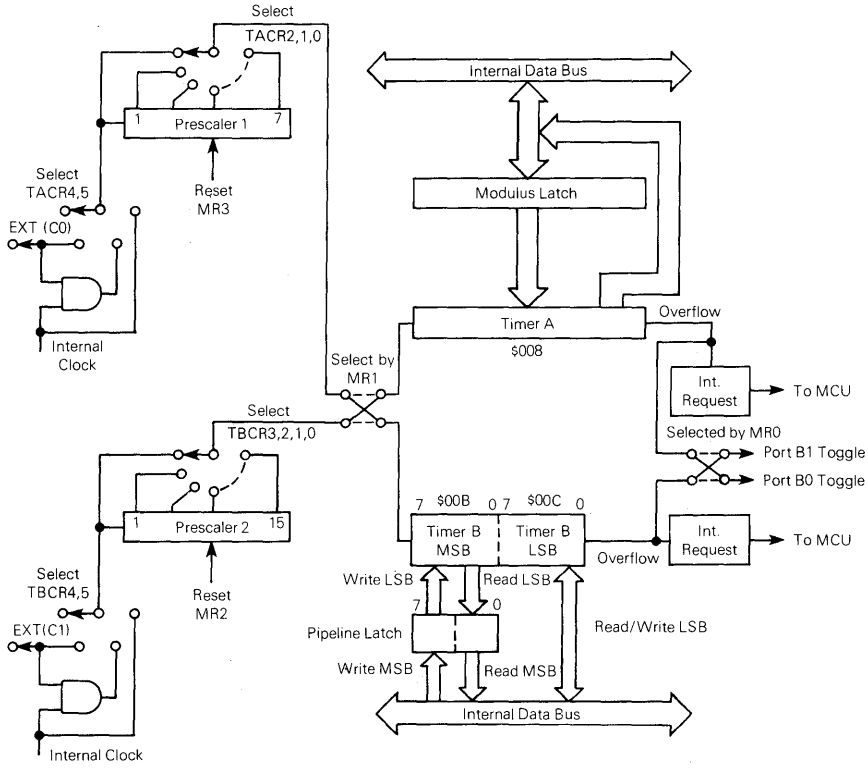


Figure 3-1. Timer A and B Block Diagram

3.1 TIMER A

Timer A is an 8-bit programmable down counter, which may be loaded under program control (see Figures 3-2 and 3-3). Included in this timer is a modulus latch which allows the timer to be "auto reloaded." Address $08 is the location of TIMER A's data register. Upon every clock input received, timer A decrements toward $00. Upon reaching this value, bit 7 in the timer A control register (TACR located at $09) is set, signifying a timer interrupt has been generated. At the same time, the timer is reloaded with the contents of the modulus latch. In addition to setting the interrupt bit, the transition to state $00 also generates an overflow condition which can be used to toggle bit 0 or bit 1 of port B directly, under the control of the miscellaneous register bit 3 (MR3), the serial peripheral interface control register, and the port B data direction register. The bit selected depends upon the state of bit 0 of the miscellaneous register. The timer interrupt may be masked by setting bit 6 of the TACR. Of course, the I bit in the condition code register will also prevent a timer interrupt from being processed. The timer interrupt vector locations are $FF8 and $FF9. The timer interrupt request bit *MUST* be cleared by software.

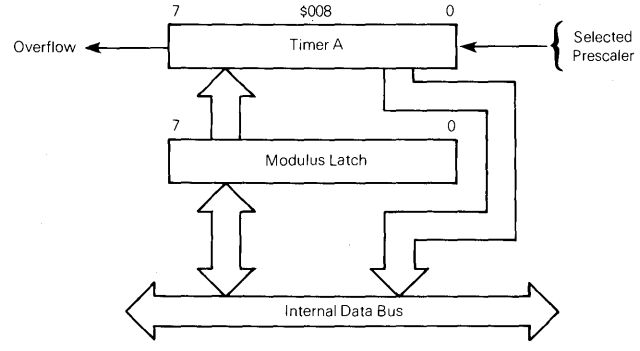


Figure 3-2. Timer A

TACR7	TACR6	TACR5	TACR4	TACR3	TACR2	TACR1	TACR0

TACR7 = Timer A Interrupt Request Flag
TACR6 = Timer A Interrupt Request Mask
TACR5 = External/Internal
TACR4 = External Enable
TACR3 = Timer A Load Mode Control
TACR2, TACR1, TACR0 } Prescaler 1 Divide Ratio Select

TACR5	TACR4	Prescaler 1 Clock
0	0	Internal Clock
0	1	AND of Internal Clock•PC0
1	0	Clock Disabled
1	1	PC0 Positive Transition

Figure 3-3. Timer A Control Register

There are three ways of loading data from the modulus latch into timer A as described in the following paragraphs.

3.1.1 Direct Loading

When the MCU writes to the timer A data register, the data is latched by the modulus latch, and forced into the timer. For this operation, TACR bit 3 must be clear.

3.1.2 Asynchronous External Event Loading

When TACR bit 3 is a logic one, the contents of the modulus latch are transferred to the timer at the rising edge of the $\overline{INT2}$ interrupt request bit (MR7) gated with interrupt request mask bit (MR6).

NOTE

If this feature is used, then care must be taken in programming as it will start an interrupt service routine if the I bit in condition code register (CC) is clear.

Loading $00 to timer A allows a countdown of 256 clocks before next $00 state is reached.

3.1.3 Auto-Loading

Auto-loading of the modulus latch occurs whenever the timer reaches the $00 state. This mode is independent of the status of bit 3 in TACR.

NOTE

Loading modes 1 and 2 are mutually exclusive, and auto-loading occurs in both modes 1 and 2.

Timer A may be read at any time without disturbing the countdown mechanism of the timer. At reset, both the timer and modulus latch contents are set to $FF.

NOTE

Loading $01 to timer A should be avoided when operating with a divide-by-one prescaler. Doing so will inhibit timer A auto-loading, interrupt generation, and port B toggle mechanisms.

3.2 TIMER A CONTROL REGISTER

Timer A control register (TACR) occupies memory location $09 (see Figure 3-4). Five bits are allocated to timer A and three bits are used to control prescaler 1.

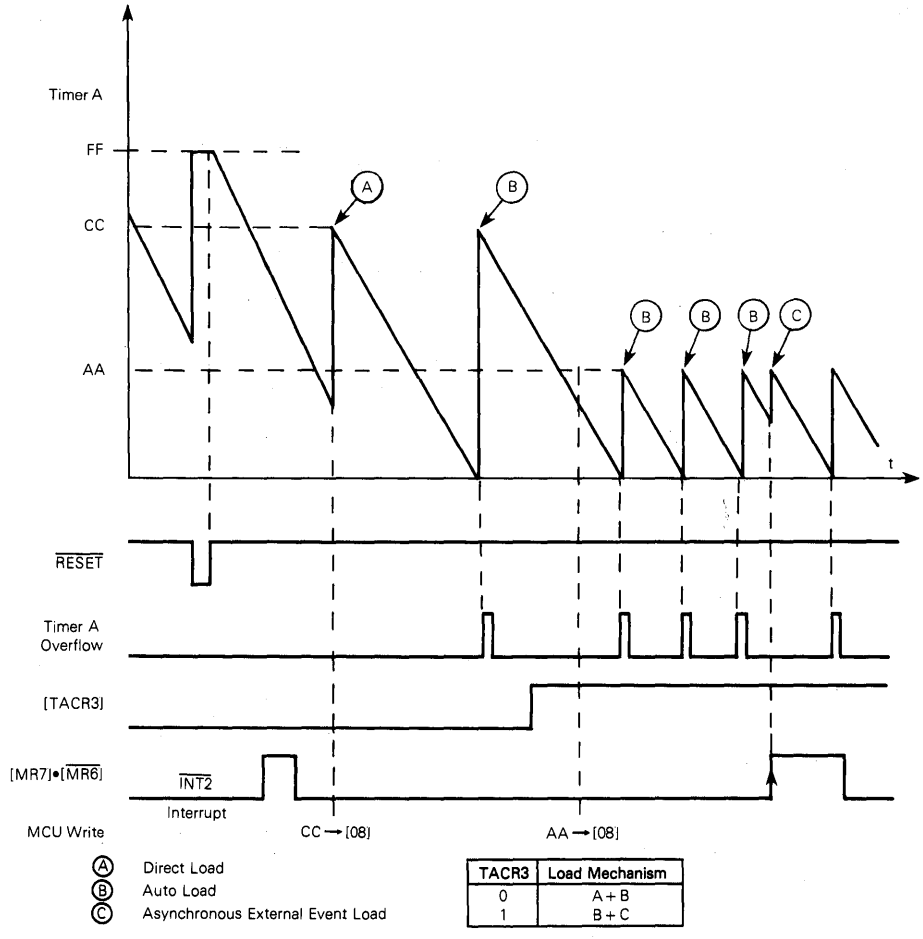


Figure 3-4. Timer A Operation

TACR7 Timer A Interrupt Request Flag
Set at the transition of timer A to $00 state. Cleared by software or at reset.

TACR6 Timer A Interrupt Request Mask
Set at reset or under program control. When set, timer A interrupt requests to the processor are inhibited. Cleared under program control.

TACR5 External or Internal Bit
Set under program control. When set, selects the input clock source for prescaler 1 to be the PC0 input, otherwise the internal clock (f_{OSC} divided by four) is the input clock source. Cleared at reset or under program control.

TACR4 External Enable Bit
Control bit used to enable the external timer pin (PC0).

TACR5	TACR4	Prescaler 1 Clock Source
0	0	Internal Clock
0	1	AND of Internal Clock and PC0*
1	0	Inputs Disabled
1	1	PC0* Low-to-High Transition

TACR3 Timer A Load Mode Control
Set under program control. When set, allows asynchronous external event loading of timer A ($\overline{INT2}$ driven loading is enabled). Cleared under program control or at reset. When clear, allows direct loading of timer A. Auto-loading takes place independent of TACR3 status. Cleared by reset or by program control.

TACR2, TACR1, TACR0 Prescaler 1 Division Ratio Control Bits
Set or cleared under program control, also cleared at reset. When set, these bits select one of the eight possible outputs on prescaler 1.

TACR2	TACR1	TACR0	Prescaler 1 Division Ratio
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

3.3 TIMER B

Timer B is a 16-bit timer which is accessed via two registers at $0B for the most-significant byte (MSB) and $0C for the least-significant byte (LSB) (refer to Figure 3-5). Included within the MSB of timer B is a "pipeline" latch, which allows a "snap shot" value of the entire 16 bits to be read.

---

* The status of PC0 depends upon the data direction status of PC0. If PC0 is an output, then the clock source is equal to the port data register content, independent of the port electrical loading. If an input, then the clock source is the logic level of PC0.

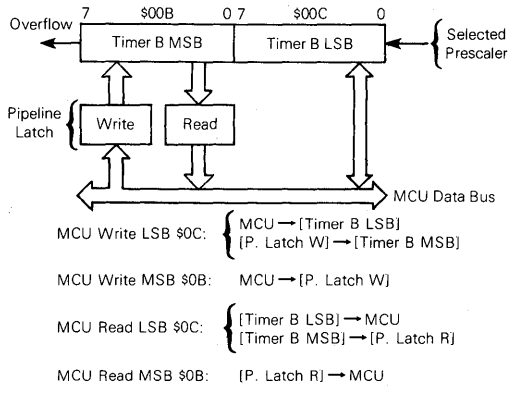


Figure 3-5. Timer B

Read/write operations to the LSB are direct. Reading the LSB can occur at any time without disturbing the count. At the time of the LSB read, the contents of the MSB are loaded into the pipeline latch, so when the MPU reads the MSB, it actually reads the latch.

Writing to the LSB of timer B may occur at any time, and the contents are immediately entered into the timer. At the same time the contents of the pipeline latch are forced into the MSB of the timer. Hence, a 16-bit word may be placed into the entire timer data register during a LSB write operation.

In order to manipulate a 16-bit word in timer B during a read, it is recommended that a read of the LSB be done first, then the MSB. A 16-bit write should be done in the opposite order. (First, write the MSB and then the LSB.) After reset, timer B contains $FFFF.

Like timer A, timer B decrements toward zero upon every clock input received and during the transition to state $00 TBCR7 in the timer B control register is set (TBCR is located at $0D).

The timer interrupt can be masked by setting the timer interrupt mask bit (TBCR6) (Figure 3-6). The I bit in the condition code register will also prevent a timer interrupt from being processed. The MCU responds to a timer interrupt by saving the current MCU state in the stack, fetching the vector from $FF8 and $FF9, and executing the interrupt routine. The timer interrupt routine bit *MUST* be cleared by software.

TBCR7	TBCR6	TBCR5	TBCR4	TCBR3	TBCR2	TBCR1	TBCR0

TBCR7 = Timer B Interrupt Request Flag
TBCR6 = Timer B Interrupt Request Mask
TBCR5 = External/Internal
TBCR4 = External Enable
TBCR3, TBCR2, TBCR1, TBCR0 } Prescaler 2 Divide Ratio Select

TBCR5	TBCR4	Prescaler 2 Clock
0	0	Internal
0	1	AND of Internal Clock•PC1
1	0	Clock Disabled
1	1	PC1 Positive Transition

Figure 3-6. Timer B Control Register

The transition to $00 generates an overflow pulse which may be used to force a port B data register toggle under the control of the miscellaneous register bit 3 (MR3), the SPI control register, and the port B data direction register. (See **6.5 MISCELLANEOUS REGISTER** and **4.3 SERIAL PERIPHERAL INTERFACE CONTROL AND STATUS REGISTER**.)

3.4 TIMER B CONTROL A STATUS REGISTER

Timer B control and status register (TBCR) occupies memory location $0D (see Figure 3-6). Four bits are allocated to timer B and four bits are used to control prescaler 2.

TBCR7 Timer B Interrupt Request Flag
Set at the transition of timer B to $00. Cleared by software or at reset.

TBCR6 Timer B Interrupt Request Mask
Set at reset or under program control. When set, inhibits timer B interrupt requests to the processor. Cleared under program control.

TBCR5 External or Internal Bit
Set under program control. When set, selects the input clock source for prescaler 2 to be the PC1 input, otherwise the internal clock (f_{OSC} divided by four) is the input clock source. Cleared at reset or under program control.

TBCR4 External Enable Bit
Set under program control or at reset. When set, this bit enables the external timer pin (PC1). Cleared under program control.

TBCR5	TBCR4	Prescaler 2 Clock Source
0	0	Internal Clock
0	1	AND of Internal Clock and PC1*
1	0	Inputs Disabled
1	1	PC1* Low-to-High Transition

TBCR3
TBCR2
TBCR1
TBCR0
Prescaler 2 Division Ratio Control Bits
Set or cleared under program control. When set, these bits select one of the 16 possible outputs on prescaler 2. All bits are cleared at reset.

* PC1 status depends on the data direction status of PC1. If PC1 is an output, then the clock source is equal to the port data register content, independent of the port electrical loading. If an input then the clock source is the logic level on PC1.

TBCR3	TBCR2	TBCR1	TBCR0	Division Ratio
0	0	0	0	1
0	0	0	1	2
0	0	1	0	4
0	0	1	1	8
0	1	0	0	16
0	1	0	1	32
0	1	1	0	64
0	1	1	1	128
1	0	0	0	256
1	0	0	1	512
1	0	1	0	1024
1	0	1	1	2048
1	1	0	0	4096
1	1	0	1	8192
1	1	1	0	16384
1	1	1	1	32768

3.5 PRESCALER 1

Prescaler 1 is a 7-bit binary down counter whose value is selected by TACR2, TACR1, and TACR0. The selected output is used as the clock input to either timer A or timer B, depending upon the status of the prescaler cross-couple bit (MR1). The type of clock source to prescaler 1 may be selected by TACR5 and TACR4 (see **3.1 TIMER A**).

Prescaler 1 is set to $7F at reset or under program control when a one is written to prescaler 1 clear bit (MR3).

NOTE

When changing outputs on the prescaler, a prescaler clear should be done first to avoid truncation errors.

3.6 PRESCALER 2

Prescaler 2 is a 15-bit down counter whose value is selected by TBCR3, TBCR2, TBCR1, and TBCR0. The selected output is used as the clock input to either timer A or timer B, depending upon the status of MR1. The type of clock source to prescaler 2 can be selected by TBCR5 and TBCR4 (see **3.3 TIMER B**).

Prescaler 2 is preset to $7FFF at reset, under program control when a logic one is written to prescaler 2 clear bit (MR2).

NOTE

When changing outputs on the prescaler, a prescaler clear should be done first to avoid truncation errors.

3.7 AUXILIARY COUNTER

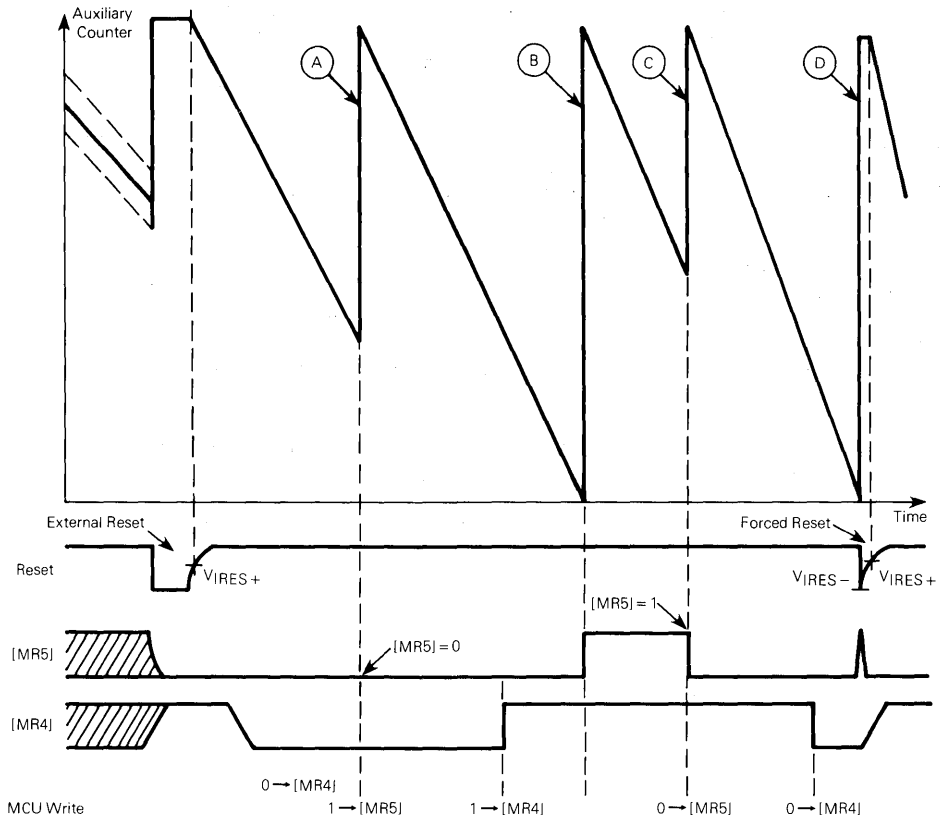
The third timer register in the MC6805S2 is the auxiliary counter, or "watchdog" timer. It is a fixed counter which is clocked by the internal clock (f_{OSC} divided by four). The total count period is 4095

cycles. The MCU communicates with this counter via the miscellaneous register bits 5 and 4 (MR5, MR4). Upon overflow, the auxiliary counter control/status bit in the miscellaneous register (MR5) is set. Countdown may be aborted at any time under program control, which also will reset the counter to 4095. To do this, the MCU must write to MR5 the inverse of what is read from MR5.

At reset, the counter is preset to its maximum count of 4095, and MR5 is cleared. The value of the counter is not accessible to the MCU; however, the possibility of detecting an underflow and presetting it at any time under program control allows it to be used as a fixed rate polled timer in applications requiring lengthy time out periods.

When the auxiliary counter reset mask bit in the miscellaneous register (MR4) is clear and the MR5 is set as a result of counter time out, the reset pin is internally pulled to ground (V_{SS}). This feature is useful in many applications, e.g., automotive, where the MCU operates in a noisy environment. Due to high energy spikes on the power supply and I/O lines, the MCU may lose control of the program and execute through non-valid memory space. The "watchdog" timer will bring the MCU back to reset. MR4 is automatically set at reset or under program control.

To return from a catastropic system runaway, the reset line is pulsed, which will restart the entire program. This program should regularly preset the auxiliary counter at a rate higher than the counter time out so as not to allow a forced reset. If program runaway does occur, it is likely that regular presetting of the auxiliary counter will not take place, and an overflow will force the MCU to regain control. (See Figure 3-7.)



(A) Counter Preset by Writing "1"

(B) Overflow: MR5 → 1; No Forced Reset

(C) Counter Reset by Writing "0"

(D) Overflow MR5 → 1 Forced Reset

Figure 3-7. Auxiliary Counter Operation

SECTION 4
SERIAL PERIPHERAL INTERFACE (SPI)

This section describes the operation of the serial peripheral interface (SPI) on the MC6805S2.

4.1 INTRODUCTION

The serial peripheral interface (SPI) on the MC6805S2 has several versatile operating modes. Arbitration on data and clock lines is provided. The SPI communicates with the MCU via data and control registers located at memory addresses $10 and $11, respectively. Operation of the SPI occurs via port B (see Figure 4-1).

The SPI consists of:

a) an 8-bit shift register (MSB out first; MSB in first) which may also be used as an SPI data register,
b) a divide-by-eight counter,
c) slave select/arbitration logic,
d) an SPI control register, and
e) start and stop bit detection capability.

Unlike the I/O port operation, the SPI data and clock inputs are always taken from their respective I/O port pins, regardless of the status of the data direction register relative to that port. This makes it easy to do data and clock arbitration.

Serial peripheral interface operation is enabled when the SPI enable bit (SPICR4) is set. When enabled, the SPI is capable of operating in the following modes:

a) one wire — auto clocked (e.g., NRZ),
b) two wire — half duplex,
c) two wire — half duplex with clock arbitration,
d) three wire — half duplex with slave select/busy line,
e) three wire — full duplex,
f) three wire — full duplex with clock arbitration, and
g) four wire — full duplex with slave select/busy line.

4.2 SPI TERMINOLOGY

The following explanations are provided to facilitate user understanding of the various operating modes of the serial peripheral interface (see Table 4-1).

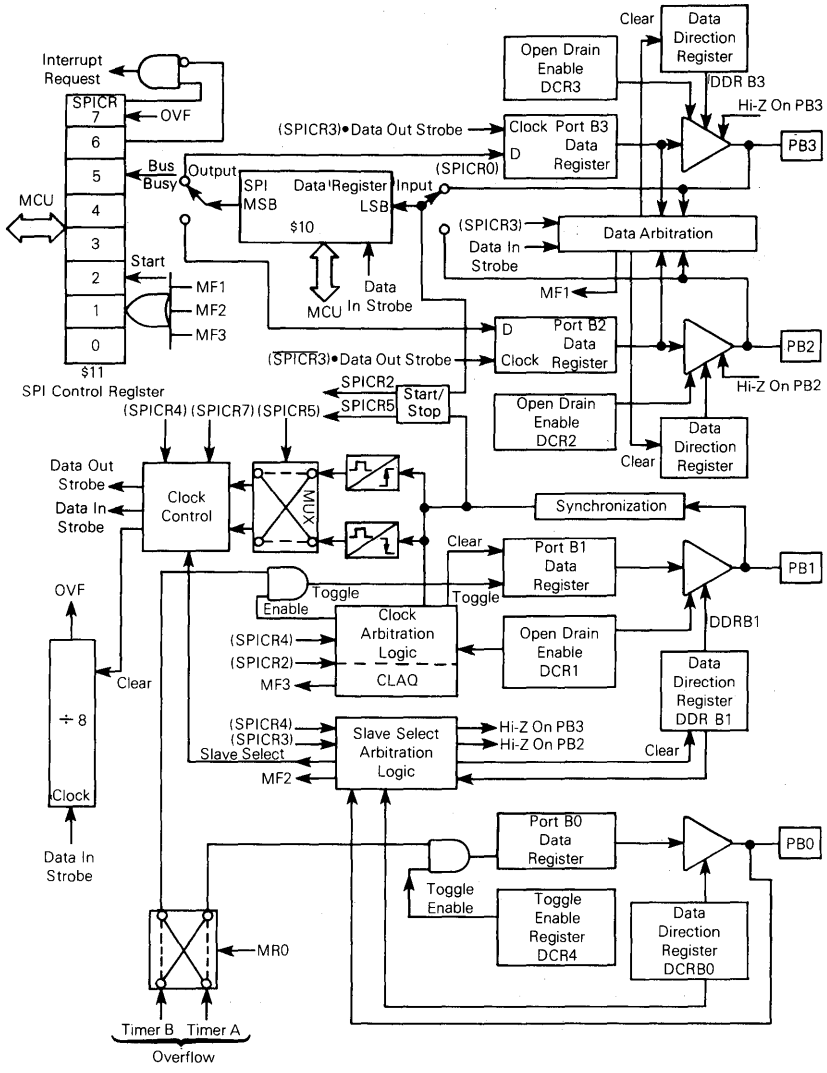


Figure 4-1. Serial Peripheral Interface (SPI)

Table 4-1. SPI Operation

DEFINTIONS

Transmitter – Data Master: DDRB2 or 3 = 1
Receiver – Data Slave: DDRB2 or 3 = 0
Clock Master: DDRB1 = 1
Clock Slave: DDRB1 = 0
Transaction Mode: SPICR4 = 1
1) Active: $\overline{SPICR7}$•($\overline{DDRB0}$•$\overline{PB0}$ + DDRB0) if DDRB1 = 0 (clock slave mode) or
$\overline{SPICR7}$•($\overline{DDRB0}$•PB0 + DDRB0) if DDRB1 = 1 (clock master mode)
Clock pulses allowed, data shifted
2) Idle: SPICR7 + $\overline{DDRB0}$•PB0 if DDRB1 = 0 (clock slave mode)
Clock pulses blocked, data output line in high-impedance state
Deselect Mode: SPICR4 = 0 – No SPI Operations

SLAVE SELECT INPUT

Slave Select Input: SPISS – PB0
If DDRB0 = 0 then no SPISS action on MCU
1) Master Mode: SPISS = 1 DDRB1 = 1
SPISS 1 – 0: Switch to Slave Mode (DDRB1 1 – 0)
Set SPICR1 (Mode Fault Flag)
2) Slave Mode: SPISS = 0 DDRB1 = 0
External clock is allowed to shift data in/out. If SPISS is pulled high, the external clock input pulses are inhibited; no data shift; divide-by-eight counter cleared; SPID (PB2 or PB3) switched to high-impedance state.
Used as Chip-Select Input

DATA ARBITRATION

Data master loses data mastership when data collision occurs during internal data strobe time.

If SPID output port (PB2 or PB3) = 1 while actual pin level is pulled low externally – conflict detected at internal data strobe time.

Then SPICR1 (mode fault flag) is set; SPID output port DDR (B2 or B3) 1 → 0 (high-impedance state).

CLOCK ARBITRATION

MCU has clock mastership (DDRB1 = 1)
1) Via SPISS line (DDRB0 = 0). If SPISS is pulled low, then clock mastership lost; DDRB1 1 → 0 (high-impedance state); SPICR1 is set (mode fault flag).
2) Via clock line SPICL (DDRB1 = 1 and DCRB5 = 0)
Condition: SPICL must have open-drain output (DCRB5 = 0)

If clock line is held low externally then clock mastership is not lost; minimum t_{CLH} and t_{CLK} times are guaranteed.

If SPICL goes low during idle mode then SPICR1 = 1 and clock line is switched low to inhibit the system clock.

MODE FAULT FLAG OPERATION (SPICR1)

Flag set when any of the following conditions occur.
Data arbitration occurs on SPID output.
Clock arbitration with SPISS during master to slave switching.
Clock arbitration via clock line if SPICL 1 → 0 during idle.

START, STOP, AND CLOCK IDLE CONDITIONS

Clock Idle: The clock level just prior to the transition that causes data on the serial output data line to be changed is defined as the SPI clock idle state.
SPICR5 = 0: SPICL Idle = Low State
SPICR5 = 1: SPICL Idle = High State

These definitions are necessary for determining start and stop conditions.

NOTE

Clock idle state can only be defined if SPICR4 = 0 (Deselect Mode)

Start Condition: Any negative transition of the data input line (PB2 or PB3) during an SPICL idle state.
Stop Condition: Any positive transition of the data input line during an SPICL idle state.

4.2.1 Clock Mastership

The SPI clock source is always taken from port B1. When the clock level on pin PB1 is defined by the MCU, it is said that the MCU has clock mastership. The principle condition for clock mastership during an SPI operation is that port B1 must be initialized by its DDR bit so that the port is in the output mode. When PB1 DDR is clear (i.e., configured as an input) during an SPI operation, and external device provides the SPI clock on pin PB1. This is referred to as the "clock slave" mode.

4.2.2 Data Mastership

SPI data transactions (transmission/reception) can occur through port B2, port B3, or through both of these ports as determined by the software. The MCU is said to have data mastership when the data output on the selected data output port is defined by the processor. The main requirement for data mastership during an SPI operation is that the selected SPI data output port, PB2 or PB3, be initialized by its DDR bit to be in the output mode. Routing of output data to the proper port data register is done by SPICR3. The MC6805S2 may be a "receiver" in any mode of operation.

4.2.3 SPI Transaction Mode

This is the mode where the SPI is allowed to operate (see Figure 4-2). Operation takes place via port B lines. SPI transactions are enabled when the SPI control register bit 4 (SPICR4) is set.

4.2.4 SPI Deselect Mode

SPICR4 is clear in this mode. All SPI operations and actions relative to the SPI operation are blocked in the SPI deselect mode. This mode is selected at reset.

4.2.5 SPI Active Mode

The SPI active mode is part of the transaction mode (Figure 4-3). In addition to the transaction mode requirements, the two following requirements must be met for the MC6805S2 to operate in the SPI active mode: 1) SPICR7 = 0, and SPISS (port B0) = 0 if PB1 DDR = 0 (clock slave mode) and 2) SPICR7 = 0 and SPISS = 1 if PB1 DDR = 1 (clock master mode). In this mode, the SPI clock pulses are allowed to shift serial information.

4.2.6 SPI Idle Mode

This is part of the transaction mode and is characterized additionally by 1) SPICR 7 = 1 or 2) slave select input (port B0) = 1 if DDRB1 = 0 (clock slave mode). In this mode all SPI clock pulses are blocked and, if the MCU is in the clock slave mode, the serial data out line is forced to high impedance if slave select input PB0 = 1. In this mode the MCU is processing serial data or is deselected under external control.

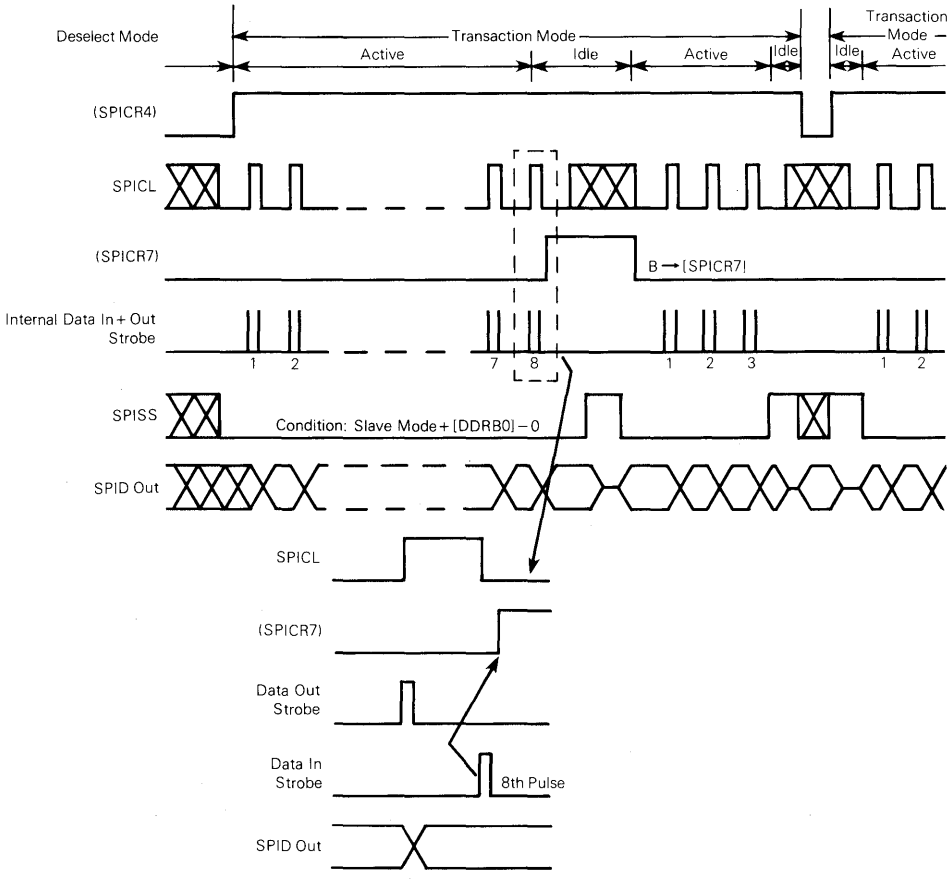


Figure 4-2. SPI Operation (Example: Clock Slave Mode)

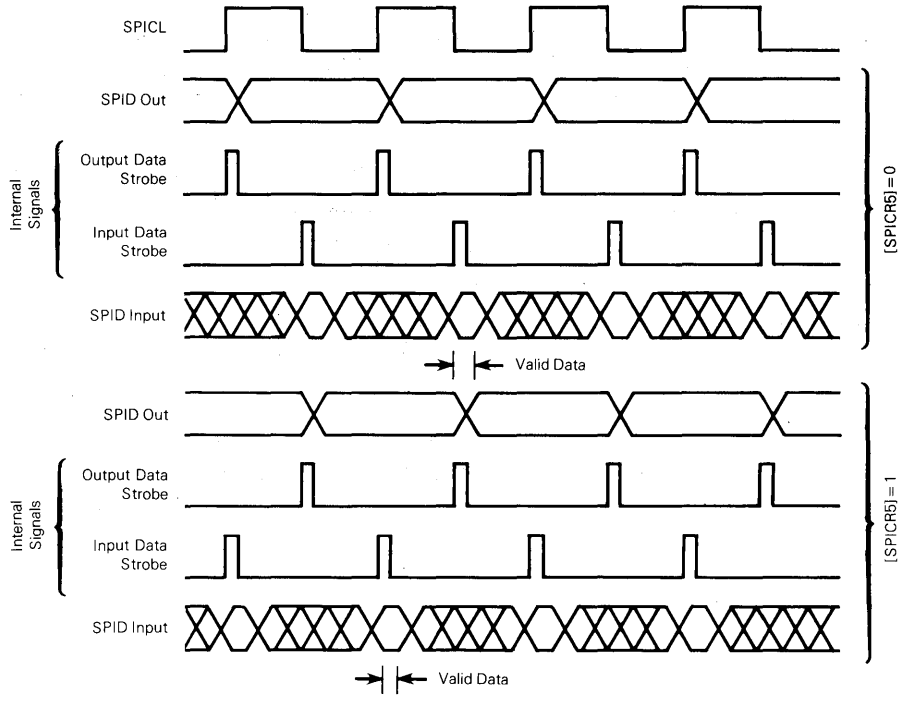


Figure 4-3. SPI Clock (Active Transaction)

4.3 SPI CONTROL AND STATUS REGISTER

This register, illustrated in Figure 4-4, contains the status and control bits related to SPI operations.

SPICR7 SPI Interrupt Request Bit
This bit is cleared at reset or under program control. When the eighth SPI data input strobe is detected from the SPI clock input this SPI interrupt request bit becomes set. When set, it forces the SPI into the idle mode. It remains in the idle mode until it is serviced. Only if SPICR7 is not masked by SPICR6 is the processor allowed to receive an interrupt request. The processor services this interrupt if the I bit is clear in the condition code register. It does so by fetching the interrupt vector from addresses $FF8 and $FF9. As long as SPICR7 is set the SPI remains in the idle mode during SPI transactions. SPICR7 is also cleared at the zero to one transition of SPICR2 due to a "start bit" detection during the transaction mode.

SPICR6 SPI Interrupt Request Mask Bit
This bit is set at reset or under program control. When set, it inhibits interrupt requests from SPICR7. This bit is cleared under program control, or at the zero to one transition of SPICR2 due to a "start bit" detection during the transaction mode.

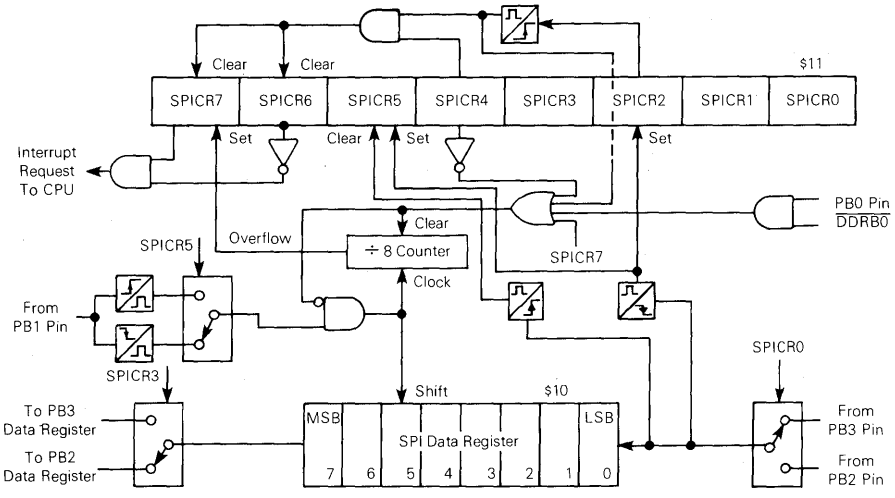


Figure 4-4. SPI Control Register Operation

SPICR5 SPI Clock Sense Bit/Bus-Busy Flag

This is a dual-function bit controlled by the status of SPICR4. The function of this bit is the following:

SPICR4	Mode	SPICR5 Function
0	SPI Deselect	SPI Clock Sense Bit (Read/Write)
1	SPI Transaction	SPI Bus-Busy Flag (Read Only)

If the SPI is in the deselect mode (SPICR4 = 0), SPICR5 becomes a read/write bit that controls the clock sense and SPICL idle level. When low, this bit causes SPI input data to be latched into the SPI data register on the negative edge of the SPI clock and output data to be changed on the positive edge of the SPI clock. This corresponds to a low SPICL idle level. When high, input data is latched on the positive edge and output data changed on the negative edge of the SPI clock. This corresponds to a high SPICL idle level. Data in the SPI data register is shifted by one location to the left at the SPI clock edge that latches SPI input data. This clock edge is referred to as the data input strobe.

During SPI operation (SPICR4 = 1), SPICR5 becomes a read-only bit that serves as a "bus-busy" flag. This flag is set due to a start condition and cleared due to a stop condition or at reset. A received MCU or a clock slave can poll this flag to determine the appropriate time to "capture the bus" and become a transmitter or clock master. This flag provides a "clean" hook-unhook mechanism to the serial bus to allow true multi-master operation.

In a properly ordered system, only one MCU has data mastership between a given start and stop condition. Outside this busy zone, the serial bus is considered free and is

signalled to the MCUs via the bus-busy flag. In the case that more than one processor attempts to gain access to the bus during this free zone, a normal data arbitration will take place. The MCUs with low priority can then get off the bus and remain as slaves until the next free zone.

SPICR4 SPI Operation Enable Bit
This bit is cleared at reset or under program control. When set under program control, it allows SPI operation and actions relative to it. When it is cleared, the divide-by-eight counter is reset; the SPI data register is disabled from shifting; and data and clock arbitration logic, as well as the slave select input logic, actions are inhibited. Logic status of this bit determines which of the dual functions is performed by SPICR2 and SPICR5.

SPICR3 SPI Data Output Select Bit
This bit is cleared at reset or under program control. When set under program control, this bit allows the output of the SPI data register to be loaded to the port B3 data register at the appropriate SPI clock edge selected by SPICR5, during the active transaction mode. When clear, the port B2 data register is loaded with the output of the SPI data register at the appropriate SPI clock edge during the active transaction mode.

SPICR2 Port B1 Toggle Enable/Start Bit
This is a dual function bit controlled by the status of SPICR4. The function of this bit is the following:

SPICR4	Mode	SPICR2 Function
0	SPI Deselect	Port B1 Toggle Enable
1	SPI Transaction	Start Bit

During non-SPI operations (SPICR4 = 0), when set under program control, SPICR2 enables port B1 data register toggle facility. Its prime use is in applications where continuous toggle operation may be required. This bit is cleared under program control or at reset.

During SPI operation (SPICR4 = 1), this bit is set by the negative transition of the data input of the SPI data shift register while the clock is in its idle level. The (SPICL) idle level is defined as the high level of SPICL if SPICR5 = 1 or the low level of SPICR5 = 0. Note that SPICR5 must be defined during the SPI deselect mode (SPICR4 = 0).

At the protocol level, this means that a "start" condition may be defined as an exceptional change of state of data input while this condition does not occur or should not be allowed to occur during the data transmission. "Start" condition information may be used to distinguish address and data transmissions, as well as transmission resync after transmission synchronization has been lost. This bit is cleared or set under program control.

SPICR1 Mode Fault Flag
This bit is cleared at reset or under program control. It is set under the following conditions:

1) When SPI data output arbitration occurs on the SPI data output port (PB3 or PB2) selected by SPICR3, the SPI data output port DDR is cleared (switches to high-impedance state), MCU loses data mastership, and the mode fault flag is set.

2) When the MCU has clock mastership (i.e., port B1 DDR = 1), slave select input PB0, if used as such in the application, should stay high. If a low level is detected on this input, then the MCU loses clock mastership, switches to clock slave mode, the port B1 DDR is cleared, and the mode fault flag is set.

3) When the MCU operates in the master mode where clock arbitration is done via the clock line, then the mode fault flag is set during the idle mode when a negative clock edge is detected on the SPI clock input. Simultaneously the port B1 data register is cleared.

This feature allows the MCU to detect that some other device has attempted to drive the SPI clock input while the MCU was not ready to perform a serial transaction; or that MCU has lost data mastership or clock mastership.

SPICR0 SPI Input Data Select Bit
This bit is cleared at reset or under program control. When set under program control, it allows SPI data from port B3 to be latched into the SPI data register. When clear, SPI data from port B2 is routed to the input of the SPI data register.

4.4 SPI DATA REGISTER

This register can be written into at any time. It can be read "on the fly" irrespective of serial operation without disturbing the data. Data is shifted left by one bit every time there is a data input strobe while the LSB is loaded with data from port B2 or B3 according to the status of SPICR0.

The MSB is loaded to the data register of port B2 or B3 according to the status of SPICR3 every time there is a data output strobe. Data input and output strobes are generated during the transitions of the SPI clock input to the MCU under the control of SPICR5. Data input and output strobes are generated internally only during the active transaction time.

4.5 SPI DIVIDE-BY-EIGHT COUNTER

This counter is cleared during SPI deselect or idle modes. It counts at every data input strobe during the SPI active transaction mode. At overflow, it sets SPICR7, which in turn puts the SPI in the idle mode and blocks all data input and output strobes. This counter is also cleared when the slave select input (PB0) is high while the MCU is operating the SPI with slave select in the slave mode, or when a "start" condition is detected. Clearing of the counter by the "start" condition allows resynchronization of data transmission between MCUs.

4.6 SPI OPERATION

The SPI may operate in a variety of ways depending on user application needs. The main modes are described below; however, this list is neither exhaustive nor absolute. Software assisted protocols may be defined to upgrade the hardware versatility and/or system performance of the MC6805S2. Some features common to all operating modes are outlined below.

1) The SPI data input and output paths may be individually routed under program control via SPICR3 and SPICR0 to or from either PB2 or PB3 (see Table 4-2). This gives rise to four possible routings useful in half duplex and full duplex operations, as well as allowing bidirectional information to flow in daisy-chained systems.

2) When data input and output is done on the same pin (PB2 or PB3), i.e., SPICR3 ⊕ SPICR0 = 0, then half duplex operation takes place. The unused port line (PB2 or PB3) is free for any other use.

3) Data input is always relative to the port pin logic level regardless of the data direction register status on that pin.

   If SPICR3 ⊕ SPICR0 = 0, then in case of data arbitration on the data output line, the data input to the SPI data register is always equal to the logic level imposed on the data input pin by the device which wins the data arbitration.

Table 4-2. Port B Status During SPI Operation

Port Name	Use	Input	Output	Comments
PB0	SPISS	Yes	No	Used as slave select input
PB0	Data	No	Yes	Used as "busy" signal or any digital output
PB1	SPICL	Yes	No	Clock slave
PB1	SPICL	No	Yes	Clock master
PB2	SPID	Yes	No	SPI data input SPICR0 = 0
PB2	SPID	No	Yes	SPI data output SPICR3 = 0
PB2	Data	Yes	Yes	Any digital signal SPICR3 = 1
PB3	SPID	Yes	No	SPI data input SPICR0 = 1
PB3	SPID	No	Yes	SPI data output SPICR3 = 1
PB3	Data	Yes	Yes	Any digital signal SPICR3 = 0

4) When full duplex operation is required, then SPICR3 ⊕ SPICR0 = 1. In this mode, 16 bits of information may be transferred with eight clock pulses between at least two devices with transmit capability. In this mode both PB2 and PB3 are used for SPI data transfer. Moreover, the same shift register is used for data out and data in. Thus, the byte transmitted is replaced by the byte received, removing the need for separate status bits for XMIT EMPTY and REC FULL. A single status bit, SPICR7, is used to signify that the input/output operation is complete.

5) The SPI clock is always provided on port B1. In the clock slave mode, the port B1 DDR is clear (i.e., input mode). In the clock master mode, the port B1 DDR is set and hence the MCU imposes the clock level on pin PB1 until there is clock arbitration on the clock line or until the MCU loses clock mastership when the slave select input PB0 goes low.

6) Clock pulse generation in the case of clock mastership is accomplished via the data register toggle facility provided on port B1. According to the status of MR0, the overflow pulse of either timer A or B is used as a toggle clock source during the active transaction time. Hence, the port B1 data register changes state every time there is a timer overflow. Clock frequency generated by this method is therefore half the overflow frequency of the selected timer. There is no fixed baud rate generation. The clock frequency is dependent on the prescaler clock source option, prescaler divide ratio, and timer divide ratio as well as the port C status in case of external clocking for the timer. Toggling of the port B1 data register is automatically allowed during the active transaction mode.

7) For correct transfer of data between devices connected to the SPI, all devices must have their output data strobe and input data strobe on the same clock edges.

8) For proper transmission, the first clock edge during the active transaction mode must be the output data strobe. When this occurs, the MSBs of the data registers of all transmitters are copied on to the data output pins (e.g., this is valid for devices with such output capability) and the MCU copies the MSB of its SPI on to the port B2 or B3 data register, according to SPICR3 status.

   On the opposite clock edge, all receivers internally generate the data input strobe and shift by one location the contents of the SPI data register. Data for the receivers is assumed to be stable on this clock edge. Hence, error-free master-slave type serial data transfer is accomplished. It is therefore important that before a serial data transfer starts, the master clock level has to be initialized under program control so as to create an output data strobe on the initial SPI clock edge.

   NOTE

   If the initial clock edge is the input data strobe, the MSB of all receivers are lost, and transmitted MCU data will have a framing error. However, if a peripheral transmitter device (without the selective data output and input strobe feature) is transmitting data to the MCU, then, the first clock edge should generate the data input strobe for the MCU.

9) The data direction registers of port B are always accessible during SPI operation. This is also true for data control registers of port B which control open-drain enables and the port B output toggle enables (DCR7 through DCR4). However, during SPI active transaction mode, the following data registers are not write accessible under program control:
   a) PB1 data register;
   b) PB2 data register if SPICR 3=0, and
   c) PB3 data register if SPICR 3=1.

   This allows write instructions to port B lines not used for SPI operation during the active transaction mode without affecting the contents of data registers used for SPI.

10) The toggle enable of the port B1 data register is asserted during the active transaction mode by the SPI logic. This starts the generation of SPI clock pulses if the MCU has the clock mastership. If the MCU is in the clock slave mode (DDR B1=0), then an external device provides the clock pulses.

11) Port B lines not used for SPI can be used for other digital functions, e.g.; a) in half-duplex or one-wire operation the unselected SPI data port may be used as I/O, and b) port B0 may always be used as digital output in the modes where SPI operates without slave select input.

4.7 START BIT OPERATION

In all operating modes of the SPI, it is implied that all data transmissions are sensitive to the clock edges. Depending on the state of SPICR5, data changes either as the result of the rising or falling edge of the clock SPICL.

The clock level prior to the transition that causes data on the serial data line to be changed is called the "idle" level. It is assumed that data must be stable just prior to and during the idle level during transmission.

Optional creation of an exception to this rule may be interpreted as additional information such as to 1) signal the beginning of a transmission; 2) to separate address and data fields and/or 3) to synchronize transmitter and receivers.

Negative transition of data input while the clock line (SPICL) is in its "idle" level is being defined as an exceptional condition on the MC6805S2 SPI. This condition causes SPICR2 and SPICR5 to be set and is defined as the start condition.

The rising edge of SPICR2 causes the divide-by-eight counter, SPICR7 and SPICR6 to be cleared.

Refer to Figure 4-5 for clock idle level definition, to Figure 4-6 for the start bit definition, and to Figure 4-7 for stop bit definition.

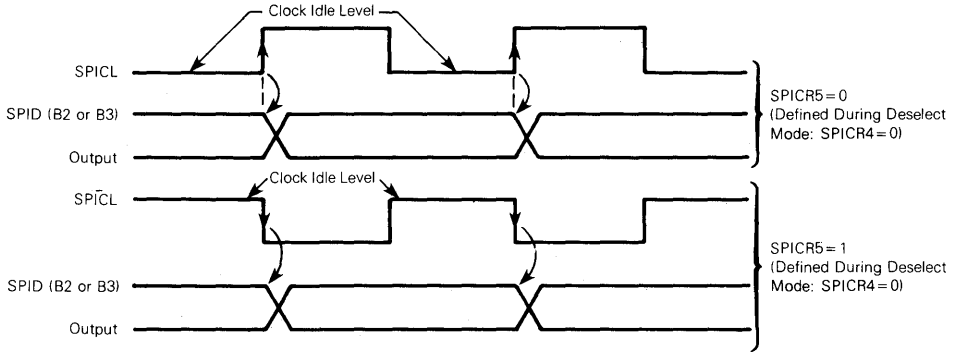


Figure 4-5. SPI Clock Idle Level Definition

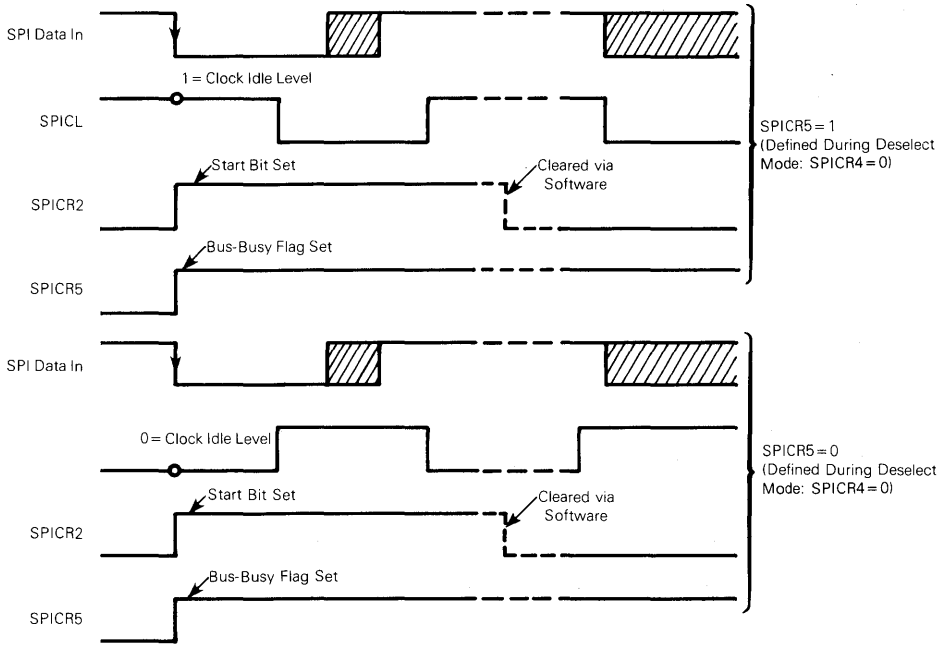


Figure 4-6. SPI Start Bit Definition

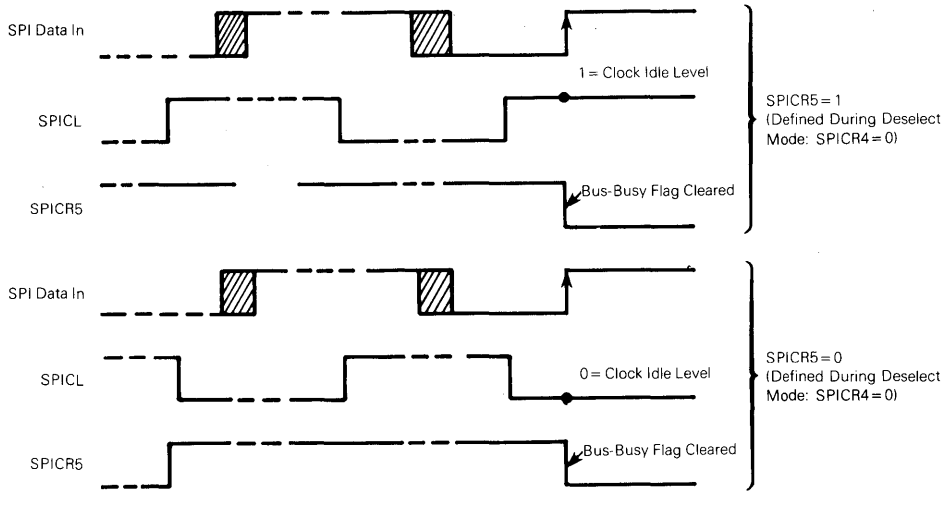


Figure 4-7. SPI Stop Bit Definition

4.8 ADDRESS AND DATA FIELD SEPARATION

In systems connected together on a serial bus without individual chip-selects for individual elements connected to this bus, serial transmission must convey not only data but the address of the receiver element to which data is sent. Since all transmissions are byte long, recognition of address from a data pattern is not possible unless the address can be distinguished from data by a start bit. In many standard accepted systems, an address field follows a start condition which is then followed by a number of data fields depending on the transaction relative to that address.

Detection of this start condition sets SPICR2, hence at the end of an 8-bit transmission it is possible to check if the received byte corresponds to address or data fields. It must be emphasized that a start condition does not occur normally during the transmission but it is provoked by the transmitter, prior to address field transmission.

Secondly, zero-to-one transition of SPICR2 caused by the start condition causes the divide-by-eight counter to be cleared, hence allows all receiver MCUs to be synched-up simultaneously.

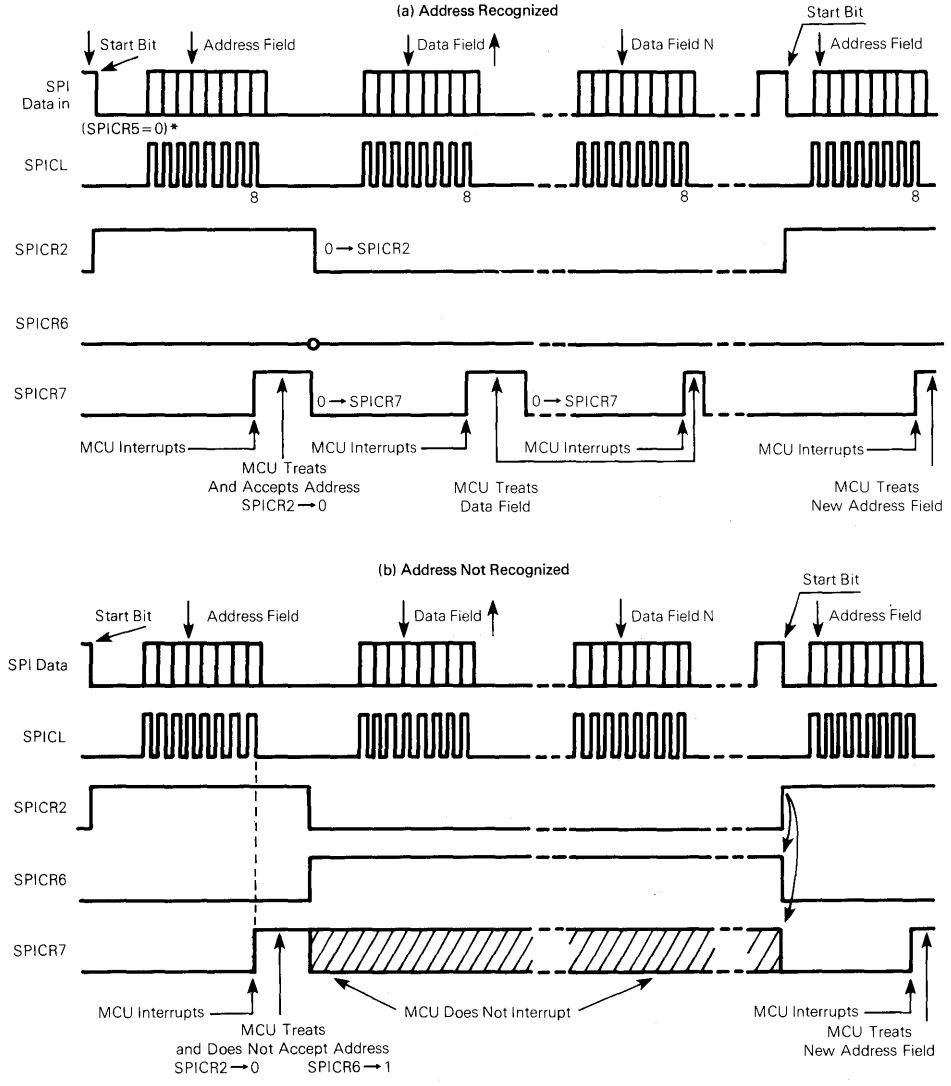
The third important consideration is the rate of occupation of the MCU in serving the information flow on the serial bus with respect to the background tasks.

In case of high-speed transmissions (up to 100K Baud) and heavy information flow not related to a given MCU on the serial data bus, it is possible, if no precaution is taken, that a non-selected receiver MCU has to analyze every field; data or address, to check for a particular address field that is of concern. This causes a very high interruption rate to service the SPI and leaves very little time for background tasks. In order to mask an undesirable data field transmission, that requires interrupt driven analysis, SPICR6 may be set and SPICR2 can be cleared after analyzing an invalid address field. Then, the MCU becomes immune to all SPI interrupt requests due to subsequent data fields. On the next start bit preceding a new message, SPICR2 is set which in turn causes SPICR7 and SPICR6 to be cleared. The MCU is then ready to service an incoming new address field via interrupt.

Refer to Figure 4-8 which illustrates the time for SPI address and data field separation (reception).

4.9 DATA FIELD ONLY OPERATION

In applications where: 1) only data patterns are transmitted or 2) the effect of the rising edge of SPICR2 having cleared SPICR6, SPICR7, and divide-by-eight counter needs to be inhibited, it is sufficient to set SPICR2 under program control before transmission. SPICR7 and SPICR6 are not cleared by the software controlled setting of SPICR2.



*As defined during SPI deselect mode (SPICR4=0).

Figure 4-8. SPI Address and Data Field Separation (Reception)

4.10 DATA ARBITRATION

Data arbitration occurs when two or more transmitters try to control a common data line. Refer to Figure 4-9 for data arbitration timing.

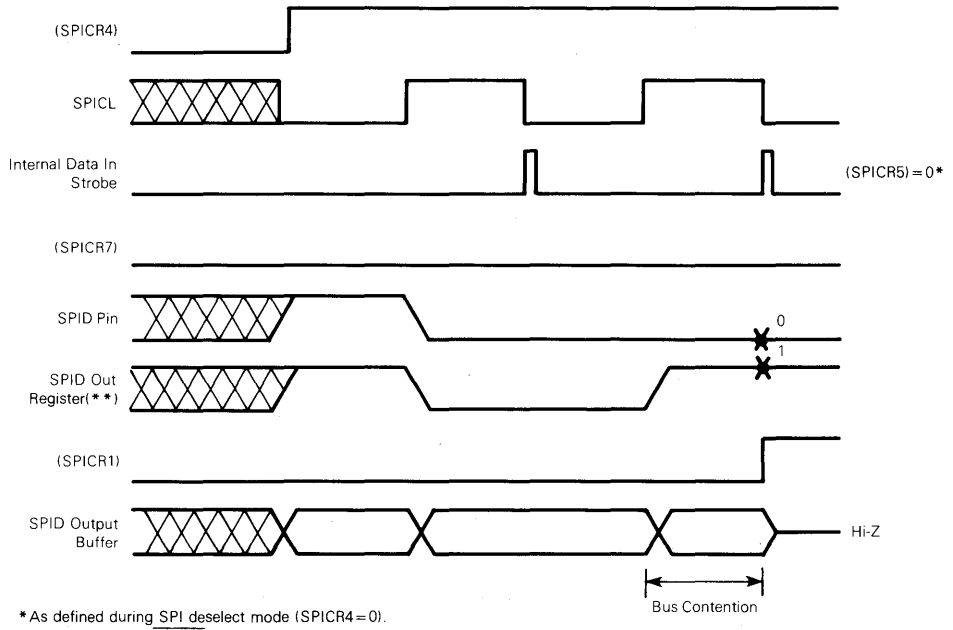


Figure 4-9. SPI Data Arbitration Timing Diagram

The MCU handles the data arbitration in the following ways:

Starting Conditions

1) The MCU has data mastership, i.e., port B2 or B3 are used for SPI data transfer and have their data direction registers in the output mode.
2) SPICR3 is preset properly to output the SPI data on the selected data output port (PB2 or PB3)
3) The SPI is in the active transaction mode.

Arbitration Criterion

The SPI data output line logic level on the pin is compared with contents of the data register of that line during the data input strobe. If the data register content is one while the SPI data output line logic level is zero then it is decided that an external device(s) is (are) trying to control the data line.

Action

When the arbitration criteria are met, the mode fault flag (SPICR1) is set, the MCU loses data mastership and the SPI data output line DDR is cleared putting the line in the input mode.

NOTE

Complementary type of arbitration (i.e., output data line equals one; port data register equals zero) is not implemented and should not occur in the system as this will cause excessive dissipation on the port and may result in a catastrophic failure of the circuit.

4.11 CLOCK ARBITRATION

Clock arbitration is done in two ways: 1) via the slave select input line and 2) via the serial peripheral interface clock line. Both types of arbitration may be used simultaneously in an application.

4.11.1 Clock Arbitration via Slave Select Input Line

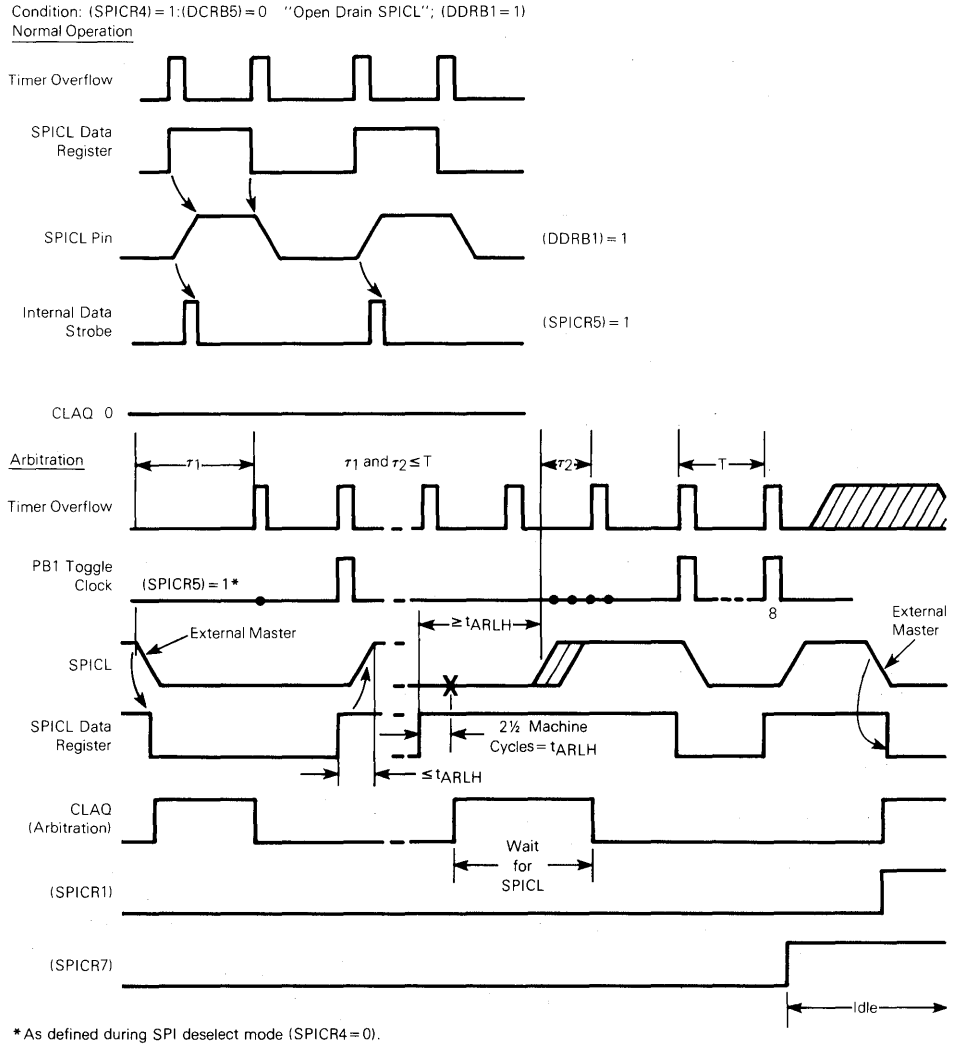
During serial peripheral interface transactions, port B0 serves as the slave select input if port B0 is in the input mode (DDR B0 = 0).

When the MCU has clock mastership, PB0 should remain high. When an external device requests clock mastership this input is pulled low. The MCU loses clock mastership and switches to slave type operation, the clock line data direction register bit is cleared, (DDR B1 = 0), and the mode fault flag is set.

This clock arbitration may happen during active or idle transaction modes (see **4.12 SLAVE SELECT INPUT OPERATION**).

4.11.2 Clock Arbitration via Serial Peripheral Interface Clock Line

This type of arbitration is enabled only when the MCU operates as clock master while the clock line output buffer works in the open-drain mode (DCR B5 = 0). Unlike the clock arbitration described previously, the MCU does not lose clock mastership. The clock output data register status is monitored under control of the clock arbitration flip-flop to guarantee minimum clock high and clock low times on the clock line, in case two or more clock masters are trying to control the clock line simultaneously. Each clock master may be assumed to be asynchronous with respect to the other(s) and to run with different clock frequencies. When set, the clock arbitration flip-flop (CLAQ) blocks the toggle enable of port B1 effectively inhibiting the port data register from changing state by toggling during the toggle pulse. Refer to Figure 4-10 for timing.



*As defined during SPI deselect mode (SPICR4 = 0).

Figure 4-10. Clock Arbitration via Clock Line Timing

CLAQ status is modified under the following conditions:

a) CLAQ is cleared when:
   1) SPICR 4=0 or DDR B1=1
   2) Toggle pulse is generated for port B1.

b) CLAQ is set when:
   1) A negative edge is detected on the SPI clock input if the port B1 data register is high. Simultaneously, the port B1 data register is cleared. If this occurs in the idle mode, while the MCU is not ready for serial transmission, the mode fault flag (SPICR 1) is set as well. In this way, the MCU will keep the clock line low, effectively blocking all clock pulses on the clock line, and detecting that the clock line was driven low during the idle mode. If the MCU was set up as a transmitter, the clock edge occurring during the idle mode cannot generate an internal data output strobe. Hence, during subsequent serial transmission receivers it would "miss" the MSB of the data transmitted from the MCU. Protocols can be set up to avoid, or recover from, this type of framing error.
   2) If the SPI clock line is still low 2½ machine cycles after the port B1 data register is set, the CLAQ set command will remain active, as long as the SPI clock line remains low. The clock arbitration operation is explained in more detail in **4.13.3 Two-Wire Half Duplex Mode with Clock Arbitration.**

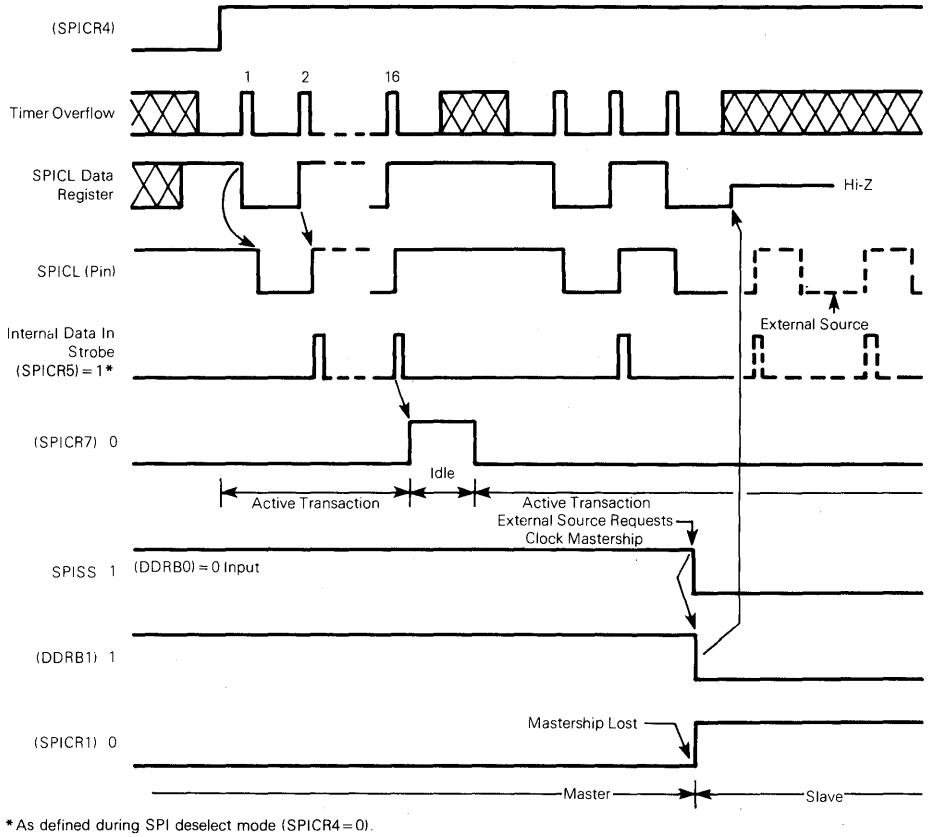
4.12 SLAVE SELECT INPUT OPERATION

Slave select information is supplied to the MCU via port B0 by an external device. If port B0 is in the output mode then slave select actions are inhibited. If the slave select feature is not used in an application, port B0 should be used in the output mode.

Slave select input generates various actions depending on whether the SPI is operating in the clock master mode or clock slave mode. These are outlined in the following paragraphs.

4.12.1 Slave Select Input Actions During Master Mode

In this mode, the slave select input is monitored to assure that it stays false (high). If slave select becomes true (low), the device immediately exits the master mode and becomes a slave (DDR B1=0). The significance of this is that a collision has occurred; that is, two devices have both become or are willing to become masters. This is normally the result of a software error, although some systems may allow the default master to "knock all other masters off the bus" if an erroneous bus state is detected. This is a castastrophic event and it is the responsibility of the default master to completely "clean up" the system. Moreover, the mode fault flag is set to signal to the MCU that clock mastership is lost. These actions can take place during either active or idle transaction modes. Refer to Figure 4-11.



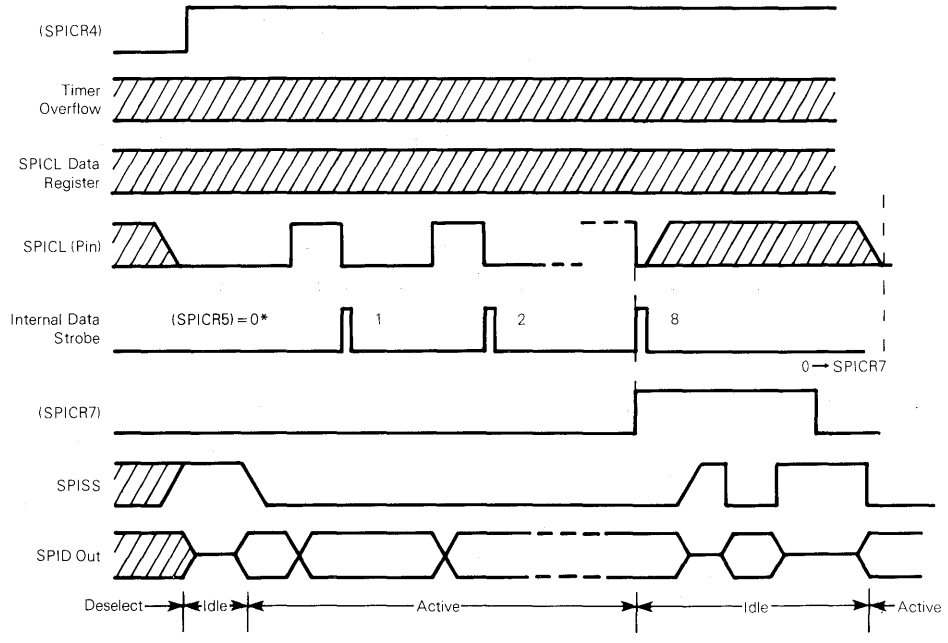
*As defined during SPI deselect mode (SPICR4 = 0).

Figure 4-11. Clock Arbitration via Slave Select Input — Master Mode

4.12.2 Slave Select Input Actions During Slave Mode

The slave select (SS) input is generated by the current clock master (parallel port may be used) and used to enable one of several possible slaves to accept and/or return data. The SS signal must be low prior to occurrence of serial clock pulses and must not become high until the eighth (last) serial clock cycle. A high level on SS forces serial data output to the high-impedance state without affecting the data direction register status relative to the data output. Also, when SS is high the serial clock input pulses (if any) are inhibited from generating internal data output and input strobe pulses, and also the eight-bit counter is cleared.

The significance of this is that the slave select acts as a chip-enable line and the MCU receives and/or is allowed to transmit back information only when SS is pulled low by the current clock master. Individual lines must be used from the master for each slave select input. A single line is sufficient in the case of daisy chain or cascade connection of multiple slaves. Refer to Figure 4-12.



*As defined during SPI deselect mode (SPICR4 = 0).

Figure 4-12. Clock Arbitration via Slave Select Input — Slave Mode

4.13 SPI OPERATING MODES

A brief description of the serial peripheral interface (SPI) operating modes is contained in the following paragraphs.

4.13.1 One-Wire — Autoclocked Mode

In this mode, various circuits are connected to each other via a single wire one which data transfer takes place. The clock is implicit during transmission and each circuit is its own clock master. The MCU should be initialized as clock master and port B1 is not connected externally. In order to achieve the precise timing required for this transmission it may be useful to start the active transaction mode with an interrupt. Hence, the data input/output line can be connected on the MCU to the $\overline{INT2}$ line.

With the assistance of software to generate the start bit and stop bits, and swap the order of bits in the data, NRZ-type serial transmission compatible with MC6801 can be achieved in this mode. (See Figures 4-13 and 4-14.) Unused SPI data port B2 or B3 may be used as a normal input/output. Port B0 may be used only as an output.

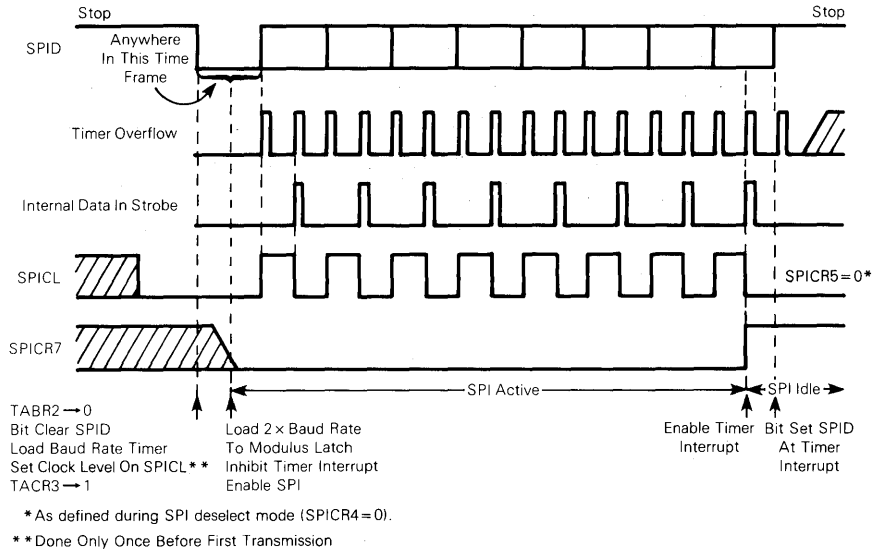


Figure 4-13. SPI — NRZ Operation (Transmit) Timing

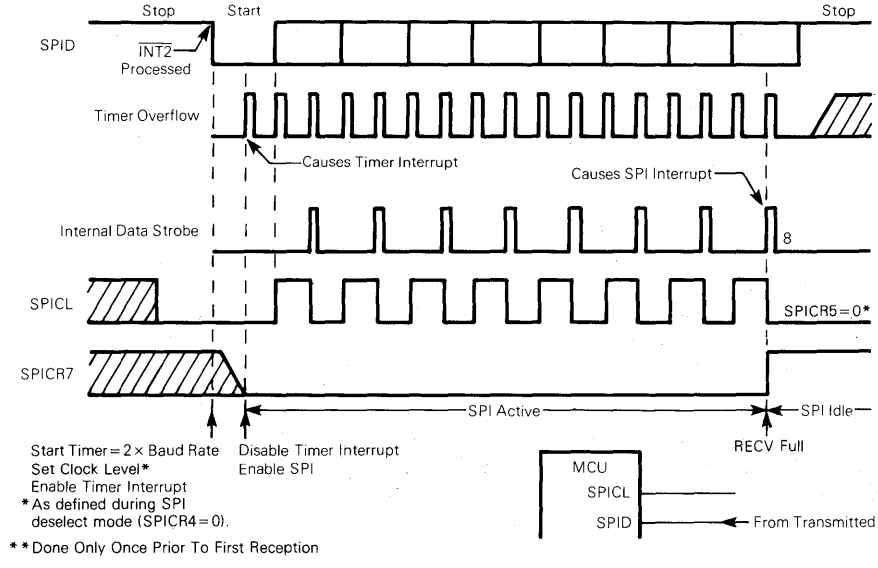


Figure 4-14. SPI NRZ Operation Timing (Receive)

4.13.2 Two-Wire Half-Duplex Mode

In this mode, the data and clock lines are connected between various circuits in the system. Data and clock mastership should be monitored via protocol included in the data patterns transmitted between circuits. Moreover, data arbitration is possible on the MCU data line. Any transmitter can "knock out" all others by transmitting all zeros.

4.13.3 Two-Wire Half-Duplex Mode with Clock Arbitration

In this mode, the MCU is assumed to operate as a clock master with an open-drain SPI clock output buffer. Clock and data arbitration is accomplished as explained in **4.11 CLOCK ARBITRATION.** More than one clock master (and transmitter) is allowed at the same time in this mode.

An interesting protocol occurs when the clock lines of all masters operate with open-drain outputs. If no master other than the MCU is operating on the clock line, then the clock arbitration flip-flop (CLAQ) is never set and every toggle pulse creates an edge on the SPI clock line (SPICL). This is the normal mode of operation.

However, if an external master pulls the SPI clock line low, the MCU sets CLAQ to inhibit the next timer overflow from generating a toggle pulse on the SPI clock port. The SPI clock port data register is also cleared. At the next timer overflow, CLAQ is reset and the SPI clock port is allowed to toggle during future timer overflows. In the meantime, other master clock outputs may go high. However, the SPI clock line is held low by the MCU until a low-to-high transition occurs on its SPICL data register line. (In wire-or configuration, any master with a low output imposes a low clock line on the total system.)

This mechanism guarantees that in case of clock arbitration (a process which is asynchronous to the timer overflows) the SPI clock low time is not shorter than one toggle period. Hence, narrow negative glitches are avoided on the clock line. Some devices in the system may be operated totally under software control by using polling techniques. Polling is generally much slower than hardwired logic. Potential appearance of narrow glitches could cause castastrophic system faults, as some devices in the system might respond to them and some might not.

The clock arbitration flip-flop is also set when the SPICL data register toggles high while an external master keeps the SPI clock line low after two and one-half machine cycles. CLAQ remains set until the SPICL line returns to a high state. At the next timer overflow, CLAQ is reset. Future timer overflows will be allowed to toggle the SPICL data register to the low state.

This mechanism guarantees that in case of a clock arbitration situation, the SPI clock high time is not shorter than one toggle period. This avoids narrow positive glitches. The same comments are applicable to positive glitches with regard to system performance.

In such a system, the longest clock low time is imposed by the clock master with the longest clock low time. The shortest high time is determined by the device with the shortest high clock time.

4.13.4 Three-Wire Half-Duplex Mode with Slave Select Input

This mode is similar to the two-wire half-duplex mode except that the slave select input provides the possibility of using the MCU as a peripheral circuit in a system (or in systems) where clock mastership may be passed through the slave select line.

A typical method of doing this is to wire the slave select lines together. The current master puts its slave select line (SPISS) in the output mode prior to a serial transmission and pulls the SPISS line low signifying that the system is busy. In this way, the clock master will keep its mastership until the end of the transmission. Software protocol can be arranged such that slaves do not request mastership until their SPISS lines go high. At the end of a transmission, the current master pulls the SPISS line high and puts its SPISS port (PB0) in the input mode. A slave requesting clock mastership can now pull the SPISS line low, "knocking out" the current master. To avoid simultaneous mastership requests, time multiplexed protocols may be required.

4.13.5 Three-Wire Full-Duplex Mode

In this mode, the MCU can operate as a transmitter and receiver at the same time. Bus oriented or daisy chain type networks are feasible. Protocols included in the data stream are required to change the clock masters, number of transmitters in the system, or the direction of information flow in daisy chained systems with "collision." In this mode, it is possible for the MCU to shift out one byte while receiving another. This removes the need for XMIT EMPTY or REC FULL status bits. Refer to Figure 4-15.

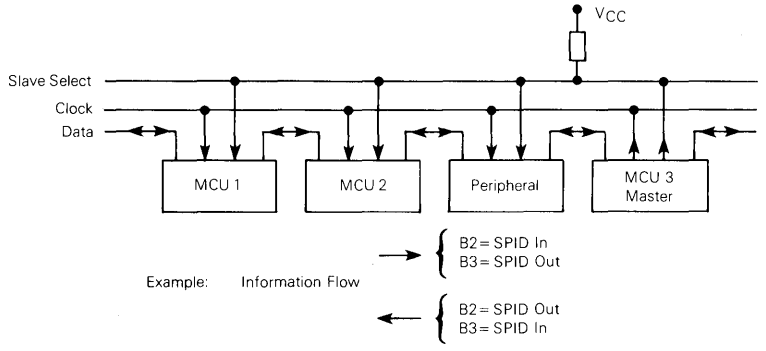


Figure 4-15. Daisy Chain/Cascade Organization

4.13.6 Three-Wire Full-Duplex Mode with Clock Arbitration

This mode is a mix of the three-wire full-duplex mode and the two-wire half-duplex mode with clock arbitration, where the SPI clock line operates in a wire-or fashion in the system. Simultaneous masters are allowed and clock arbitration is accomplished via the clock line.

4.13.7 Four-Wire Full-Duplex Mode with Slave-Select Input

This mode is similar to the three-wire full-duplex mode with regard to network and to the three-wire half-duplex mode with slave-select input in respect to clock arbitration and slave selection. Refer to Figure 4-16.

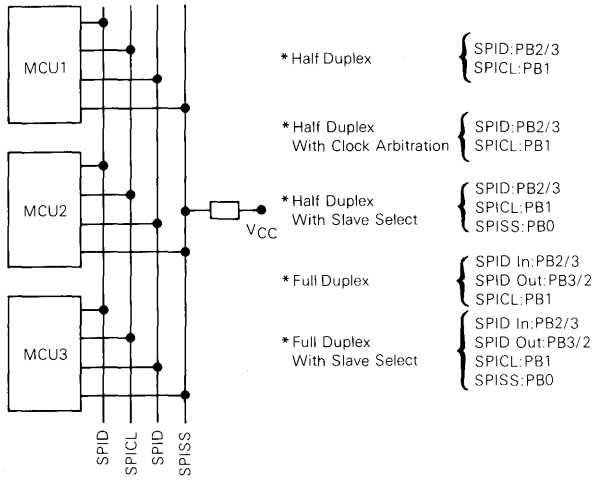


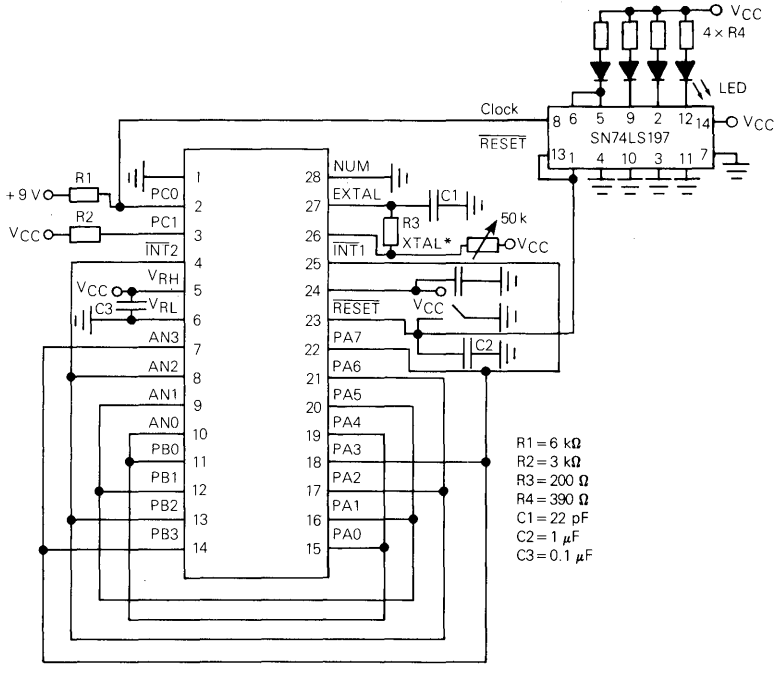
Figure 4-16. SPI Operation Bus Organization

SECTION 5
SELF-CHECK, RESETS, CLOCK GENERATOR OPTIONS, AND INTERRUPTS

This section describes the self-check capability, resets, clock generator options, and interrupts.

5.1 SELF-CHECK

The self-check capability of the MC6805S2 MCU provides an internal check to determine if the part is functional. Connect the MCU as shown in Figure 5-1 and monitor the output of port C bit 0 for an oscillation of approximately 7 Hz. A 9-volt level on PC0, pin 2, detected as the device under test comes out of reset, energizes the ROM-based self-check feature. The self-check program exercises the CPU, RAM, ROM, A/D, timers, interrupts, I/O ports, and auxiliary counter.



*RC Oscillator Option Shown If Q0-Q2 LEDs Blinking = Device Passes Test
Q3 Blinking = Watchdog Reset Problem

Figure 5-1. Self-Check Connections

Several of the self-check subroutines can be called by a user program with a JSR or BSR instruction. They are the RAM, ROM, and 4-channel A/D tests. The timer routine may also be called if the timer input is the internal phase two clock.

5.1.1 RAM Self-Check Subroutine

The RAM self-check is called at location $F39 and returns with the Z bit clear if any error is detected; otherwise, the Z bit is set. The walking diagnostic pattern method is used.

The RAM test must be called with the stack pointer at $07F. When run, the test checks every RAM cell except for $07F and $07E which are assumed to contain the return address.

The A and X registers and all RAM locations except the top two are modified.

5.1.2 ROM Checksum Subroutine

The ROM self-check is called at location $F54 and returns with the Z bit cleared if any error was found; otherwise Z = 1, X = 0 on return, and A = 0 if the test passed. RAM locations $040-$043 are overwritten.

5.1.3 Analog-to-Digital Converter Self-Check

The A/D self-check is called at location $F6E and returns with the Z bit cleared if any error was found; otherwise Z = 1.

The A and X register contents are lost. The X register must be set to four before the call. On return, X = 8 and A/D channel 7 is selected. The A/D test uses the internal voltage references and confirms port connections.

5.1.4 Timer Self-Check Subroutine

The timer self-check is called at location $F99 and returns with the Z bit cleared if any error was found; otherwise Z = 1.

In order to work correctly as a user subroutine, the internal phase two clock must be the clock source and interrupts must be disabled. Also, on exit, the clock is running and the interrupt mask is not set so the caller must protect from interrupts if necessary.

The A and X register contents are lost. The timer self-check routine counts how many times the clock counts in 128 cycles. The number of counts should be a power of two since the prescaler (1) is a power of two. If not, the timer probably is not counting correctly. The routine also detects if timer A is not running.

5.2 RESETS

The MCU can be reset four ways: by initial power up, by the external reset input ($\overline{\text{RESET}}$), by a forced reset generated by a timeout of the MCUs auxiliary or "watchdog" counter, and by an optional internal low voltage detect circuit. The $\overline{\text{RESET}}$ input consists mainly of a Schmitt trigger which senses the $\overline{\text{RESET}}$ line logic level. A typical reset Schmitt trigger hysteresis curve is shown in Figure 5-2. The Schmitt trigger provides an internal reset voltage if it senses a logic zero on the $\overline{\text{RESET}}$ pin. Refer to the reset circuit in Figure 5-3 and to Figure 5-9, under **5.4 INTERRUPTS,** for the complete reset sequence.

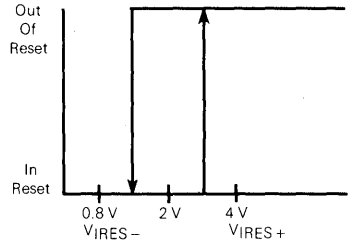


Figure 5-2. Typical Reset Schmitt Trigger Hysteresis

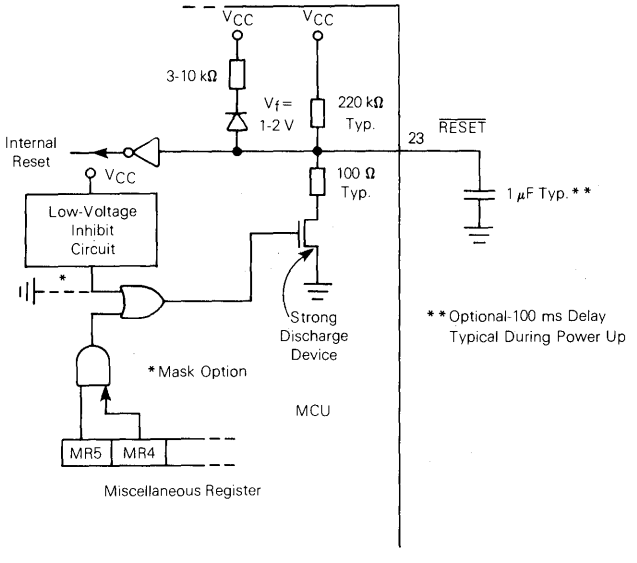


Figure 5-3. Reset Circuit

5.2.1 Power-On Reset (POR)

An internal reset is generated upon power up that allows the internal clock generator to stabilize. A delay of t_{RHL} milliseconds is required before allowing the $\overline{RESET}$ input to go high. Refer to the power and reset timing diagram of Figure 5-4. Connecting a capacitor to the $\overline{RESET}$ input (as illustrated in Figure 5-5) typically provides sufficient delay. During power up, the Schmitt trigger switches on (removes reset) when the $\overline{RESET}$ rises to V_{IRES+}.

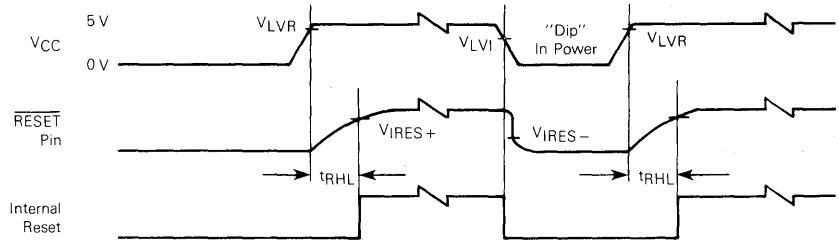


Figure 5-4. Power and Reset Timing

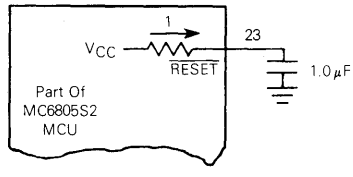


Figure 5-5. Power-Up Reset Delay Circuit

5.2.2 External Reset Input

The MCU will be reset if a logic zero is applied to the $\overline{RESET}$ input for a period longer than one machine cycle (t_{CYC}). Under this type of reset, the Schmitt trigger switches off at V_{IRES-} to provide an internal reset voltage.

5.2.3 Low Voltage Inhibit (LVI)

The optional low-voltage detection circuit causes a reset of the MCU if the power supply voltage falls below a certain level (V_{LVI}). The only requirement is that V_{CC} remains at or below the V_{LVI} threshold for one t_{CYC} minimum. In typical applications, the V_{CC} bus filter capacitor will eliminate negative-going voltage glitches of less than one t_{CYC}. The output from the low-voltage detector is connected directly to the internal reset circuitry. It also forces the $\overline{RESET}$ pin low via a strong discharge device through a resistor. The internal reset will be removed once the power supply voltage rises above a recovery level (V_{LVR}), at which time a normal power-on-reset occurs.

5.2.4 Forced Reset

If the auxiliary counter reset mask bit in the miscellaneous counter (MR4) is clear and the auxiliary counter status bit (MR5) is set, as a result of counter overflow, a switch to V_{SS} is turned on, pulling the $\overline{RESET}$ pin low. A consequent voltage drop below V_{IRES-} on $\overline{RESET}$ causes an MCU reset, which in turn sets MR4. Switching to V_{SS} when the $\overline{RESET}$ pin is turned off allows voltage to rise above V_{IRES+}, after which the MCU reset is released.

$\overline{RESET}$ pin voltage variations occurring as a result of forced reset may be amplified externally in order to provide a reset to other peripheral circuits in the system. The reset output from the MCU is not TTL compatible.

5.2.5 Reset Initialization

The minimum low time for all four modes of reset is one t_{CYC} + 250 nanoseconds (t_{CYC} = oscillator frequency divided by four). When reset is detected, the MCU initialization takes place. The following are the actions taken on the internal circuitry:

a) FF	Timer A Modulus Latch and Timer A	i) 40	Serial Peripheral Interface Control Register
b) FFFF	Timer B	j) 00	Port A Data Direction Register*
c) 7F	Prescaler 1	k) FC	Port C Data Direction Register*
d) 7FFF	Prescaler 2	l) F0	Port B Data Control Register*
e) 50	Timer A Control Register	m) 1	Interrupt (Mask Bit I in Condition Code Register)
f) 50	Timer B Control Register	n) 7F	Stack Pointer
g) 50	Miscellaneous Register	o) FFE	Program Counter
h) 07	A/D Status Control Register		

5.3 INTERNAL CLOCK GENERATOR OPTIONS

The internal clock generator circuit is designed to require a minimum of external components. A crystal, a resistor, a jumper wire, or an external signal may be used to control the internal clock generator with various stability/cost tradeoffs. A manufacturing mask option is used to select the crystal or resistor option. The oscillator frequency is internally divided by four to produce the internal system clocks.

The different connection methods are shown in Figure 5-6. The crystal specifications and suggested PC board layout are given in Figure 5-7. A resistor selection graph is shown in Figure 5-8.

The crystal oscillator start-up time is a function of many variables: crystal parameters (especially R_S), oscillator load capacitance, IC parameters, ambient temperatures, and supply oscillator start-up. Neither the crystal characteristics nor the load capacitances should exceed recommendations.

---

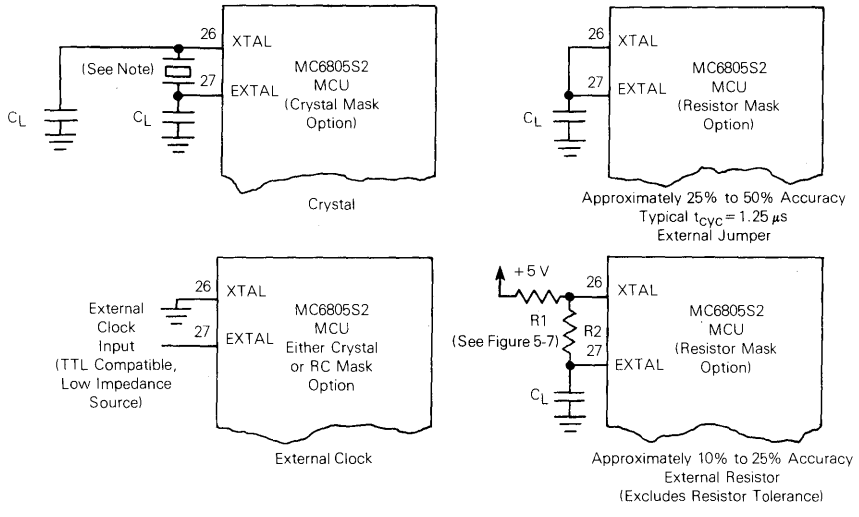
* Reads as $FF

When utilizing the on-board oscillator, the MCU should remain in a reset condition (reset pin voltage below V_{IRES+}) until the oscillator has stabilized at its operating frequency. Several factors are involved in calculating the external reset capacitor required to satisfy this condition: the oscillator start-up voltage, the oscillator stabilization time, the minimum V_{IRES+}, and the reset charging current specification.

Once V_{CC} minimum is reached, the external $\overline{RESET}$ capacitor will begin to charge at a rate dependent on the capacitor value. The charging current is supplied from V_{CC} through a large resistor, so it appears almost like a constant current source until the reset voltage rises above V_{IRES+}. Therefore, the $\overline{RESET}$ pin will charge at approximately:

$$(V_{IRES+}) \bullet C_{ext} = I_{RES} \bullet t_{RHL}$$

Assuming the external capacitor is initially discharged.



NOTE:
The recommended C_L value with a 4.0 MHz crystal is 27 pF, maximum, including system distributed capacitance. For crystal frequencies other than 4 MHz, the total capacitance on each pin should be scaled as the inverse of the frequency ratio. For example, with 2 MHz crystal, use approximately 50 pF on EXTAL and approximately 50 pF on XTAL. The exact value depends on the Motional-Arm parameters of the crystal used.

Figure 5-6. Clock Generator Options

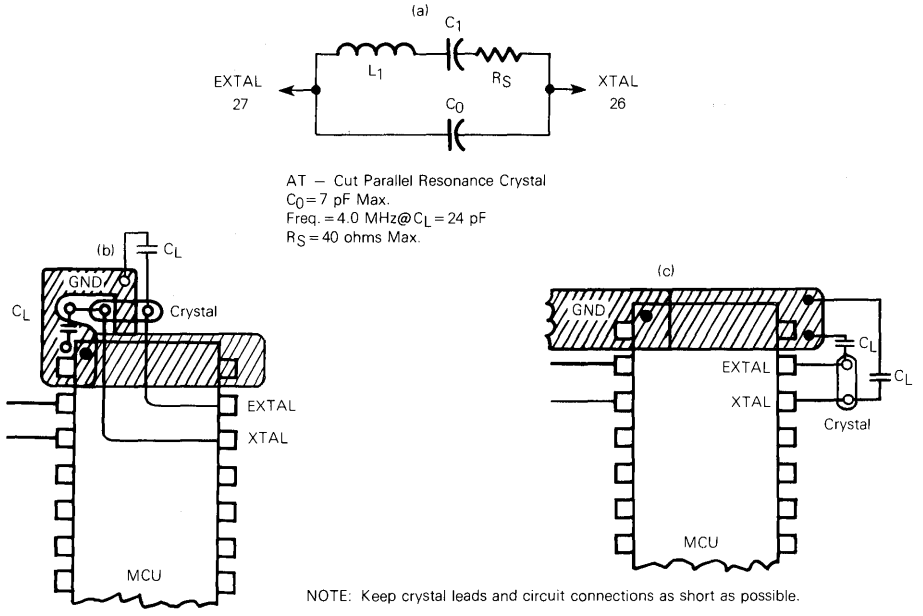


Figure 5-7. Crystal Motional Arm Parameters and Suggested PC Board Layout

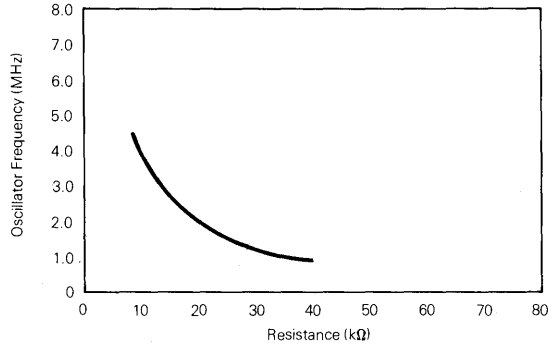


Figure 5-8. Typical Frequency Selection for Resistor Oscillator Option

5.4 INTERRUPTS

The MC6805S2 MCU can be interrupted seven different ways: at reset, through the external interrupt ($\overline{INT1}$) input pin, the internal timer (either A or B) interrupt request, the SPI interrupt request, the external port D bit 6 ($\overline{INT2}$) input pin, and a software interrupt instruction (SWI).

The reset interrupt has priority over all other interrupts and is not maskable. It is serviced immediately at its occurrence independent of the instruction being executed (see **5.2 RESETS**). All other interrupts are maskable and do not cause the current instruction execution to be halted, but are considered pending until the current instruction execution is complete. Pending $\overline{INT1}$, $\overline{INT2}$, timer A, timer B, or SPI interrupts are acknowledged by the MCU only if the I bit in the condition code register is clear.

When any interrupt (except reset) is acknowledged, processing is suspended following completion of the current instruction being executed, the present MCU state is pushed onto the stack, the interrupt bit (I bit) in the condition code register is set, the address of the interrupt routine is obtained from the appropriate interrupt vector address, and the interrupt routine is executed. Stacking the CPU registers, setting the I bit, and vector fetching requires a total of 11 t_{cyc} periods for completion. Note that interrupts which are masked are latched internally for later interrupt service once the mask bit(s) is (are) cleared. Refer to Figure 5-9 for a flowchart. The interrupt service routine must end with a return from interrupt (RTI) instruction which allows the MCU to resume processing of the program prior to the interrupt.

Table 5-1 provides a listing of the interrupts, their priority, and the address of the vector which contains the starting address of the appropriate interrupt service routine. This priority applies to those interrupts pending when the CPU is ready to accept an interrupt. In addition, each of these interrupts, except $\overline{INT1}$, have a separate mask bit which must also be cleared, in addition to the I bit, for the MCU to acknowledge the interrupt. Specifically, the $\overline{INT2}$, timer A, timer B, and SPI interrupts each have their own independent mask bits contained in MR6, TACR6, TBCR6, and SPICR6, respectively.

NOTE

The timer A, timer B, $\overline{INT2}$, and SPI interrupts share the same vector address. The interrupt routine must determine the source of the interrupt by examining the interrupt request bits, namely TACR7, TBCR7, MR7, and SPICR7. These bits are not automatically cleared following interrupt servicing and must be cleared via software. The $\overline{INT1}$ interrupt has its own unique vector address. Therefore, the $\overline{INT1}$ interrupt request is cleared automatically when the $\overline{INT1}$ vector is serviced.

Table 5-1. Interrupt Priorities

Interrupt	Priority	Vector Address
$\overline{RESET}$	1	$FFE and $FFF
SWI	2*	$FFC and $FFD
$\overline{INT1}$	3	$FFA and $FFB
TIMER/$\overline{INT2}$/SPI	4	$FF8 and $FF9

*Priority 2 applies when the I bit in the condition code register is set. When I=0, SWI has a priority of four (like any other instruction). The priority of $\overline{INT1}$ thus becomes two and the TIMER/$\overline{INT2}$/SPI becomes three.

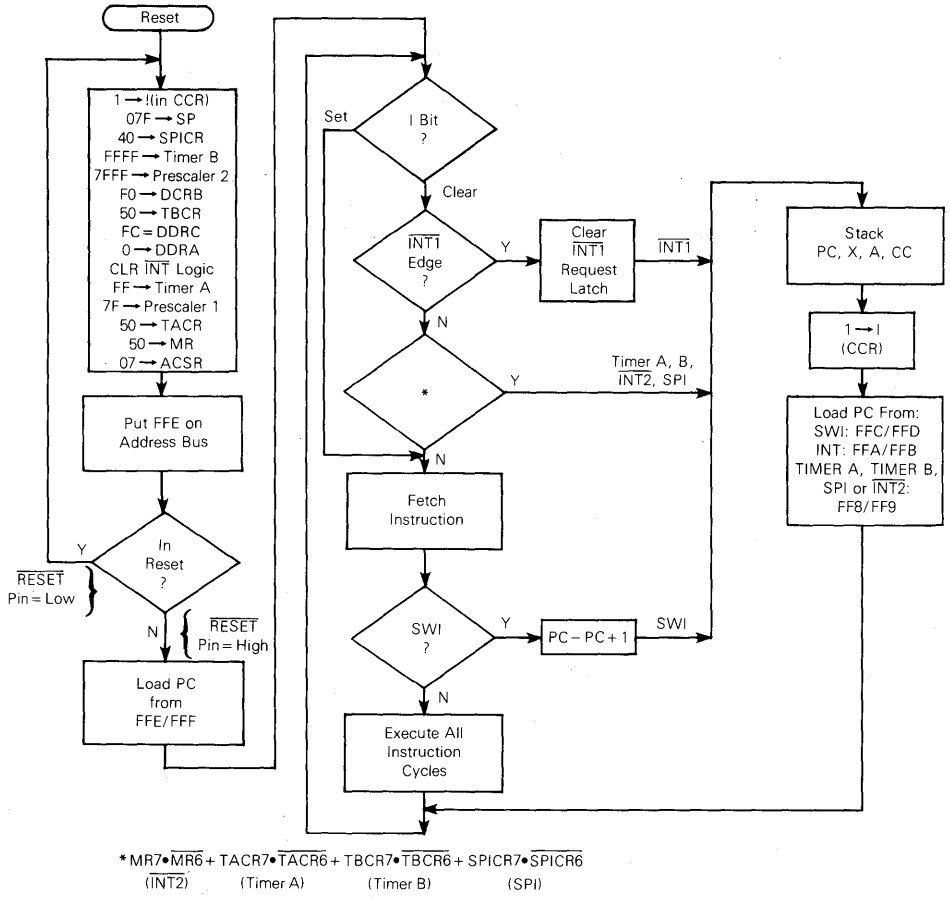


Figure 5-9. Reset and Interrupt Processing Flowchart

A software interrupt (SWI) is an executable instruction which is executed regardless of the state of the I bit in the condition code register. SWIs are usually used as breakpoints for debugging or as system calls.

The external interrupts, $\overline{INT1}$ and $\overline{INT2}$, are latched and/or sensed on the falling edge of the input signal. Timer A and B interrupt request bits are set when these timers make transition to $00 and $0000, respectively.

A sinusoidal input signal (f_{INT} maximum) can be used to operate an external interrupt ($\overline{INT1}$), as shown in Figure 5-10, for use as a zero-crossing detector with hysteresis included. An interrupt request is generated for each negative-slope zero crossing of the ac signal. This allows applications such as servicing time-of-day routines and engaging/disengaging ac power control devices. Off-chip full wave rectification provides an interrupt at every zero crossing of the ac signal and thereby provides a 2f clock.

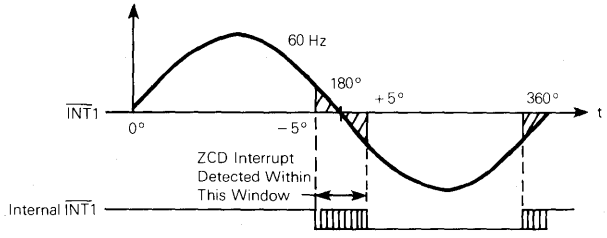
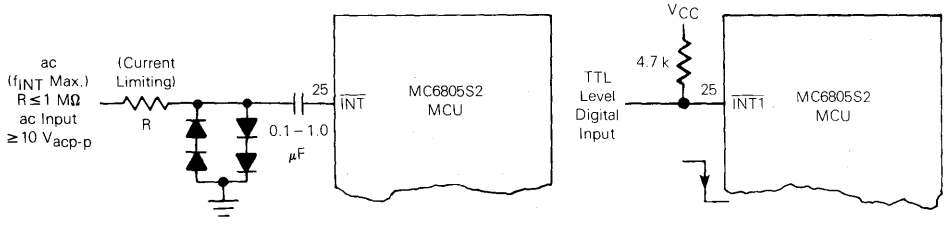


Figure 5-10. Interrupt Timing

For digital applications, $\overline{INT1}$ can be driven directly by a digital signal. The maximum frequency of a signal that can be recognized by the timer and $\overline{INT1}$ pin logic is dependent on the parameters labeled t_{WL} and t_{WH}. The pin logic that recognizes the high (or low) state on the pin must also recognize the low (or high) state on the pin, in order to re-arm the internal logic. Therefore, the period can be calculated as follows: (assumes 50/50 duty cycle for a given period)

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

The period is not simply t_{WL}, t_{WH}. This computation is allowable, but it does reduce the maximum allowable frequency by defining an unnecessarily longer period (250 ns twice). See Figure 5-11. For the $\overline{INT1}$ function, the maximum allowable frequency is also determined by the software response of the $\overline{INT1}$ service routine.



(a) Zero-Crossing Interrupt

(b) Digital-Signal Interrupt

Figure 5-11. Typical Interrupt Circuits ($\overline{INT1}$)

SECTION 6
INPUT/OUTPUT PORTS AND ANALOG-TO-DIGITAL CONVERTERS

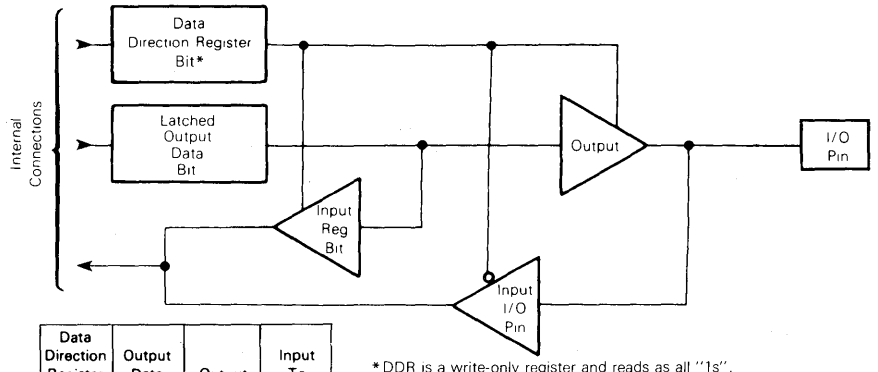
This section describes the input/output pins, the port data registers, the miscellaneous register, and the analog-to-digital converter.

6.1 INPUT/OUTPUT

There are 14 input or input/output pins. The $\overline{INT1}$ pin may also be polled with branch instructions to provide an additional input pin. All pins on ports A, B, and C are programmable as either inputs or outputs under software control of the corresponding data direction registers (DDRs). The port I/O programming is accomplished by setting the corresponding bit in the port DDR to a logic one for output or a logic zero for input. On reset all the DDRs are initialized to a logic zero state, placing the ports in the input mode. The port output registers are not initialized on reset and should be initialized by software before changing the DDRs from input to output. When programmed as outputs, all I/O pins read latched output data, regardless of the logic levels at the output pin due to output loading; refer to Figure 6-1 and Table 6-1.

CAUTION

The port data registers are not initialized during reset. The contents of these registers should be written to a known state for any port pins that are expected to become outputs. This will avoid any spurious transitions before initializing the corresponding DDR bits to the output mode.



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

*DDR is a write-only register and reads as all "1s".
**Ports A (with CMOS drive disabled), B, and C are three-state ports. Port A has optional internal pullup devices to provide CMOS drive capability. See Electrical Characteristics tables for complete information.

Figure 6-1. Typical Port I/O Circuitry

Table 6-1. Digital Input/Output Ports

Name	Number	Input	Output	Input		Output		Comment
				TTL	Special	TTL	CMOS	
Port A	8	Yes	Yes	*		*	*(a)	a: If Pull-Up Option
Port B LED Drive	4	Yes	Yes	*(b)		*(a)		a: 10 mA Sink; Current Limited Source. PB1-PB3 can be programmed to open-drain configuration via PB DCR. b: Hi-Z Input
Port C	2(b)	Yes	Yes	*(a)		*		a: Hi-Z Input b: Shared with EXT Timer Inputs
Port D	7(a)	Yes	No	*				a: PD5 and PD4 Share a 15 kilohm resistor (typ)
$\overline{INT1}$	1	Yes	No		*			BIL/BIH Instruction

All input/output lines are TTL compatible as both inputs and outputs. Port A lines are CMOS compatible as outputs using a mask option. Ports B1, B2, and B3 can be software programmed to operate as open-drain outputs. Port B, C, and D lines are CMOS compatible as inputs. When programmed as outputs, port B is capable of sinking 10 milliamperes and sourcing 1 milliampere on each pin (TTL output state). Port D lines are input only; thus, there is no corresponding DDR.

Port D provides the multiplexed analog inputs, reference voltages, and $\overline{INT2}$. All of these lines are shared with the port D digital inputs. PD0-PD3 may always be used as digital inputs and may also be used as analog inputs. The V_{RL} and V_{RH} lines (PD4 and PD5) are internally connected by the A/D resistor. Analog inputs may be prescaled to attain the V_{RL} and V_{RH} recommended input voltage range.

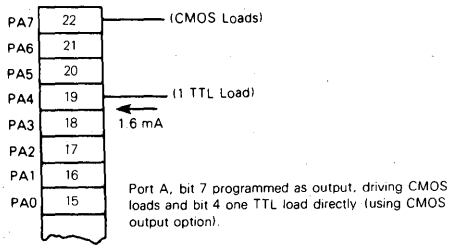
Figure 6-2 provides some examples of port connections. The address map in Figure 2-1 gives the addresses of data registers and DDRs.

CAUTION

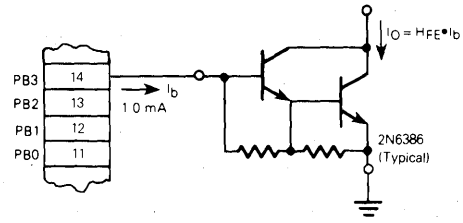
The corresponding DDRs for ports A, B, and C are write-only registers (locations $004, $005, and $006). A read operation on these registers reads as all ones. Since BSET and BCLR are read-modify-write functions, they cannot be used to set a DDR bit (all unaffected bits would be set). It is recommended that all DDR bits in a port be written using a single-store instruction.

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

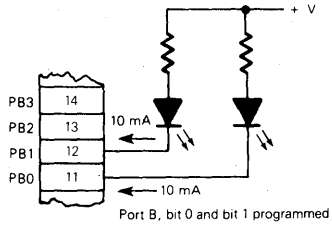
(a) Output Modes



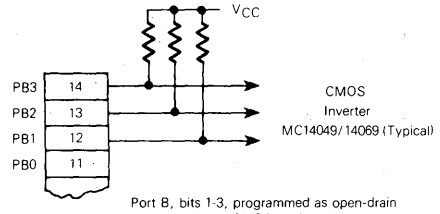
Port A, bit 7 programmed as output, driving CMOS loads and bit 4 one TTL load directly (using CMOS output option).



Port B, bit 3 programmed as output, driving Darlington-base directly.

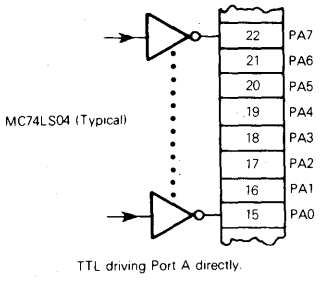


Port B, bit 0 and bit 1 programmed as output, driving LEDs directly.

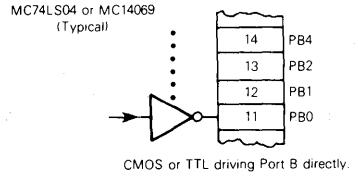


Port B, bits 1-3, programmed as open-drain outputs, driving CMOS loads using external pull-up resistors.

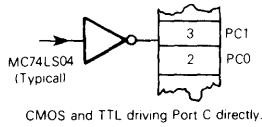
(b) Input Modes



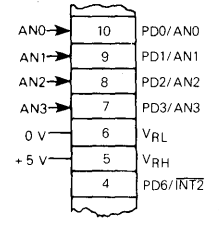
TTL driving Port A directly.



CMOS or TTL driving Port B directly.



CMOS and TTL driving Port C directly.



Port D used as 4-channel A/D input

Figure 6-2. Typical Port Connections

6.2 PORT B TOGGLE CAPABILITY

Port B1 and B0 data registers have toggle capability at the timer overflow times. Under the control of the timer output cross-couple bit in the miscellaneous register, (MR0), the overflow pulses from timer A and timer B are directed to port B1 and B0 data registers. (See Figures 6-3 and 6-4.)

An incoming toggle pulse on port B0 is allowed to toggle the data register if port B DCR bit 4 (DCR4) is cleared. At reset this bit is set. An incoming toggle pulse on port B1 is allowed to toggle the port B1 data register under the following conditions governed by control bits in SPI control register and SPI clock arbitration flip-flop status.

PB1 toggle enable = $(\overline{SPICR7} \bullet SPICR4 \bullet (\overline{PB0} + DDRB0) + SPICR2 \bullet \overline{SPICR4}) \bullet \overline{CLAQ}$

where:
SPICR7 = SPI interrupt request flag bit,
SPICR4 = SPI transaction enable bit,
SPICR2 = port B 1 toggle enable bit, and
CLAQ = clock arbitration flip-flop output.

When PB1 toggle enable is asserted, MCU write to PB1 data register is inhibited. When SPI is not used, SPICR4 and CLAQ are reset. Therefore, SPICR2 can directly control the port B1 toggle capability. (See **4.6 SERIAL PERIPHERAL INTERFACE OPERATION**.)

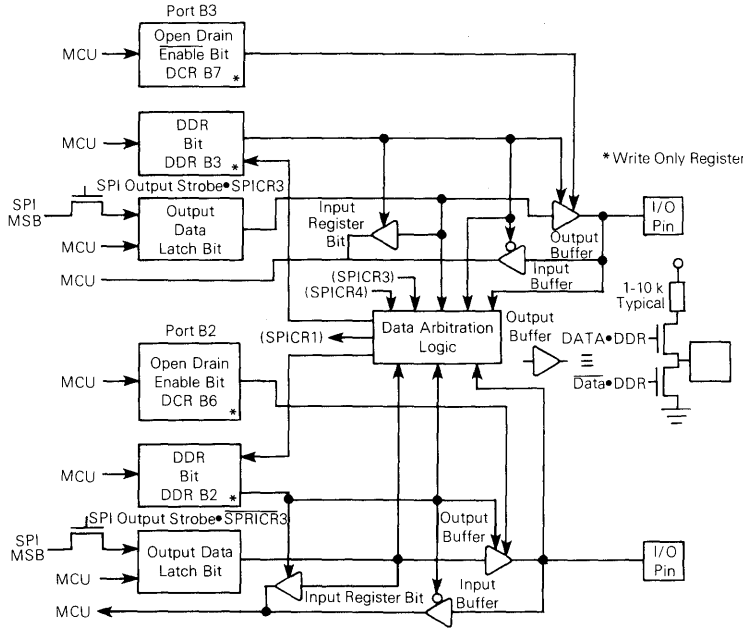
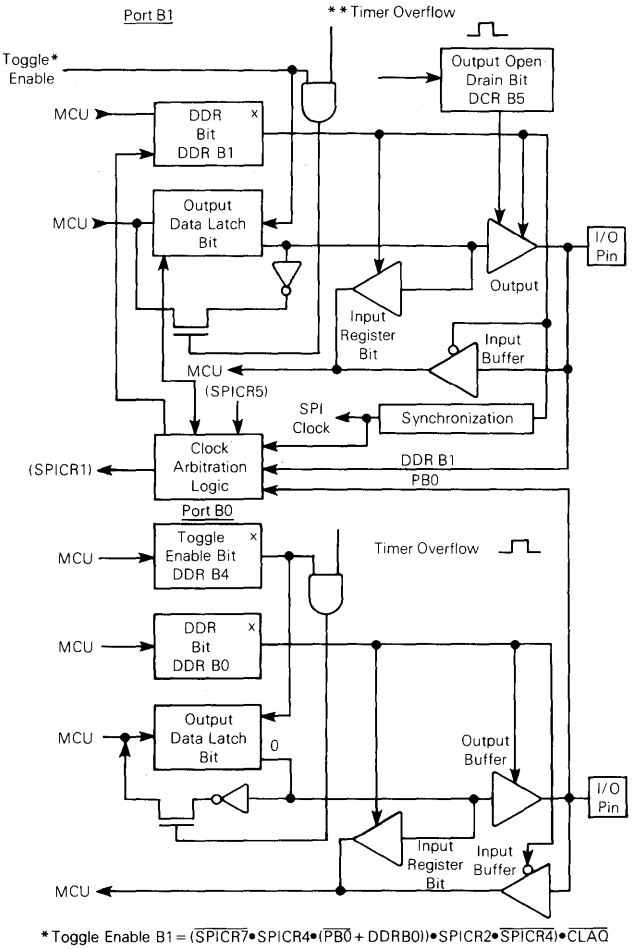
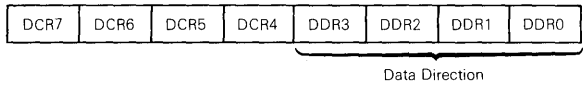


Figure 6-3. Port B Configuration (Sheet 1 of 2)



* Toggle Enable B1 = ($\overline{SPICR7}$•SPICR4•($\overline{PB0}$ + DDRB0))•SPICR2•$\overline{SPICR4}$)•$\overline{CLAQ}$
* * A, or B Depends on (MR0)
x Write Only Register

Figure 6-3. Port B Configuration (Sheet 2 of 2)



DCR7 – PB3 Output Buffer Open Drain Enable
DCR6 – PB2 Output Buffer Open Drain Enable
DCR5 – PB1 Output Buffer Open Drain Enable
DCR4 – PB0 Toggle Enable
DDR3 – PB3 Output Mode
DDR2 – PB2 Output Mode
DDR1 – PB1 Output Mode
DDR0 – PB0 Output Mode

*Write Only Register: All Bits Read as "1"

Figure 6-4. Port B Data Control Register

Port toggle capability allows action on port B0 or B1 or both as a result of timer overflows. This speeds up timer overflow to port service, compared to the normal program controlled method, and is very useful in critical real-time related applications.

Toggle capability on port B1 is fundamental for SPI operation in the clock master mode, where the clock pulses are generated by the MCU using this feature as controlled by one of the two available timers.

A write to port B0 or B1 data registers is inhibited while the individual port toggle enable is asserted. This allows a write to other port B data registers without disturbing the toggle feature of the selected port line.

6.3 PORT B DATA CONTROL REGISTER

The port B data control register consists of four status bits (DCR7 through DCR4) and four data direction bits (DCR3 through DCR0). DCR7, DCR6, and DCR5 are respectively port B3, B2, and B1 open-drain output control bits. These bits are set at reset or under program control and cleared under program control. When clear, the port output buffers operate in the open-drain mode, if the port lines are in the output mode. When set, the port output buffers operate in the push-pull mode.

DCR4 is a toggle enable control bit for port B0. This bit is set at reset or under program control and cleared under program control. When cleared, the timer overflow pulse causes the data register on port B0 to toggle.

When PB0 toggle enable is asserted by clearing DCR4, MCU write to the PB0 data register is inhibited.

DCR3, DCR2, DCR1, and DCR0 are respectively the port B3, B2, B1, and B0 data direction registers.

6.4 PORT A AND C DATA DIRECTION REGISTERS

Port A has an 8-bit and port C has a 2-bit wide data direction register. All bits are cleared at reset to the input mode. These registers are write only; they read as $FF.

6.5 MISCELLANEOUS REGISTER

The miscellaneous register (shown below), at memory location $0A, contains control and status information related to $\overline{INT2}$, auxiliary counter, prescaler 1 and 2, and timer overflow.

MR7	MR6	MR5	MR4	MR3	MR2	MR1	MR0	$0A

MR7 $\overline{INT2}$ Interrupt Request Bit
This bit is set when a negative edge is detected on $\overline{INT2}$ pin. If not masked by $\overline{INT2}$ interrupt mask bit, (MR6) it causes an interrupt request to the MCU. If the I bit in the condition code register is clear, the MCU will acknowledge interrupt by executing the interrupt procedure. The interrupt vector is fetched from address $FF8 and $FF9. This bit is cleared under program control or at reset.

MR6 $\overline{INT2}$ Interrupt Request Mask
This bit is set at reset or under program control. When set, it inhibits the $\overline{INT2}$ interrupt request from being acknowledged by the MCU. MR6 is cleared under program control.

MR5 Auxiliary Counter Status/Preset Bit
This bit is set whenever the auxiliary counter overflows. If it is not masked by the auxiliary counter reset mask MR4, it will drive a switch to V_{SS} on the $\overline{RESET}$ pin causing an MCU reset. This bit is cleared at reset or under program control. MR5 may be used as an auxiliary counter preset bit. If MR5 is clear it is possible to preset the auxiliary counter by writing a logic one to MR5. If MR5 is set (i.e., the auxiliary counter has already overflowed at least once) a logic zero written to MR5 clears the MR5 bit and presets the auxiliary counter. MR5 is cleared at reset. Refer to Figure 3-7 for auxiliary counter timing information.

MR4 Auxiliary Counter Reset Mask Bit
This bit is set at reset or under program control. When set, it inhibits activation of the reset switch controlled by MR5 on the $\overline{RESET}$ pin. MR4 is cleared under program control.

MR3 Prescaler 1 Clear Bit
This bit is used to preset the contents of prescaler 1 to $7F. This bit reads as a zero. In order to preset prescaler 1, a logic one must be written into MR3.

MR2 Prescaler 2 Clear Bit
This bit is used to preset the contents of prescaler 2 to $7FFF. This bit reads as a zero. In order to preset prescaler 2, a logic one must be written into MR2.

MR1 Prescaler Cross-Couple Bit
This bit controls the outputs of prescalers 1 and 2 and directs them to either timer A or timer B clock inputs. This bit is cleared at reset or under program control and set under program control. When MR1 is clear the output of prescaler 1 is used as a clock input of timer A and the output of prescaler 2 is used as clock input for timer B. When MR1 is set, outputs of the prescalers are cross-coupled. Thus, prescaler 1 feeds the timer B clock input and prescaler 2 feeds the timer A clock input.

To avoid truncation errors at the time of cross coupling, both prescalers may be preset by writing a one to MR3 and MR2 simultaneously.

MR0 Port B Toggle Cross-Couple Bit
This bit controls the overflow pulses of timers A and B and directs them to either port B1 or B0. This bit is cleared at reset or under program control and set under program control. When MR0 is clear, the overflow output pulse of timer A is used as a port B1 data register toggle clock source. Similarly, the timer B overflow output pulse is directed to port B0 toggle clock input.

When MR0 is set, timer A overflow output is directed to port B0 and timer B output is directed toward port B1.

6.6 ANALOG-TO-DIGITAL CONVERTER (A/D)

The MC6805S2 microcomputers have an 8-bit analog-to-digital (A/D) converter implemented on the chip using a successive approximation technique. Up to four external analog inputs, via port D, are connected to the A/D through a multiplexer. Four internal analog channels may be selected for calibration purposes (V_{RH}-V_{RL}, V_{RH}-$V_{RL}/2$, V_{RH}-$V_{RL}/4$, and V_{RL}). The accuracy of these internal channels will not necessarily meet the accuracy specifications of the external channels.

A fifth external analog input (AN4) is available via mask option. When selected, it replaces the V_{RH} internal channel. Due to signal routing, the accuracy of this fifth channel may be slightly less than AN0-AN3. The fifth A/D channel could be used to conveniently monitor the standby RAM supply voltage, as an example.

The multiplexer selection is controlled by the A/D control register (ACR) bits 0, 1, and 2.

Whenever the ACR is written, the conversion in progress is aborted, the conversion complete flag (ACR bit 7) is cleared, and the selected input is sampled for five machine cycles and held internally. During these five cycles the analog input will appear approximately like a 25 picofarads (maximum) capacitor charging through a 2.6 kilohm resistor, typical (see Figure 6-5).

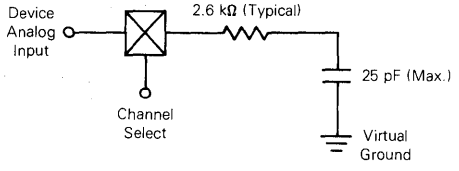


Figure 6-5. Effective Analog Input Impedance (During Sampling Only)

The converter operates continuously using 30 machine cycles to complete a conversion of the sampled analog input. When the conversion is complete, the digitized sample of digital value is placed in the A/D result register (ARR), the conversion complete flag is set, the selected input is sampled again, and a new conversion is started.

The A/D is ratiometric. Two reference voltages (V_{RH} and V_{RL}) are supplied to the converter via port D pins. An input voltage equal to V_{RH} converts to $FF (full scale) and an input voltage equal to V_{RL} converts to $00. An input voltage greater than V_{RH} converts to $FF and no overflow indication is provided. Similarly, an input voltage less than V_{RL}, but greater than V_{SS} converts to $00. Maximum and minimum ratings must not be exceeded. For ratiometric conversion, the source of each analog input should use V_{RH} as the supply voltage and be referenced to V_{RL}. To maintain the full accuracy on the A/D, V_{RH} should be equal to or less than V_{DD}, V_{RL} should be equal to or greater than V_{SS} but less than the maximum specification and (V_{RH}-V_{RL}) should be equal to or greater than 4 volts.

The A/D has a built-in ½ LSB offset intended to reduce the magnitude of the quantizing error to ± ½ LSB, rather than +0, −1 LSB with no offset. This implies that, ignoring errors, the transition point from $00 to $01 occurs at ½ LSB above V_{RL}. Similarly, the transition from $FE to $FF occurs 1.5 LSB below V_{RH}, ideally. Refer to Figures 6-6 and 6-7.

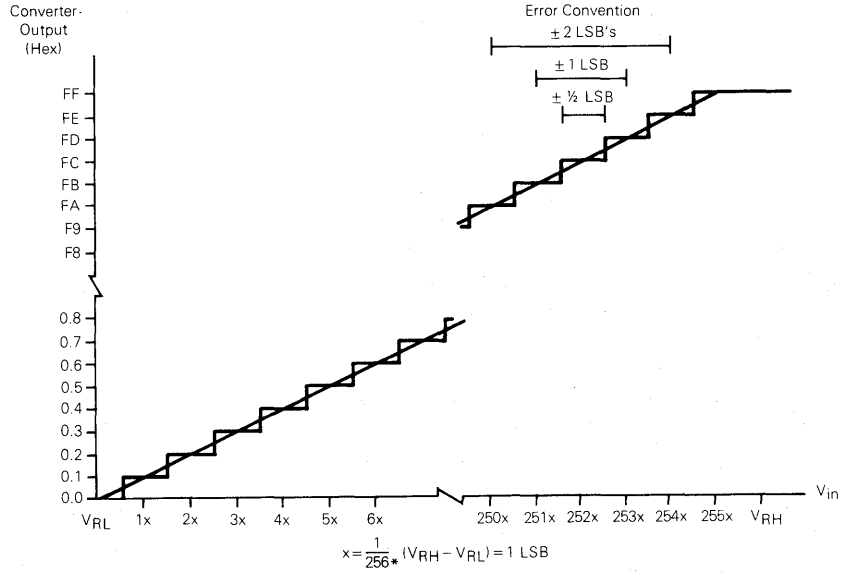


Figure 6-6. Ideal Converter Transfer Characteristic

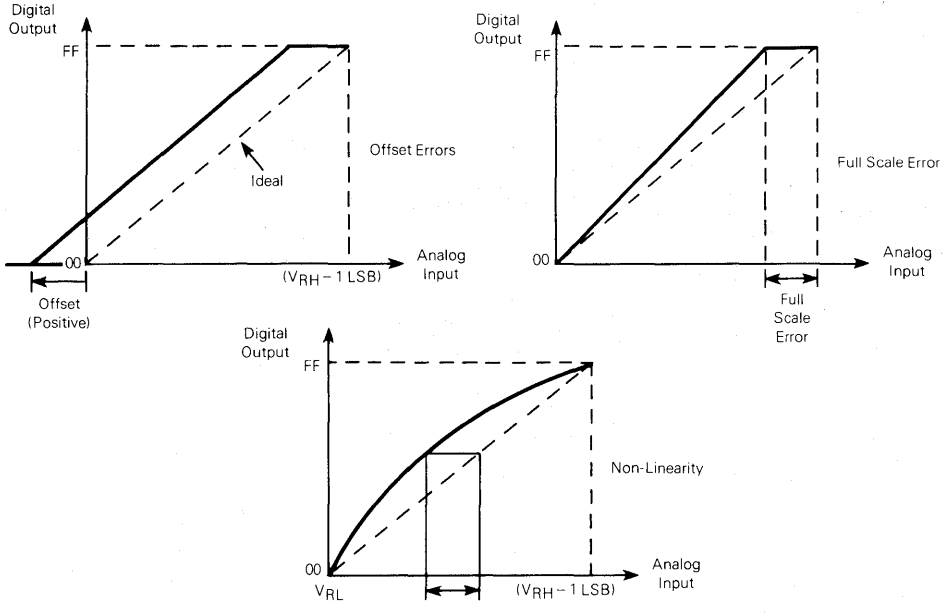


Figure 6-7. Types of Conversion Errors

On release of reset the A/D control register (ACR) is cleared therefore after reset channel zero will be selected and the conversion complete flag will be clear. Refer to Figure 6-8 and Table 6-2.

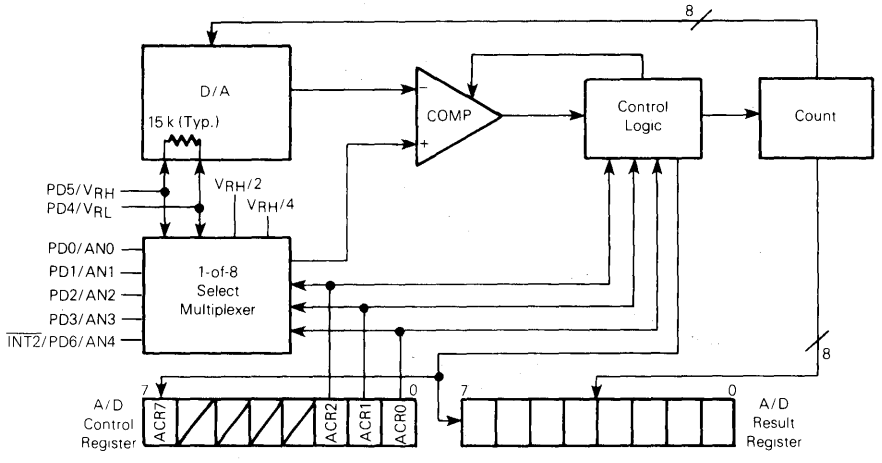


Figure 6-8. A/D Block Diagram

Table 6-2. A/D Input Mux Selection

A/D Control Register				A/D Output (Hex)		
ACR2	ACR1	ACR0	Input Selected	Min	Typ	Max
0	0	0	AN0			
0	0	1	AN1			
0	1	0	AN2			
0	1	1	AN3			
1	0	0	V_{RH}**	FE**	FF**	FF**
1	0	1	V_{RL}*	00	00	01
1	1	0	$V_{RH/4}$*	3F	40	41
1	1	1	$V_{RH/2}$*	7F	80	81

*Internal (calibration) levels

**AN4 may replace the V_{RH} calibration channel if selected via mask option.

SECTION 7
SOFTWARE AND INSTRUCTION SET

This section describes the software and instruction set for the MC6805S2.

7.1 SOFTWARE

The following paragraphs describe the software available to the user.

7.1.1 Bit Manipulation

The MC6805S2 MCU has the ability to set or clear any single RAM or input/output bit (except the data direction registers; see Caution under INPUT/OUTPUT) with a single instruction (BRSET, BCLR). Any bit in page zero, including ROM except the DDRs, can be tested using the BRSET and BRCLR instructions and the program branches as a result of its state. The carry bit (C) equals the value of the bit referenced by BRSET or BRCLR. The capability of working with any bit in RAM, ROM, or I/O allows the user to have individual flags in RAM or to handle single I/O bits as control lines.

The coding example in Figure 7-1 illustrates the usefulness of the bit manipulation and test instructions. Assume that the MCU is to communicate with an external serial device. The external device has a data ready signal, a data output line, and a clock line to clock data one bit at a time, LSB first out of the device. The MCU waits until the data is ready, clocks the external device, picks up the data in the carry flag (C bit), clears the clock line, and finally accumulates the data bit in a RAM location.

```
*       BRSET   2,PORTA,*       WAIT FOR READY
        BSET    1, PORTA        CLOCK NEXT BIT IN
        BRCLR   0, PORTA, NEXT  PICKUP BIT IN C-BIT
NEXT    BCLR    1, PORTA        RETURN CLOCK LINE HIGH
        ROR     RAMLOC          MOVE C-BIT INTO RAM
```

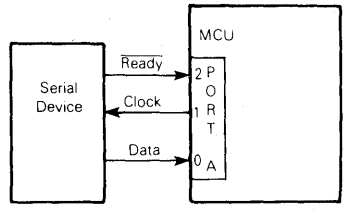


Figure 7-1. Bit Manipulation Example

7.1.2 Addressing Modes

The MC6805S2 MCU has ten addressing modes available for use by the programmer. They are explained briefly in the following paragraphs. For additional details and graphical illustrations, refer to the *M6805 HMOS/M146805 CMOS Family Users Manual.*

The term "effective address" (EA) is used in describing the addressing modes. EA is defined as the address from which the argument for an instruction is fetched or stored.

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

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

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

7.1.2.4 RELATIVE. The relative addressing mode is only used in branch instructions. In relative addressing, the contents of the 8-bit signed byte following the opcode (the offset) is added to the PC if, and only if, the branch condition is true. Otherwise, control proceeds to the next instruction. The span of relative addressing is from − 126 to + 129 from the opcode address. The programmer need not worry about calculating the correct offset if he uses the Motorola assembler, since it calculates the proper offset and checks to see if it is within the span of the branch.

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

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

7.1.2.7 INDEXED, 16-BIT OFFSET. In the indexed, 16-bit offset addressing mode, the effective address is the sum of the contents of the unsigned 8-bit index register and the two unsigned bytes following the opcode. This addressing mode can be used in a manner similar to indexed, 8-bit offset except that this three-byte instruction allows tables to be anywhere in memory. As with direct and extended, the Motorola assembler determines the shortest from of indexed addressing.

7.1.2.8 BIT SET/CLEAR. In the bit set/clear addressing mode, the bit to be set or cleared is part of the opcode, and the byte following the opcode specifies the direct address of the byte in which the specified bit is to be set or cleared. Thus, any read/write bit in the first 256 locations of memory, including I/O, can be selectively set or cleared with a single two-byte instruction. See Caution under **6.1 INPUT/OUTPUT.**

7.1.2.9 BIT TEST AND BRANCH. The bit test and branch addressing mode is a combination of direct addressing and relative addressing. The bit which is to be tested and condition (set or clear) is included in the opcode, and the address of the byte to be tested is in the single byte immediately following the opcode byte. The signed relative 8-bit offset in the third byte is added to the PC if the specified bit is set or cleared in the specified memory location. This single three-byte instruction allows the program to branch based on the condition of any readable bit in the first 256 locations of memory. The span of branching is from − 125 to + 130 from the opcode address. The state of the tested bit is also transferred to the carry bit of the condition code registers. See Caution under **6.1 INPUT/OUTPUT.**

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

7.2 INSTRUCTION SET

The MC6805S2 MCU has a set of 59 basic instructions, which when combined with the 10 addressing modes produce 207 usable opcodes. They can be divided into five different types: register/memory, read-modify-write, branch, bit manipulation, and control. The following paragraphs briefly explain each type. All the instruction within a given type are presented in individual tables.

7.2.1 Register/Memory Instructions

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump-to-subroutine (JSR) instructions have no register operand. Refer to Table 7-1.

Table 7-1. Register/Memory Instructions

		Addressing Modes																	
		Immediate			Direct			Extended			Indexed (No Offset)			Indexed (8-Bit Offset)			Indexed (16-Bit Offset)		
Function	Mnemonic	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	OP Code	# Bytes	# Cycles
Load A from Memory	LDA	A6	2	2	B6	2	4	C6	3	5	F6	1	4	E6	2	5	D6	3	6
Load X from Memory	LDX	AE	2	2	BE	2	4	CE	3	5	FE	1	4	EE	2	5	DF	3	6
Store A in Memory	STA	–			B7	2	5	C7	3	6	F7	1	5	E7	2	6	D7	3	7
Store X in Memory	STX	–			BF	2	5	CF	3	6	FF	1	5	EF	2	6	DF	3	7
Add Memory to A	ADD	AB	2	2	BB	2	4	CB	3	5	FB	1	4	EB	2	5	DB	3	6
Add Memory and Carry to A	ADC	A9	2	2	B9	2	4	C9	3	5	F9	1	4	E9	2	5	D9	3	6
Subtract Memory	SUB	A0	2	2	B0	2	4	C0	3	5	F0	1	4	E0	2	5	D0	3	6
Subtract Memory from A with Borrow	SBC	A2	2	2	B2	2	4	C2	3	5	F2	1	4	E2	2	5	D2	3	6
AND Memory to A	AND	A4	2	2	B4	2	4	C4	3	5	F4	1	4	E4	2	5	D4	3	6
OR Memory with A	ORA	AA	2	2	BA	2	4	CA	3	5	FA	1	4	EA	2	5	DA	3	6
Exclusive OR Memory with A	EOR	A8	2	2	B8	2	4	C8	3	5	F8	1	4	E8	2	5	D8	3	6
Arithmetic Compare A with Memory	CMP	A1	2	2	B1	2	4	C1	3	5	F1	1	4	E1	2	5	D1	3	6
Arithmetic Compare X with Memory	CPX	A3	2	2	B3	2	4	C3	3	5	F3	1	4	E3	2	5	D3	3	6
Bit Test Memory with A (Logical Compare)	BIT	A5	2	2	B5	2	4	C5	3	5	F5	1	4	E5	2	5	D5	3	6
Jump Unconditional	JMP	–			BC	2	3	CC	3	4	FC	1	3	EC	2	4	DC	3	5
Jump to Subroutine	JSR	–			BD	2	7	CD	3	8	FD	1	7	ED	2	8	DD	3	9

7.2.2 Read-Modify-Write Instructions

These instructions read a memory location or a register, modify or test it contents, and write the modified value back to memory or to the register; see Caution under **6.1 INPUT/OUTPUT.** The test for negative or zero (TST) instruction is included in the read-modify-write instruction though it does not perform the write. Refer to Table 7-2.

Table 7-2. Read-Modify-Write Instructions

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

7.2.3 Branch Instructions

The branch instructions cause a branch from the program when a certain condition is met. Refer to Table 7-3.

Table 7-3. Branch Instructions

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

7.2.4 Bit Manipulation Instructions

The instructions are used on any bit in the first 256 bytes of memory; see Caution under **6.1 INPUT/OUTPUT.** One group either sets or clears. The other group performs the bit test and branch operations. Refer to Table 7-4.

Table 7-4. Bit Manipulation Instructions

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

7.2.5 Control Instructions

The control instructions control the MCU operations during program execution. Refer to Table 7-5.

Table 7-5. Control Instructions

		Inherent		
Function	Mnemonic	Op Code	# Bytes	# Cycles
Transfer A to X	TAX	97	1	2
Transfer X to A	TXA	9F	1	2
Set Carry Bit	SEC	99	1	2
Clear Carry Bit	CLC	98	1	2
Set Interrupt Mask Bit	SEI	9B	1	2
Clear Interrupt Mask Bit	CLI	9A	1	2
Software Interrupt	SWI	83	1	11
Return from Subroutine	RTS	81	1	6
Return from Interrupt	RTI	80	1	9
Reset Stack Pointer	RSP	9C	1	2
No Operation	NOP	9D	1	2

7.2.6 Alphabetical Listing

The complete instruction set is given in a alphabetical order in Table 7-6.

Table 7-6. Instruction Set (Sheet 1 of 2)

	Addressing Modes										Condition Code				
Mnemonic	Inherent	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/ Clear	Bit Test & Branch	H	I	N	Z	C
ADC		X	X	X		X	X	X			^	•	^	^	^
ADD		X	X	X		X	X	X			^	•	^	^	^
AND		X	X	X		X	X	X			•	•	^	^	•
ASL	X		X			X	X				•	•	^	^	^
ASR	X		X			X	X				•	•	^	^	^
BCC					X						•	•	•	•	•
BCLR									X		•	•	•	•	•
BCS					X						•	•	•	•	•
BEQ					X						•	•	•	•	•
BHCC					X						•	•	•	•	•
BHCS					X						•	•	•	•	•
BHI					X						•	•	•	•	•
BHS					X						•	•	•	•	•
BIH					X						•	•	•	•	•
BIL					X						•	•	•	•	•
BIT		X	X	X		X	X	X			•	•	^	^	•
BLO					X						•	•	•	•	•
BLS					X						•	•	•	•	•
BMC					X						•	•	•	•	•
BMI					X						•	•	•	•	•
BMS					X						•	•	•	•	•
BNE					X						•	•	•	•	•
BPL					X						•	•	•	•	•
BRA					X						•	•	•	•	•
BRN					X						•	•	•	•	•
BRCLR										X	•	•	•	•	^
BRSET										X	•	•	•	•	^
BSET									X		•	•	•	•	•
BSR					X						•	•	•	•	•
CLL	X										•	•	•	•	0
CLI	X										•	0	•	•	•
CLR	X		X			X	X				•	•	0	1	•
CMP		X	X	X		X	X	X			•	•	^	^	^
COM	X		X			X	X				•	•	^	^	1
CPX		X	X	X		X	X	X			•	•	^	^	^
DEC	X		X			X	X				•	•	^	^	•
EOR		X	X	X		X	X	X			•	•	^	^	•
INC	X		X			X	X				•	•	^	^	•
JMP			X	X		X	X	X			•	•	•	•	•
JSR			X	X		X	X	X			•	•	•	•	•

Condition Code Symbols:

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

Table 7-6. Instruction Set (Sheet 2 of 2)

	Addressing Modes										Condition Code				
Mnemonic	Inherent	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/ Clear	Bit Test & Branch	H	I	N	Z	C
LDA		X	X	X		X	X	X			•	•	^	^	•
LDX		X	X	X		X	X	X			•	•	^	^	•
LSL	X		X			X	X				•	•	^	^	^
LSR	X		X			X	X				•	•	0	^	^
NEG	X		X			X	X				•	•	^	^	^
NOP	X										•	•	•	•	•
ORA		X	X	X		X	X	X			•	•	^	^	•
ROL	X		X			X	X				•	•	^	^	^
RSP	X										•	•	•	•	•
RTI	X										?	?	?	?	?
RTS	X										•	•	•	•	•
SBC		X	X	X		X	X	X			•	•	^	^	^
SEC	X										•	•	•	•	1
SEI	X										•	1	•	•	•
STA			X	X		X	X	X			•	•	^	^	•
STX			X	X		X	X	X			•	•	^	^	•
SUB		X	X	X		X	X	X			•	•	^	^	^
SWI	X										•	1	•	•	•
TAX	X										•	•	•	•	•
TST	X		X			X	X				•	•	^	^	•
TXA	X										•	•	•	•	•

Condition Code Symbols:

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

7.2.7 Opcode Map

Table 7-7 is an opcode map for the instruction used on the MCU.

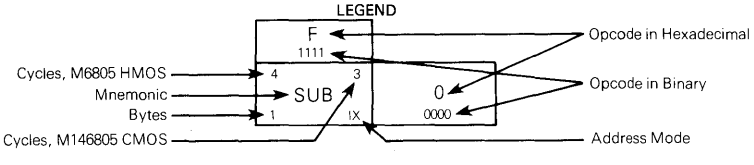
Table 7-7. M6805 HMOS/M146805 CMOS Family Instruction Set Opcode Map

	Bit Manipulation		Branch	Read/Modify/Write				
	BTB	BSC	REL	DIR	A	X	IX1	IX
Hi / Low	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111
0 0000	10 5 BRSET0 3 BTB	7 5 BSET0 2 BSC	4 3 BRA 2 REL	6 5 NEG 2 DIR	4 3 NEG 1 A	4 3 NEG 1 X	7 6 NEG 2 IX1	6 5 NEG 1 IX
1 0001	10 5 BRCLR0 3 BTB	7 5 BCLR0 2 BSC	4 3 BRN 2 REL					
2 0010	10 5 BRSET1 3 BTB	7 5 BSET1 2 BSC	4 3 BHI 2 REL					
3 0011	10 5 BRCLR1 3 BTB	7 5 BCLR1 2 BSC	4 3 BLS 2 REL	6 5 COM 2 DIR	4 3 COM 1 A	4 3 COM 1 X	7 6 COM 2 IX1	6 5 COM 1 IX
4 0100	10 5 BRSET2 3 BTB	7 5 BSET2 2 BSC	4 3 BCC 2 REL	6 5 LSR 2 DIR	4 3 LSR 1 A	4 3 LSR 1 X	7 6 LSR 2 IX1	6 5 LSR 1 IX
5 0101	10 5 BRCLR2 3 BTB	7 5 BCLR2 2 BSC	4 3 BCS 2 REL					
6 0110	10 5 BRSET3 3 BTB	7 5 BSET3 2 BSC	4 3 BNE 2 REL	6 5 ROR 2 DIR	4 3 ROR 1 A	4 3 ROR 1 X	7 6 ROR 2 IX1	6 5 ROR 1 IX
7 0111	10 5 BRCLR3 3 BTB	7 5 BCLR3 2 BSC	4 3 BEQ 2 REL	6 5 ASR 2 DIR	4 3 ASR 1 A	4 3 ASR 1 X	7 6 ASR 2 IX1	6 5 ASR 1 IX
8 1000	10 5 BRSET4 3 BTB	7 5 BSET4 2 BSC	4 3 BHCC 2 REL	6 5 LSL 2 DIR	4 3 LSL 1 A	4 3 LSL 1 X	7 6 LSL 2 IX1	6 5 LSL 1 IX
9 1001	10 5 BRCLR4 3 BTB	7 5 BCLR4 2 BSC	4 3 BHCS 2 REL	6 5 ROL 2 DIR	4 3 ROL 1 A	4 3 ROL 1 X	7 6 ROL 2 IX1	6 5 ROL 1 IX
A 1010	10 5 BRSET5 3 BTB	7 5 BSET5 2 BSC	4 3 BPL 2 REL	6 5 DEC 2 DIR	4 3 DEC 1 A	4 3 DEC 1 X	7 6 DEC 2 IX1	6 5 DEC 1 IX
B 1011	10 5 BRCLR5 3 BTB	7 5 BCLR5 2 BSC	4 3 BMI 2 REL					
C 1100	10 5 BRSET6 3 BTB	7 5 BSET6 2 BSC	4 3 BMC 2 REL	6 5 INC 2 DIR	4 3 INC 1 A	4 3 INC 1 X	7 6 INC 2 IX1	6 5 INC 1 IX
D 1101	10 5 BRCLR6 3 BTB	7 5 BCLR6 2 BSC	4 3 BMS 2 REL	6 4 TST 2 DIR	4 3 TST 1 A	4 3 TST 1 X	7 5 TST 2 IX1	6 4 TST 1 IX
E 1110	10 7 BRSET7 3 BTB	7 5 BSET7 2 BSC	4 3 BIL 2 REL					
F 1111	10 7 BRCLR7 3 BTB	7 5 BCLR7 2 BSC	4 3 BIH 2 REL	6 5 CLR 2 DIR	4 3 CLR 1 A	4 3 CLR 1 X	7 6 CLR 2 IX1	6 5 CLR 1 IX

Abbreviations for Address Modes

INH	Inherent	EXT	Extended	IX	Indexed (No Offset)
A	Accumulator	REL	Relative	IX1	Indexed, 1 Byte (8-Bit) Offset
X	Index Register	BSC	Bit Set/Clear	IX2	Indexed, 2 Byte (16-Bit) Offset
IMM	Immediate	BTB	Bit Test and Branch	*	M146805 CMOS Family Only
DIR	Direct				

	Control		Register/Memory						
	INH	INH	IMM	DIR	EXT	IX2	IX1	IX	
	8 1000	9 1001	A 1010	B 1011	C 1100	D 1101	E 1110	F 1111	Hi / Low
	9 9 RTI 1 INH		2 2 SUB 2 IMM	4 3 SUB 2 DIR	5 4 SUB 3 EXT	6 5 SUB 3 IX2	5 4 SUB 2 IX1	4 3 SUB 1 IX	0 0000
	6 6 RTS 1 INH		2 2 CMP 2 IMM	4 3 CMP 2 DIR	5 4 CMP 3 EXT	6 5 CMP 3 IX2	5 4 CMP 2 IX1	4 3 CMP 1 IX	1 0001
			2 2 SBC 2 IMM	4 3 SBC 2 DIR	5 4 SBC 3 EXT	6 5 SBC 3 IX2	5 4 SBC 2 IX1	4 3 SBC 1 IX	2 0010
	11 10 SWI 1 INH		2 2 CPX 2 IMM	4 3 CPX 2 DIR	5 4 CPX 3 EXT	6 5 CPX 3 IX2	5 4 CPX 2 IX1	4 3 CPX 1 IX	3 0011
			2 2 AND 2 IMM	4 3 AND 2 DIR	5 4 AND 3 EXT	6 5 AND 3 IX2	5 4 AND 2 IX1	4 3 AND 1 IX	4 0100
			2 2 BIT 2 IMM	4 3 BIT 2 DIR	5 4 BIT 3 EXT	6 5 BIT 3 IX2	5 4 BIT 2 IX1	4 3 BIT 1 IX	5 0101
			2 2 LDA 2 IMM	4 3 LDA 2 Dir	5 4 LDA 3 EXT	6 5 LDA 3 IX2	5 4 LDA 2 IX1	4 3 LDA 1 IX	6 0110
		2 2 TAX 1 INH		5 4 STA 2 DIR	6 5 STA 3 EXT	7 6 STA 3 IX2	6 5 STA 2 IX1	5 4 STA 1 IX	7 0111
		2 2 CLC 1 INH	2 2 EOR 2 IMM	4 3 EOR 2 DIR	5 4 EOR 3 EXT	6 5 EOR 3 IX2	5 4 EOR 2 IX1	4 3 EOR 1 IX	8 1000
		2 2 SEC 1 INH	2 2 ADC 2 IMM	4 3 ADC 2 DIR	5 4 ADC 3 EXT	6 5 ADC 3 IX2	5 4 ADC 2 IX1	4 3 ADC 1 IX	9 1001
		2 2 CLI 1 INH	2 2 ORA 2 IMM	4 3 ORA 2 DIR	5 4 ORA 3 EXT	6 5 ORA 3 IX2	5 4 ORA 2 IX1	4 3 ORA 1 IX	A 1010
		2 2 SEI 1 INH	2 2 ADD 2 IMM	4 3 ADD 2 DIR	5 4 ADD 3 EXT	6 5 ADD 3 IX2	5 4 ADD 2 IX1	4 3 ADD 1 IX	B 1011
		2 2 RSP 1 INH		3 2 JMP 2 DIR	4 3 JMP 3 EXT	5 4 JMP 3 IX2	4 3 JMP 2 IX1	3 2 JMP 1 IX	C 1100
		2 2 NOP 1 INH	8 6 BSR 2 REL	7 5 JSR 2 DIR	8 6 JSR 3 EXT	9 7 JSR 3 IX2	8 6 JSR 2 IX1	7 5 JSR 1 IX	D 1101
	* 2 STOP 1 INH		2 2 LDX 2 IMM	4 3 LDX 2 DIR	5 4 LDX 3 EXT	6 5 LDX 3 IX2	5 4 LDX 2 IX1	4 3 LDX 1 IX	E 1110
	* 2 WAIT 1 INH	2 2 TXA 1 INH		5 4 STX 2 DIR	6 5 STX 3 EXT	7 6 STX 3 IX2	6 5 STX 2 IX1	5 4 STX 1 IX	F 1111



7-11/7-12

SECTION 8
ELECTRICAL SPECIFICATIONS

This section contains the electrical specifications for the MC6805S2.

8.1 MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage PC0 in Self-Check Mode All Other	V_{in}	−0.3 to +15.0 −0.3 to +7.0	V
Port A and C Source Current per Pin (One at a Time)	I_{out}	10	mA
Operating Temperature Range MC6805S2 MC6805S2C	T_A	0 to 70 −40 to 85	°C
Storage Temperature Range	T_{stg}	−55 to +150	
Junction Temperature Plastic Package Ceramic Package Cerdip	T_J	150 175 175	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electrical fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high-impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out} \leq V_{CC}$. Reliability of operation is enhanced if unused inputs except EXTAL are tied to an appropriate voltage level (e.g., either V_{SS} or V_{CC}).

8.2 THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic Ceramic Cerdip	θ_{JA}	 70 60 60	°C/W

8.3 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:

$T_A \equiv$ Ambient Temperature, °C

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

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

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

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

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

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

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

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.

8.4 ELECTRICAL CHARACTERISTICS (V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage					
$\overline{RESET}$ (4.75 ≤ V_{CC} ≤ 5.75)	V_{IH}	4.0	–	V_{CC} + 0.7	V
(V_{CC} ≤ 4.75)		V_{CC} – 0.5	–	V_{CC} + 0.7	
$\overline{INT}$ (4.75 ≤ V_{CC} ≤ 5.75)		4.0	*	V_{CC} + 0.7	
(V_{CC} ≤ 4.75)		V_{CC} – 0.5	*	V_{CC} + 0.7	
All Other		2.0	–	V_{CC} + 0.7	
Input High Voltage PC0					
Port/Timer Mode	V_{IH}	2.0	–	V_{CC} + 1.0	V
Self-Check Mode		9.0	10.0	15.0	
Input Low Voltage					
$\overline{RESET}$	V_{IL}	V_{SS}	–	0.8	V
$\overline{INT}$		V_{SS}	*	1.5	
All Other (Except A/D Inputs)		V_{SS}	–	0.8	
$\overline{RESET}$ Hysteresis Voltages (See RESETS)					
"Out of Reset"	V_{IRES+}	2.1	–	4.0	V
"Into Reset"	V_{IRES-}	0.8	–	2.0	
Standby Supply Voltage ($\overline{INT2}$ Input Option)	V_{STBY}	3.0	–	5.75	V
Standby Current ($\overline{INT2}$ Input Option) (V_{STBY} = 3.0 V)	I_{STBY}	–	1.0	TBD	mA
Power Dissipation – No Port Loading					
(V_{CC} = 5.75 V, T_A = 0°C)	P_D	–	600	TBD	mW
(V_{CC} = 5.75 V, T_A = –40°C)	P_D	–	670	TBD	
Input Capacitance (Except Analog Inputs – See Note)	C_{in}	–	10	–	pF
Low Voltage Recover	V_{LVR}	–	–	4.75	V
Low Voltage Inhibit	V_{LVI}	2.75	3.75	4.70	V
Input Current					
$\overline{INT}$					
(V_{in} = 2.4 V to V_{CC})		–	20	50	
EXTAL					
(V_{in} = 2.4 V to V_{CC} Crystal Option)	I_{in}	–	–	10	μA
(V_{in} = 0.4 V Crystal Option)		–	–	–1600	
$\overline{RESET}$					
(V_{in} = 0.8 V)		–4.0	–	–50	
(External Charging Current)					

TBD = To Be Determined

NOTE: Port D analog inputs, when selected, C_{in} = 25 pF for the first 5 out of 30 cycles.

*This input (when unused) floats to approximately 2.0 V due to internal biasing.

8.5 SWITCHING CHARACTERISTICS (V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Oscillator Frequency	f_{osc}	0.4	–	4.2	MHz
Cycle Time ($4/f_{osc}$)	t_{cyc}	0.95	–	10	μs
$\overline{INT}$, $\overline{INT2}$, and TIMER Pulse Width (See 5.4 INTERRUPTS)	t_{WL}, t_{WH}	t_{cyc} + 250	–	–	ns
$\overline{RESET}$ Pulse Width	t_{RWL}	t_{cyc} + 250	–	–	ns
$\overline{RESET}$ Delay Time (External Capacitance = 1 μF)	t_{RHL}	–	100	–	ms
$\overline{INT}$ Zero-Crossing Detection Input Frequency (for ±5° Accuracy)	f_{INT}	0.03	–	1	kHz
External Clock Input Duty Cycle (EXTAL)	–	40	50	60	%
Oscillator Startup Time Crystal*	t_{su}	–	–	100	ms
SPICL High Time	t_{SPICLH}	4	–	–	t_{cyc}
SPICL Low Time	t_{SPICHL}	4	–	–	t_{cyc}
SPICL Rise and Fall Time	t_{Sr}, t_{Sf}	–	–	1	μs
SPID Input Data Setup Time	t_{SDS}	2	–	–	t_{cyc}
SPID Input Data Hold Time	t_{SDH}	2	–	–	t_{cyc}
SPICL to SPISS Lag Time	t_{SStG}	4	–	–	t_{cyc}
SPISS to SPICL Lead Time	t_{SSLD}	4	–	–	t_{cyc}
SPISS High and Low Time	t_{SSH}, t_{SSL}	4	–	–	t_{cyc}
Start Bit to First Clock Lead Time	t_{STL}	1	–	–	t_{cyc}
External Timer Input to Timer Change Time	t_{PCT}	3	–	–	t_{cyc}
Timer Change to Port B Toggle Time	t_{TPB}	2	–	–	t_{cyc}
$\overline{INT2}$ to Timer A Load Time	t_{INTL}	3	–	–	t_{cyc}

*See Figure 5-7 for typical crystal parameters.

8.6 A/D CONVERTER CHARACTERISTICS (V_{CC} = +5.25 Vdc ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Min	Typ	Max	Unit	Comments
Resolution	8	8	8	Bits	
Non-Linearity*	–	–	± ½	LSB	After removing zero-offset and full-scale errors
Quantizing Error	–	–	± ½	LSB	
Full Scale Error*	–	–	TBD	LSB	Deviation of step $FE to $FF from ideal
Zero Offset*	–	–	TBD	LSB	Deviation of step $00 to $01 from ideal
Absolute (Total) Error*	–	–	TBD	LSB	Includes errors due to all sources, including quantizing
Conversion Range V_{RH} V_{RL}	– V_{SS}	– –	V_{CC} 0.2	V	A/D accuracy may decrease proportionately as V_{RH}-V_{RL} is reduced below 4.0 V. The sum of V_{RH} and V_{RL} must not exceed V_{CC}.
Conversion Time	30	30	30	t_{cyc}	Includes sampling time
Monotonicity	(Inherent with in total error)				
Sample Time	5	5	5	t_{cyc}	
Sample/Hold Capacitance, Input	–	–	25	pF	
Analog Input Voltage	V_{RL}	–	V_{RH}	V	Transients on any analog lines (pins 19-24) are not allowed at any time during sampling or accuracy may be degraded.

*For V_{RH} = 4.0 V to 5.0 V and V_{RL} = 0 V.

8.7 PORT ELECTRICAL CHARACTERISTICS

(V_{CC} = +5.25 Vdc, ±0.5 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Port A with CMOS Drive Enabled					
Output Low Voltage I_{Load} = 1.6 mA	V_{OL}	–	–	0.4	V
Output High Voltage I_{Load} = – 100 μA	V_{OH}	2.4	–	–	V
Output High Voltage I_{Load} = – 10 μA	V_{OH}	V_{CC} – 1.0	–	–	V
Input High Voltage I_{Load} = – 300 μA (Maximum)	V_{IH}	2.0	–	V_{CC} + 0.7	V
Input Low Voltage I_{Load} = – 500 μA (Maximum)	V_{IL}	V_{SS}	–	0.8	V
Hi-Z State Input Current (V_{in} = 2.0 V to V_{CC})	I_{IH}	–	–	– 300	μA
Hi-Z State Input Current (V_{in} = 0.4 V)	I_{IL}	–	–	– 500	μA
Port B					
Output Low Voltage I_{Load} = 3.2 mA	V_{OL}	–	–	0.4	V
Output Low Voltage I_{Load} = 10 mA (Sink)	V_{OL}	–	–	1.0	V
Output High Voltage* I_{Load} = – 200 μA	V_{OH}	2.4	–	–	V
Darlington Current Drive (Source)* V_O = 1.5 V	I_{OH}	– 1.0	–	– 10	mA
Input High Voltage	V_{IH}	2.0	–	V_{CC} + 0.7	V
Input Low Voltage	V_{IL}	V_{SS}	–	0.8	V
Hi-Z State Input Current	I_{TSI}	–	< 2	10	μA
Port C and Port A with CMOS Device Disabled					
Output Low Voltage I_{Load} = 1.6 mA	V_{OL}	–	–	0.4	V
Output High Voltage I_{Load} = – 100 μA	V_{OH}	2.4	–	–	V
Input High Voltage	V_{IH}	2.0	–	V_{CC} + 0.7	V
Input Low Voltage	V_{IL}	V_{SS}	–	0.8	V
Hi-Z State Input Current	I_{TSI}	–	< 2	10	μA
Port D (Digital Inputs Only)					
Input High Voltage	V_{IH}	2.0	–	V_{CC} + 0.7	V
Input Low Voltage	V_{IL}	V_{SS}	–	0.8	V
Input Current**	I_{in}	–	< 1	5	μA

*Not applicable if programmed to open-drain state.

**PD4/V_{RL}-PD5/V_{RH}:
The A/D conversion resistor (15 kilohm typical) is connected internally between these two lines, impacting their use as digital inputs in some applications.

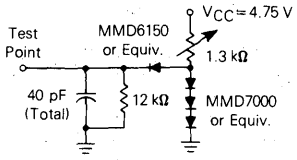


Figure 8-1. TTL Equivalent Test Load (Port B)

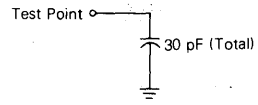


Figure 8-2. CMOS Equivalent Test Low (Port A)

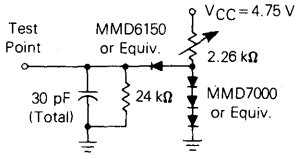


Figure 8-3. TTL Equivalent Test Load (Ports A and C)

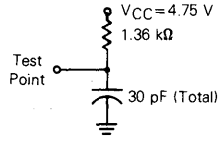


Figure 8-4. Open-Drain Equivalent Test Load (PB1, PB2, and PB3)

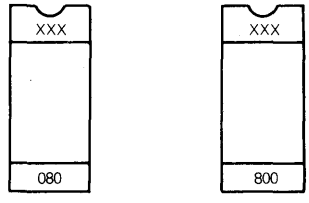
SECTION 9
ORDERING INFORMATION

The information required when ordering a custom MCU is given in the following paragraphs. The ROM program may be transmitted to Motorola on EPROM(s) or an MDOS disk file.

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

9.1 EPROMs

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



XXX = Customer ID

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

9.2 VERIFICATION MEDIA

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

9.3 ROM VERIFICATION UNITS (RVUs)

Ten MCUs containing the customer's ROM pattern will be sent for program verification. These units will have been made using the custom mask but are for the purpose of ROM verification only. For expediency they are usually unmarked, packaged in ceramic, and tested only at room temperature and 5 volts. These RVUs are included in the mask charge and are not production parts. The RVUs are thus not guaranteed by Motorola Quality Assurance, and should be discarded after verification is completed.

9.4 FLEXIBLE DISKS

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

MDOS is Motorola's disk operating system available on development systems such as EXORcisers, EXORsets, etc.

MC6805S2 MCU CUSTOM ORDERING INFORMATION

Motorola Use Only

Motorola Part Numbers
MC _____
SC _____

Date _____

Customer PO Number _____

Customer Company _____

_____

Address _____

City _____ State _____ Zip _____

Country _____

Telephone _____ Extension _____

Custom Contact Person _____

Customer Part Number _____

OPTION LIST

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

Standby RAM	Internal Oscillator Input	Port A Output Drive
No Standby	Crystal	CMOS and TTL
16-Byte Standby	Resistor	TTL Only

Analog-to-Digital Channels	Low Voltage Inhibit
Four (Standard)	Disable
Five	Enable

Pattern Media (All other media require prior factory approval)

EPROMs (MCM2716 or MCM2532)

Floppy Disk

Other* _____

_____

*Requires prior factory approval

Clock Frequency _____

Temperature Range: 0°C to +70°C (Standard) −40°C to +85°C

Marking Information (12 Characters Maximum) _____

Signature _____

Title _____

GENERIC INFORMATION

Package Type	Temperature	Generic Number
Plastic	0°C to 70°C	MC6805S2P
P Suffix	−40°C to 85°C	MC6805S2CP
Ceramic	0°C to 70°C	MC6805S2L
L Suffix	−40°C to 85°C	MC6805S2CL
Cerdip	0°C to 70°C	MC6805S2S
S Suffix	−40°C to 85°C	MC6805S2CS

SECTION 10
MECHANICAL DATA

This section contains the pin assignments and package dimensions for the MC6805S2.

10.1 PIN ASSIGNMENT