MOTOROLA

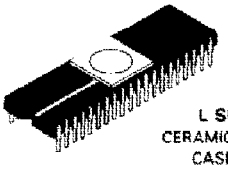
MC68701

Advance Information

MOS

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

MICROCOMPUTER WITH EPROM



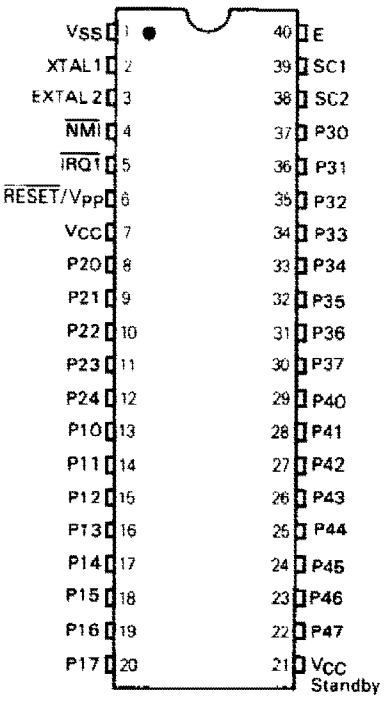
L SUFFIX
CERAMIC PACKAGE
CASE 715

MC68701 MICROCOMPUTER UNIT (MCU)

The MC68701 is an 8-bit single chip microcomputer unit (MCU) which significantly enhances the capabilities of the M6800 family of parts. It can be used in production systems to allow for easy firmware changes with minimum delay or it can be used to emulate the MC6801/03 for software development. It includes an upgraded M6800 microprocessor unit (MPU) with upward source and object code compatibility. Execution times of key instructions have been improved and several new instructions have been added including an unsigned multiply. The MCU can function as a monolithic microcomputer or can be expanded to a 64K byte address space. It is TTL compatible and requires one +5 volt power supply for nonprogramming operation. An additional V_{PP} power supply is needed for EPROM programming. On-chip resources include 2048 bytes of EPROM, 128 bytes of RAM, Serial Communications Interface (SCI), parallel I/O, and a three function Programmable Timer. A summary of MCU features includes:

- Enhanced MC6800 Instruction Set
- 8×8 Multiply Instruction
- Serial Communications Interface (SCI)
- Upward Source and Object Code Compatibility with the MC6800
- 16-Bit Three-Function Programmable Timer
- Single-Chip or Expanded Operation to 64K Byte Address Space
- Bus Compatibility with the M6800 Family
- 2048 Bytes of UV Erasable, User Programmable ROM (EPROM)
- 128 Bytes of RAM (64 Bytes Retainable on Powerdown)
- 29 Parallel I/O and Two Handshake Control Lines
- Internal Clock Generator with Divide-by-Four Output
- −40 to 85°C Temperature Range

PIN ASSIGNMENT

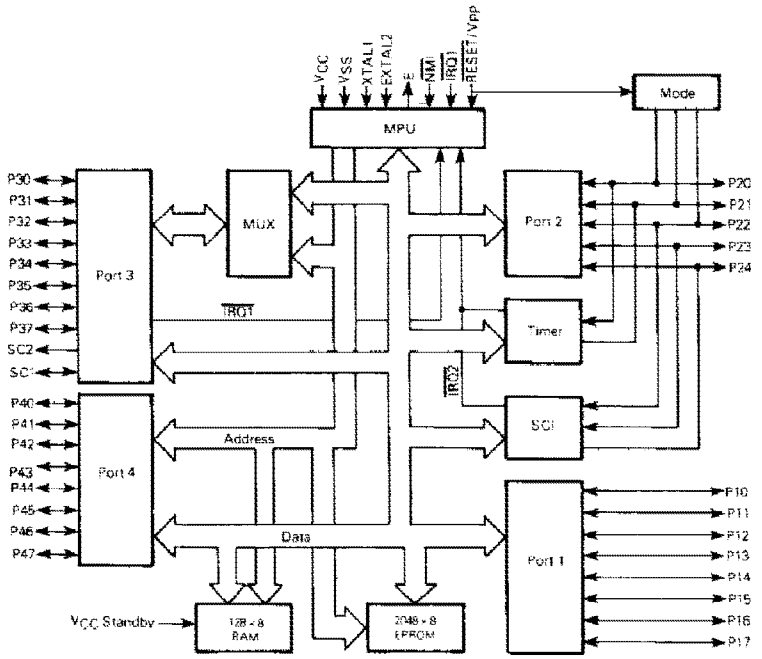


GENERIC INFORMATION

Package Type	Frequency (MHz)	Temperature	Generic Number
Ceramic L Suffix	1.0	0°C to 70°C	MC68701L
	1.0	−40°C to 85°C	MC68701CL
	1.25	0°C to 70°C	MC68701L-1
	1.25	−40°C to 85°C	MC68701CL-1
	1.5	0°C to 70°C	MC68A701L
	2.0	0°C to 70°C	MC68B701L

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

MC68701 MICROCOMPUTER BLOCK DIAGRAM



MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range MC68701 MC68701C	T_A	T_L to T_H 0 to 70 −40 to 85	°C
Storage Temperature Range	T_{stg}	0 to 85	°C

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Rating
Thermal Resistance Ceramic Package	θ_{JA}	50	°C/W

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

POWER CONSIDERATIONS

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

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

Where:

T_A = Ambient Temperature, °C

θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W

$P_D = P_{INT} + P_{PORT}$

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

P_{PORT} = Port Power Dissipation, Watts — User Determined

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

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

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

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.

CONTROL TIMING (V_{CC} = 5.0 V ±5%, V_{SS} = 0, T_A = 0 to 70°C)

Characteristic	Symbol	MC68701		MC68701-1		MC68A701		MC68B701		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	
Frequency of Operation	f_o	0.5	1.0	0.5	1.25	0.5	1.5	0.5	2.0	MHz
Crystal Frequency	f_{XTAL}	2.0	4.0	2.0	5.0	2.0	6.0	2.0	8.0	MHz
External Oscillator Frequency	$4f_o$	2.0	4.0	2.0	5.0	2.0	6.0	2.0	8.0	MHz
Crystal Oscillator Start Up Time	t_{rc}	–	100	–	100	–	100	–	100	ms
Processor Control Setup Time	t_{PCS}	200	–	170	–	140	–	110	–	ns

DC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc ±5%, V_{SS} = 0, T_A = T_L to T_H, unless otherwise noted)

Characteristic		Symbol	MC68701			MC68701C			Unit
			Min	Typ	Max	Min	Typ	Max	
Input High Voltage	$\overline{RESET}$	V_{IH}	V_{SS} + 4.0	–	V_{CC}	V_{SS} + 4.0	–	V_{CC}	V
	Other Inputs*		V_{SS} + 2.0	–	V_{CC}	V_{SS} + 2.2	–	V_{CC}	
Input Low Voltage	$\overline{RESET}$	V_{IL}	V_{SS} − 0.3	–	V_{SS} + 0.4	V_{SS} − 0.3	–	V_{SS} + 0.4	V
	Other Inputs*		V_{SS} − 0.3	–	V_{SS} + 0.8	V_{SS} − 0.3	–	V_{SS} + 0.8	
Input Current, See Note (V_{in} = 0 to 2.4 V)	Port 4	I_{in}	–	–	0.6	–	–	1.0	mA
	SCI		–	–	1.0	–	–	1.6	
Input Current (V_{in} = 0 to 5.25 V)	$\overline{NMI}$, $\overline{IRQ1}$	I_{in}	–	1.5	2.5	–	1.5	5	μA
Input Current (V_{in} = 0 to 0.4 V)	$\overline{RESET}$/V_{PP}	I_{in}	–	−2.0	–	–	−2.0	–	mA
(V_{in} = 4.0 V to V_{CC})			–	–	8.0	–	–	8.0	
Hi-Z (Off State) Input Current (V_{in} = 0.5 to 2.4 V)	Ports 1, 2, and 3	I_{TSI}	–	2	10	–	2	20	μA
Output High Voltage (I_{Load} = −65 μA, V_{CC} = Min)	Port 4, SC1, SC2	V_{OH}	V_{SS} + 2.4	–	–	V_{SS} + 2.4	–	–	V
(I_{Load} = −100 μA, V_{CC} = Min)	Other Outputs		V_{SS} + 2.4	–	–	V_{SS} + 2.4	–	–	
Output Low Voltage (I_{Load} = 2.0 mA, V_{CC} = Min)	All Outputs	V_{OL}	–	–	V_{SS} + 0.5	–	–	V_{SS} + 0.6	V
Darlington Drive Current (V_O = 1.5 V)	Port 1	I_{OH}	1.0	2.5	10.0	1.0	2.5	10.0	mA
Internal Power Dissipation (Measured at T_A = T_L in Steady-State Operation)		P_{INT}	–	–	1500	–	–	1500	mW
Input Capacitance (V_{in} = 0, T_A = 25°C, f_o = 1 MHz)	Port 3, Port 4, SCI	C_{in}	–	–	12.5	–	–	12.5	pF
	Other Inputs		–	–	10.0	–	–	10.0	
V_{CC} Standby	Powerdown	V_{SBB}	4.0	–	5.25	4.0	–	5.25	V
	Powerup	V_{SB}	4.75	–	5.25	4.75	–	5.25	
Standby Current	Powerdown	I_{SBB}	–	–	6.0	–	–	8.0	mA
Programming Time Per Byte (T_A = 25°C)		t_{PP}	25	–	50	25	–	50	ms
Programming Voltage (T_A = 25°C)		V_{PP}	20.0	21.0	22.0	20.0	21.0	22.0	V
Programming Current ($V_{\overline{RESET}}$ = V_{PP}, T_A = 25°C)		I_{PP}	–	30	50	–	30	50	mA

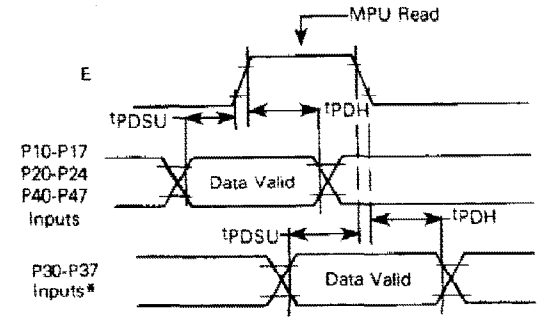
* Except mode programming levels; see Figure 15.

NOTE: $\overline{RESET}$/V_{PP} I_{in} differs from MC6801 and MC6803 values.

PERIPHERAL PORT TIMING (Refer to Figures 3-6)

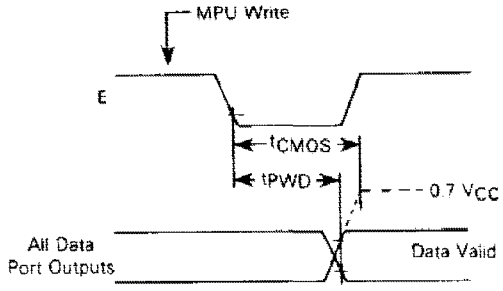
Characteristics	Symbol	MC68701		MC68701-1		MC68A701		MC68B701		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	
Peripheral Data Setup Time	t_{PDSU}	200	–	200	–	150	–	100	–	ns
Peripheral Data Hold Time	t_{PDH}	200	–	200	–	150	–	100	–	ns
Delay Time, Enable Positive Transition to $\overline{OS3}$ Negative Transition	t_{OSD1}	–	350	–	350	–	300	–	250	ns
Delay Time, Enable Positive Transition to $\overline{OS3}$ Positive Transition	t_{OSD2}	–	350	–	350	–	300	–	250	ns
Delay Time, Enable Negative Transition to Peripheral Data Valid	t_{PWD}	–	350	–	350	–	300	–	250	ns
Delay Time, Enable Negative Transition to Peripheral CMOS Data Valid	t_{CMOS}	–	2.0	–	2.0	–	2.0	–	2.0	μs
Input Strobe Pulse Width	t_{PWIS}	200	–	200	–	150	–	100	–	ns
Input Data Hold Time	t_{IH}	50	–	50	–	40	–	30	–	ns
Input Data Setup Time	t_{IS}	20	–	20	–	20	–	20	–	ns

FIGURE 1 — DATA SETUP AND HOLD TIMES (MPU READ)



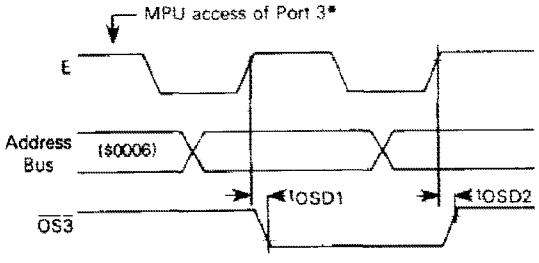
* Port 3 Non-Latched Operation (LATCHE ENABLE = 0)

FIGURE 2 — DATA SETUP AND HOLD TIMES (MPU WRITE)



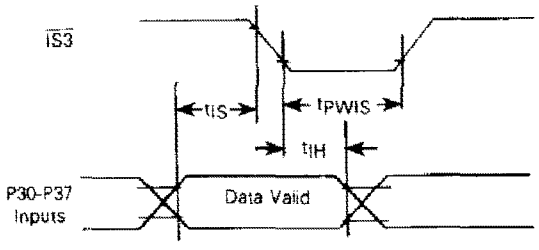
NOTES:
1. 10 k Pullup resistor required for Port 2 to reach 0.7 V_{CC}
2. Not applicable to P21
3. Port 4 cannot be pulled above V_{CC}

FIGURE 3 — PORT 3 OUTPUT STROBE TIMING (SINGLE-CHIP MODE)



* Access matches Output Strobe Select (OSS = 0, a read; OSS = 1, a write)

FIGURE 4 — PORT 3 LATCH TIMING (SINGLE-CHIP MODE)



NOTE: Timing measurements are referenced to a low voltage of 0.8 volts and a high voltage of 2.0 volts unless otherwise noted.

FIGURE 5 — CMOS LOAD

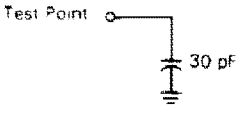
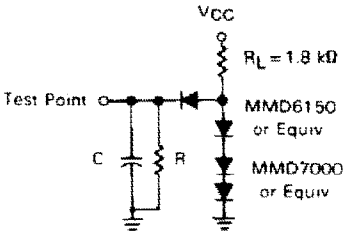


FIGURE 6 — TIMING TEST LOAD PORTS 1, 2, 3, 4



C = 90 pF for P30-P37, P40-P47, E, SC1, SC2
= 30 pF for P10-P17, P20-P24
R = 37 kΩ for P40-P47, SC1, SC2,
= 24 kΩ for P10-P17, P20-P24, P30-P37, E

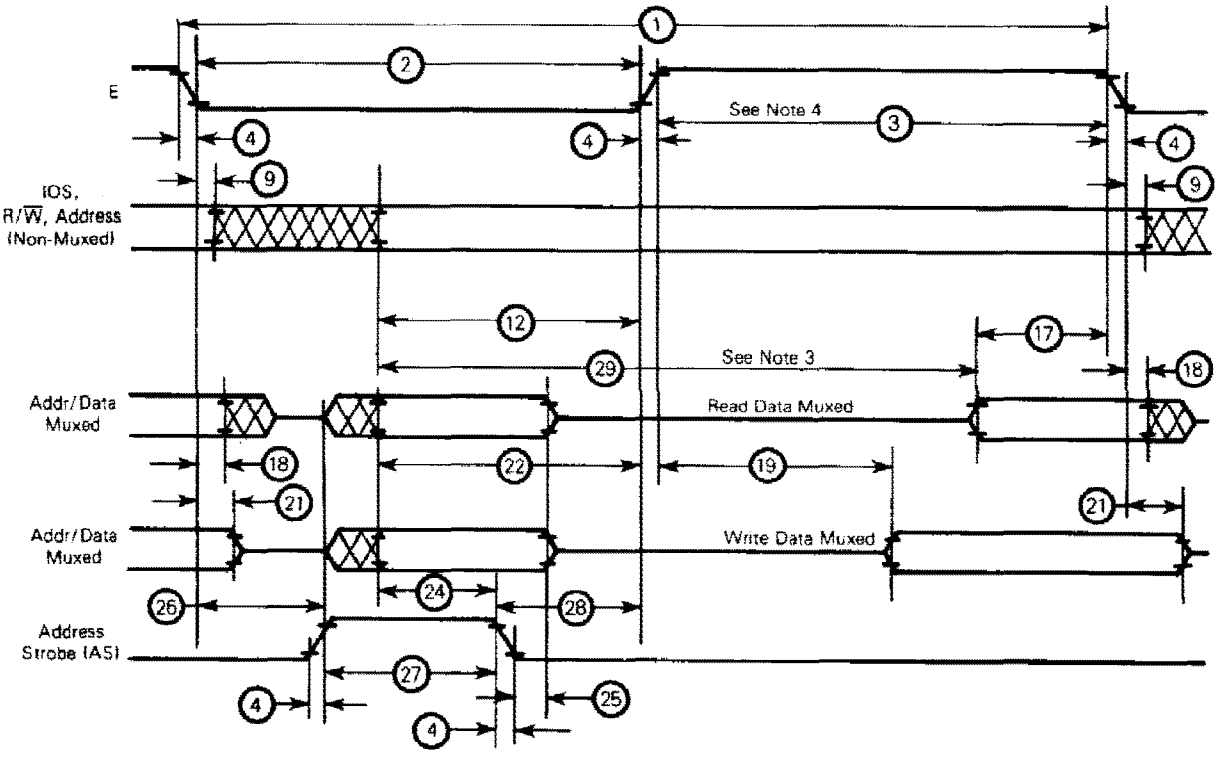
BUS TIMING (See Notes 2 and 3)

Ident Number	Characteristic	Symbol	MC68701		MC68701-1		MC68A701		MC68B701		Unit
			Min	Max	Min	Max	Min	Max	Min	Max	
1	Cycle Time	t_{cyc}	1.0	2.0	0.8	2.0	–	2.0	0.5	2.0	μs
2	Pulse Width, E Low	PW_{EL}	430	1000	360	1000	300	1000	210	1000	ns
3	Pulse Width, E High	PW_{EH}	450	1000	360	1000	300	1000	220	1000	ns
4	Clock Rise and Fall Time	t_r, t_f	–	25	–	25	–	25	–	20	ns
9	Address Hold Time	t_{AH}	20	–	20	–	20	–	10	–	ns
12	Non-Muxed Address Valid Time to E*	t_{AV}	200	–	150	–	115	–	70	–	ns
17	Read Data Setup Time	t_{DSR}	80	–	70	–	60	–	40	–	ns
18	Read Data Hold Time	t_{DHR}	10	-	10	–	10	–	10	–	ns
19	Write Data Delay Time	t_{DDW}	–	225	–	200	–	170	–	120	ns
21	Write Data Hold Time	t_{DHW}	20	–	20	–	20	–	10	–	ns
22	Multiplexed Address Valid Time to E Rise*	t_{AVM}	200	–	150	–	115	–	80	–	ns
24	Multiplexed Address Valid Time to AS Fall*	t_{ASL}	60	–	50	–	40	–	20	–	ns
25	Multiplexed Address Hold time	t_{AHL}	20	–	20	–	20	–	10	–	ns
26	Delay Time, E to AS Rise*	t_{ASD}	90**	–	70**	–	60**	–	45**	–	ns
27	Pulse Width, AS High*	PW_{ASH}	220	–	170	–	140	–	110	–	ns
28	Delay Time, AS to E Rise*	t_{ASED}	90	–	70	–	60	–	45	–	ns
29	Usable Access Time*	t_{ACC}	595	–	465	–	380	–	270	–	ns

*At specified cycle time.

**t_{ASD} parameters listed assume external TTL clock drive with 50% ±5% duty cycle. Devices driven by an external TTL clock with 50% ±1% duty cycle or which use a crystal have the following t_{ASD} specification: 100 nanoseconds minimum (1.0 MHz devices), 80 nanoseconds minimum (1.25 MHz devices), 65 nanoseconds minimum (1.5 MHz devices), 50 nanoseconds minimum (2.0 MHz devices).

FIGURE 7 — BUS TIMING



NOTES:

1. Voltage levels shown are $V_L \leq 0.6$ V, $V_H \geq 2.4$ V, unless otherwise specified.
2. Measurement points shown are 0.8 V and 2.0 V, unless otherwise specified.
3. Usable access time is computed by 12+3−17+4.
4. Memory devices should be enabled only during E high to avoid port 3 bus contention.

INTRODUCTION

The MC68701 is an 8-bit monolithic microcomputer which can be configured to function in a wide variety of applications. The facility which provides this extraordinary flexibility is its ability to be hardware programmed into eight different operating modes. The operating mode controls the configuration of 18 of the 40 MCU pins, available on-chip resources, memory map, location (internal or external) of interrupt vectors, and type of external bus. The configuration of the remaining 22 pins is not dependent on the operating mode.

Twenty-nine pins are organized as three 8-bit ports and one 5-bit port. Each port consists of at least a Data Register and a write-only Data Direction Register. The Data Direction Register is used to define whether corresponding bits in the Data Register are configured as an input (clear) or output (set).

The term "port," by itself, refers to all of the hardware associated with the port. When the port is used as a "data port" or "I/O port," it is controlled by the port Data Direction Register and the programmer has direct access to the port pins using the port Data Register. Port pins are labled as P_{ij} where i identifies one of four ports and j indicates the particular bit.

The Microprocessor Unit (MPU) is an enhanced MC6800 MPU with additional capabilities and greater throughput. It is upward source and object code compatible with the MC6800. The programming model is depicted in Figure 8 where Accumulator D is a concatenation of Accumulators A and B. A list of new operations added to the M6800 instruction set are shown in Table 1.

The basic difference between the MC6801 and the MC68701 is that the MC6801 has an onboard ROM while the MC68701 has an onboard EPROM. The MC68701 is pin and code compatible with the MC6801 and can be used to emulate the MC6801, allowing easy software development using the onboard EPROM. Software developed using the MC68701 can then be masked into the MC6801 ROM.

In order to support the onboard EPROM, the MC68701 differs from the MC6801 as follows:

(1) Mode 0 in the MC6801 is a test mode only, while in the MC68701 Mode 0 is also used to program the onboard EPROM and has interrupt vectors at $BFF0-$BFFF rather than $FFF0-$FFFF.

(2) The MC68701 RAM/EPROM Control Register has two bits used to control the EPROM in Mode 0 that are not defined in the MC6801 RAM Control Register.

(3) The $\overline{RESET}$/Vpp pin in the MC68701 is dual purpose, used to supply EPROM power as well as to reset the device; while in the MC6801 the pin is called $\overline{RESET}$ and is used only to reset the device.

FIGURE 8 — MC68701/6801/6803 PROGRAMMING MODEL

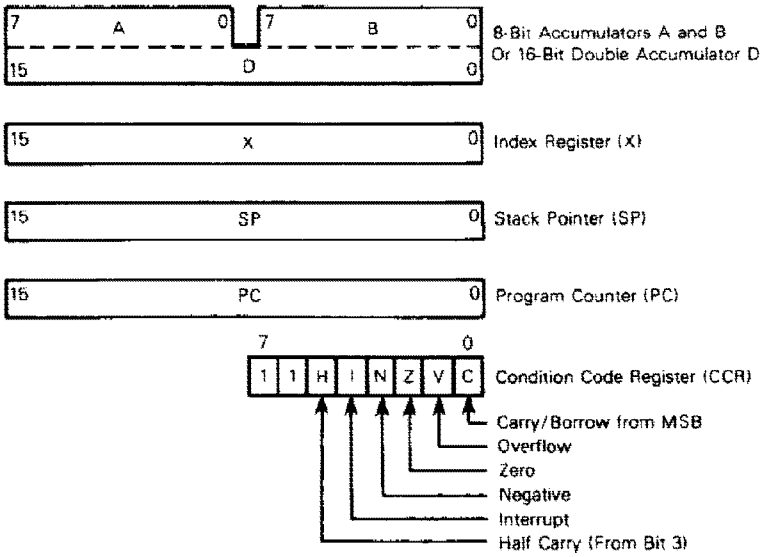


TABLE 1 — NEW INSTRUCTIONS

Instruction	Description
ABX	Unsigned addition of Accumulator B to Index Register
ADDD	Adds (without carry) the double accumulator to memory and leaves the sum in the double accumulator
ASLD or LSLD	Shifts the double accumulator left (towards MSB) one bit, the LSB is cleared and the MSB is shifted into the C-bit
BHS	Branch if Higher or Same, unsigned conditional branch (same as BCC)
BLO	Branch if Lower, Unsigned conditional branch (same as BCS)
BRN	Branch Never
JSR	Additional addressing mode: direct
LDD	Loads double accumulator from memory
LSL	Shifts memory or accumulator left (towards MSB) one bit; the LSB is cleared and the MSB is shifted into the C-bit (same as ASL)
LSRD	Shifts the double accumulator right (towards LSB) one bit; the MSB is cleared and the LSB is shifted into the C-bit
MUL	Unsigned multiply, multiplies the two accumulators and leaves the product in the double accumulator
PSHX	Pushes the Index Register to stack
PULX	Pulls the Index Register from stack
STD	Stores the double accumulator to memory
SUBD	Subtracts memory from the double accumulator and leaves the difference in the double accumulator
CPX	Internal processing modified to permit its use with any conditional branch instruction

OPERATING MODES

The MCU provides eight different operating modes which are selectable by hardware programming and referred to as Mode 0 through Mode 7. The operating mode controls the memory map, configuration of Port 3, Port 4, SC1, SC2, and the physical location of interrupt vectors.

FUNDAMENTAL MODES

The eight MCU modes can be grouped into three fundamental modes which refer to the type of bus it supports: Single Chip, Expanded Non-Multiplexed, and Expanded Multiplexed. Modes 4 and 7 are single chip modes. Mode 5 is the expanded non-multiplexed mode, and the remaining modes are expanded multiplexed modes. Table 2 summarizes the characteristics of the operating modes.

Single-Chip Modes (4, 7)

In the Single-Chip Mode, the four MCU ports are configured as parallel input/output data ports, as shown in Figure 9. The MCU functions as a monolithic microcomputer in these two modes without external address or data buses. A maximum of 29 I/O lines and two Port 3 control lines are provided. Peripherals or another MCU can be interfaced to Port 3 in a loosely coupled dual processor configuration, as shown in Figure 10.

In Single-Chip Test Mode (4), the RAM responds to $XX80 through $XXFF and the EPROM is removed from the internal address map. A test program must first be loaded into the RAM using modes 0, 1, 2, or 6. If the MCU is reset and then programmed into Mode 4, execution will begin at $XXFE:XXFF. Mode 5 can be irreversibly entered from Mode 4 without asserting $\overline{\text{RESET}}$ by setting bit 5 of the Port 2 Data Register. This mode is used primarily to test Ports 3 and 4 in the Single-Chip and Non-Multiplexed Modes.

TABLE 2 — SUMMARY OF MC68701 OPERATING MODES

Common to all Modes:
- Reserved Register Area
- Port 1
- Port 2
- Programmable Timer
- Serial Communications Interface

Single Chip Mode 7
- 128 bytes of RAM; 2048 bytes of EPROM
- Port 3 is a parallel I/O port with two control lines
- Port 4 is a parallel I/O port
- SC1 is Input Strobe 3 (IS3)
- SC2 is Output Strobe 3 (OS3)

Expanded Non-Multiplexed Mode 5
- 128 bytes of RAM; 2048 bytes of EPROM
- 256 bytes of external memory space
- Port 3 is an 8-bit data bus
- Port 4 is an input port/address bus
- SC1 is Input/Output Select (IOS)
- SC2 is Read/Write (R/W)

Expanded Multiplexed Modes 1, 2, 3, 6
- Four memory space options (64K address space):
  - (1) No internal RAM or EPROM (Mode 3)
  - (2) Internal RAM, no EPROM (Mode 2)
  - (3) Internal RAM and EPROM (Mode 1)
  - (4) Internal RAM, EPROM with partial address bus (Mode 6)
- Port 3 is a multiplexed address/data bus
- Port 4 is an address bus (inputs/address in Mode 6)
- SC1 is Address Strobe ($\overline{\text{AS}}$)
- SC2 is Read/Write (R/$\overline{\text{W}}$)

Test Mode 4
- (1) May be changed to Mode 5 without going through Reset
- (2) May be used to test Ports 3 and 4 as I/O ports

Expanded Multiplexed Mode 0
- (1) Internal RAM and EPROM
- (2) External interrupt vectors located at $BFF0-$BFFF
- (3) Used to program EPROM

FIGURE 9 — SINGLE-CHIP MODE

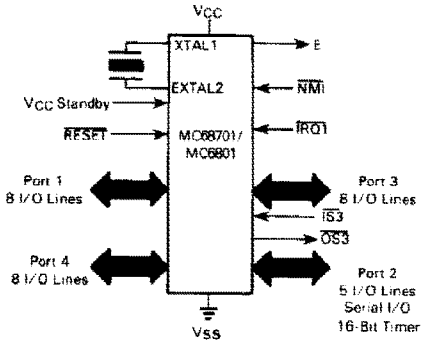


FIGURE 10 — SINGLE-CHIP DUAL PROCESSOR CONFIGURATION

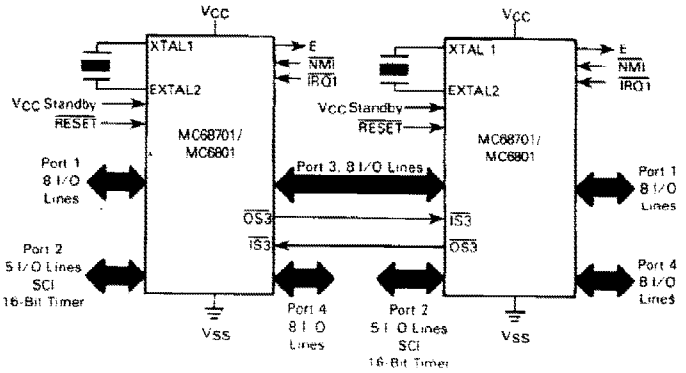
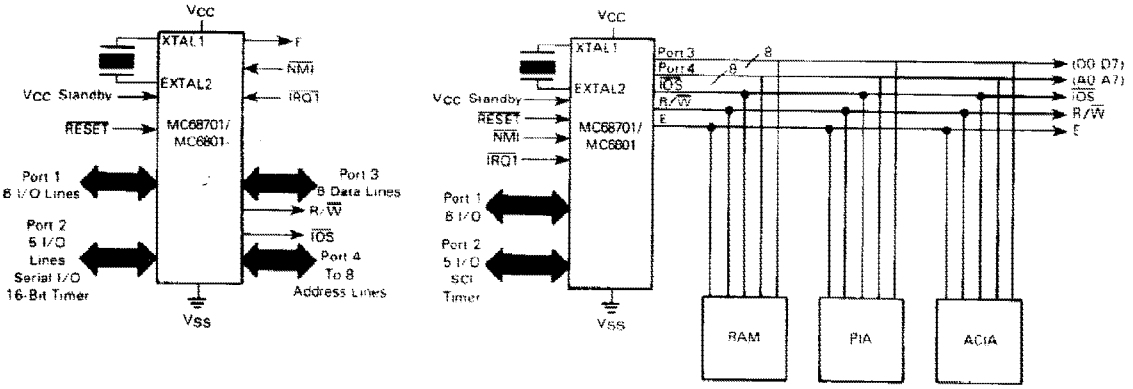


FIGURE 11 — EXPANDED NON-MULTIPLEXED CONFIGURATION



Expanded Non-Multiplexed Mode (5)

A modest amount of external memory space is provided in the Expanded Non-Multiplexed Mode while significant on-chip resources are retained. Port 3 functions as an 8-bit bidirectional data bus and Port 4 is configured initially as an input data port. Any combination of the eight least-significant address lines may be obtained by writing to the Port 4 Data Direction Register. Stated alternatively, any combination of A0 to A7 may be provided while retaining the remainder as input data lines. Internal pullup resistors are intended to pull the Port 4 lines high until the port is configured.

Figure 11 illustrates a typical system configuration in the Expanded Non-Multiplexed Mode. The MCU interfaces directly with M6800 family parts and can access 256 bytes of external address space at $100 through $1FF. $\overline{IOS}$ provides an address decode of external memory ($100-$1FF) and can be used as a memory page select or chip select line.

Expanded-Multiplexed Modes (0, 1, 2, 3, 6)

In the Expanded-Multiplexed Modes, the MCU has the ability to access a 64K byte memory space. Port 3 functions as a time multiplexed address/data bus with address valid on the negative edge of Address Strobe (AS), and data valid while E is high. In Modes 0 to 3, Port 4 provides address lines A8 to A15. In Mode 6, however, Port 4 is initially configured at $\overline{RESET}$ as an input data port. The Port 4 Data Direction Register can then be changed to provide any combination of address lines, A8 to A15. Stated alternatively, any subset of A8 to A15 can be provided while retaining the remaining Port 4 lines as input data lines. Internal pullup resistors pull the Port 4 lines high until software configures the port.

Figure 12 depicts a typical configuration for the Expanded-Multiplexed Modes. Address Strobe can be used to control a transparent D-type latch to capture addresses A0 to A7, as shown in Figure 13. This allows Port 3 to function as a Data Bus when E is high.

In Mode 0, the internal and external data buses are connected; there must therefore be no memory map overlap in order to avoid potential bus conflicts. Mode 0 is used to program the onboard EPROM. All interrupt vectors are external in this mode and are located at $BFF0-$BFFF.

PROGRAMMING THE MODE

The operating mode is determined at $\overline{RESET}$ by the levels asserted on P22, P21, and P20. These levels are latched into PC2, PC1, and PC0 of the program control register on the positive edge of $\overline{RESET}$. The operating mode may be read from the Port 2 Data Register as shown below, and programming levels and timing must be met as shown in Figure 14. A brief outline of the operating modes is shown in Table 3.

PORT 2 DATA REGISTER

7	6	5	4	3	2	1	0	
PC2	PC1	PC0	P24	P23	P22	P21	P20	$0003

Circuitry to provide the programming levels is dependent primarily on the normal system usage of the three pins. If configured as outputs, the circuit shown in Figure 15 may be used; otherwise, three-state buffers can be used to provide isolation while programming the mode.

MEMORY MAPS

The MCU can provide up to 64K byte address space depending on the operating mode. A memory map for each operating mode is shown in Figure 16. The first 32 locations of each map are reserved for the MCU internal registers as shown in Table 4, with exceptions as indicated.

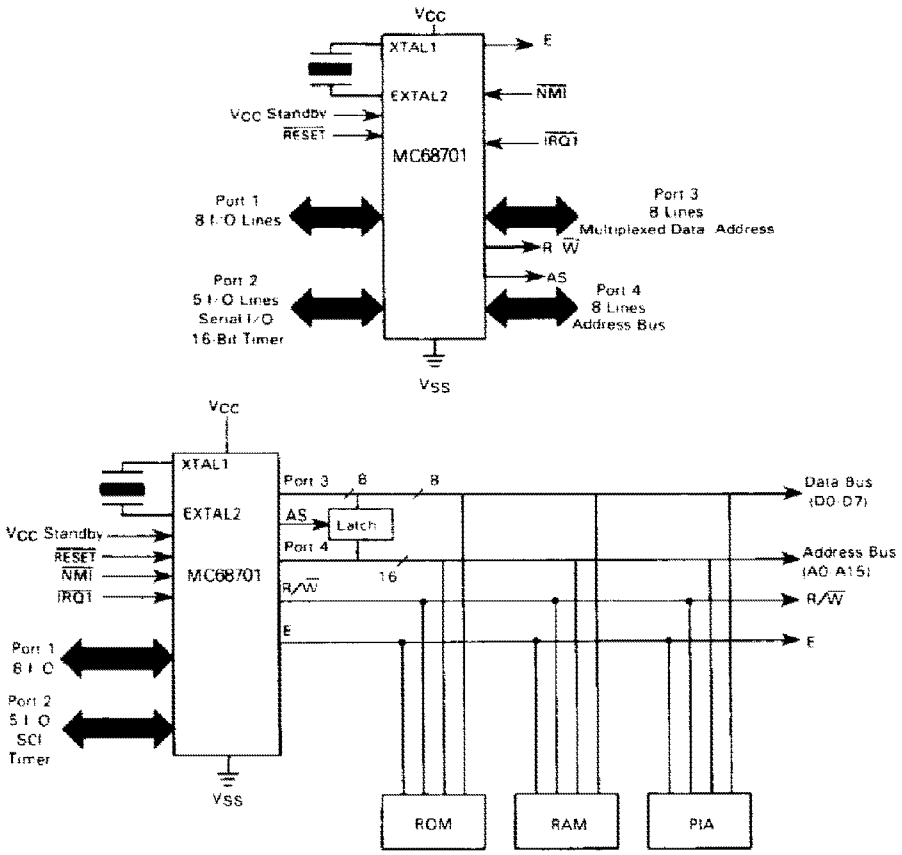
TABLE 3 — MODE SELECTION SUMMARY

Mode	P22 PC2	P21 PC1	P20 PC0	EPROM	RAM	Interrupt Vectors	Bus Mode	Operating Mode
7	H	H	H	I	I	I	I	Single Chip
6	H	H	L	I	I	I	MUX(5, 6)	Multiplexed/Partial Decode
5	H	L	H	I	I	I	NMUX(5, 6)	Non-Multiplexed/Partial Decode
4	H	L	L	I(2)	I(1)	I	I	Single Chip Test
3	L	H	H	E	E	E	MUX(4)	Multiplexed/No RAM or EPROM
2	L	H	L	E	I	E	MUX(4)	Multiplexed/RAM
1	L	L	H	I	I	E	MUX(4)	Multiplexed/RAM and EPROM
0	L	L	L	I	I	I(3)	MUX(4)	Multiplexed/Programming

Legend:
- I — Internal
- E — External
- MUX — Multiplexed
- NMUX — Non-Multiplexed
- L — Logic "0"
- H — Logic "1"

Notes:
(1) Internal RAM is addressed at $XX80
(2) Internal EPROM is disabled
(3) Interrupt vectors located at $BFF0-$BFFF
(4) Addresses associated with Ports 3 and 4 are considered external in Modes 0, 1, 2, and 3
(5) Addresses associated with Port 3 are considered external in Modes 5 and 6
(6) Port 4 default is user data input; address output is optional by writing to Port 4 Data Direction Register

FIGURE 12 — EXPANDED MULTIPLEXED CONFIGURATION



NOTE: To avoid data bus (Port 3) contention in the expanded multiplexed modes, memory devices should be enabled only during E high time.

FIGURE 13 — TYPICAL LATCH ARRANGEMENT

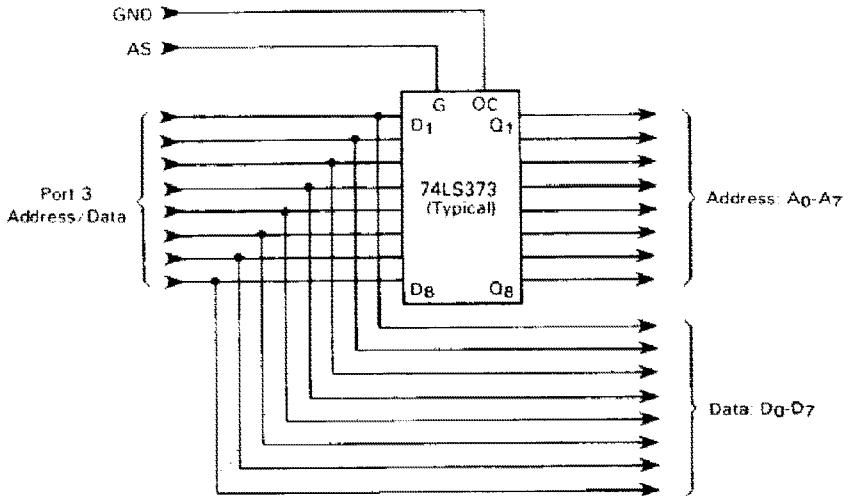
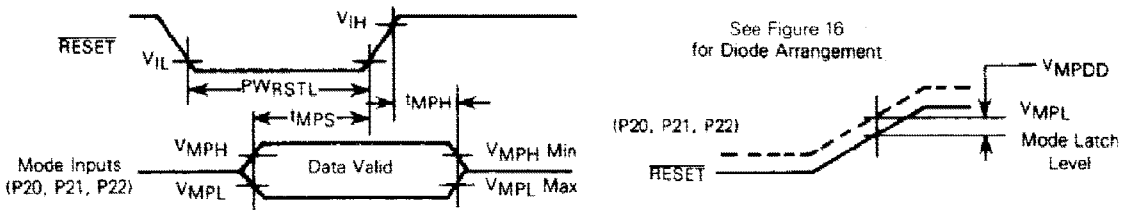


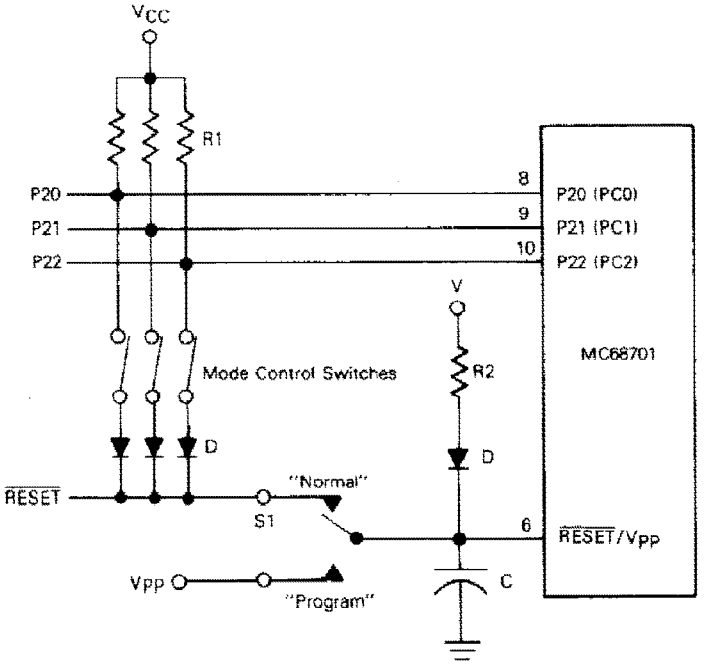
FIGURE 14 — MODE PROGRAMMING TIMING



MODE PROGRAMMING (Refer to Figure 14)

Characteristic	Symbol	Min	Typ	Max	Unit
Mode Programming Input Voltage Low	V_{MPL}	—	—	1.8	V
Mode Programming Input Voltage High	V_{MPH}	4.0	—	—	V
Mode Programming Diode Differential	V_{MPDD}	0.6	—	—	V
$\overline{RESET}$ Low Pulse Width	PW_{RSTL}	3.0	—	—	E-Cycles
Mode Programming Set-Up Time	t_{MPS}	2.0	—	—	E-Cycles
Mode Programming Hold Time $\overline{RESET}$ Rise Time ≥ 1 μs $\overline{RESET}$ Rise Time < 1 μs	t_{MPH}	 0 100	 — —	 — —	ns

FIGURE 15 — TYPICAL MODE PROGRAMMING CIRCUIT



Notes:

1. Mode 0 as shown (switches closed).
2. R1 = 10k ohms (typical).
3. The $\overline{RESET}$ time constant is equal to RC where R is the equivalent parallel resistance of R2 and the number of resistors (R1) placed in the circuit by closed mode control switches.
4. D = 1N914, 1N4001 (typical).
5. If $V = V_{CC}$, then R2 = 50 ohms (typical) to meet V_{IH} for the $\overline{RESET}/V_{PP}$ pin. $V = V_{CC}$ is also compatible with MC6801. The $\overline{RESET}$ time constant in this case is approximately R2*C.
6. Switch S1 allows selection of normal ($\overline{RESET}$) or programming (V_{PP}) as the input to the $\overline{RESET}/V_{PP}$ pin. During switching, the input level is held at a value determined by a diode (D), resistor (R2) and input voltage (V).
7. While S1 is in the "Program" position, $\overline{RESET}$ should not be asserted.
8. From powerup, $\overline{RESET}$ must be held low for at least t_{RC}. The capacitor, C, is shown for conceptual purposes only and is on the order of 1000 μF for the circuit shown. Typically, a buffer with an RC input will be used to drive $\overline{RESET}$, eliminating the need for the larger capacitor.
9. Diode V_f should not exceed V_{MPDD} min.

FIGURE 16 — MC68701 MEMORY MAPS

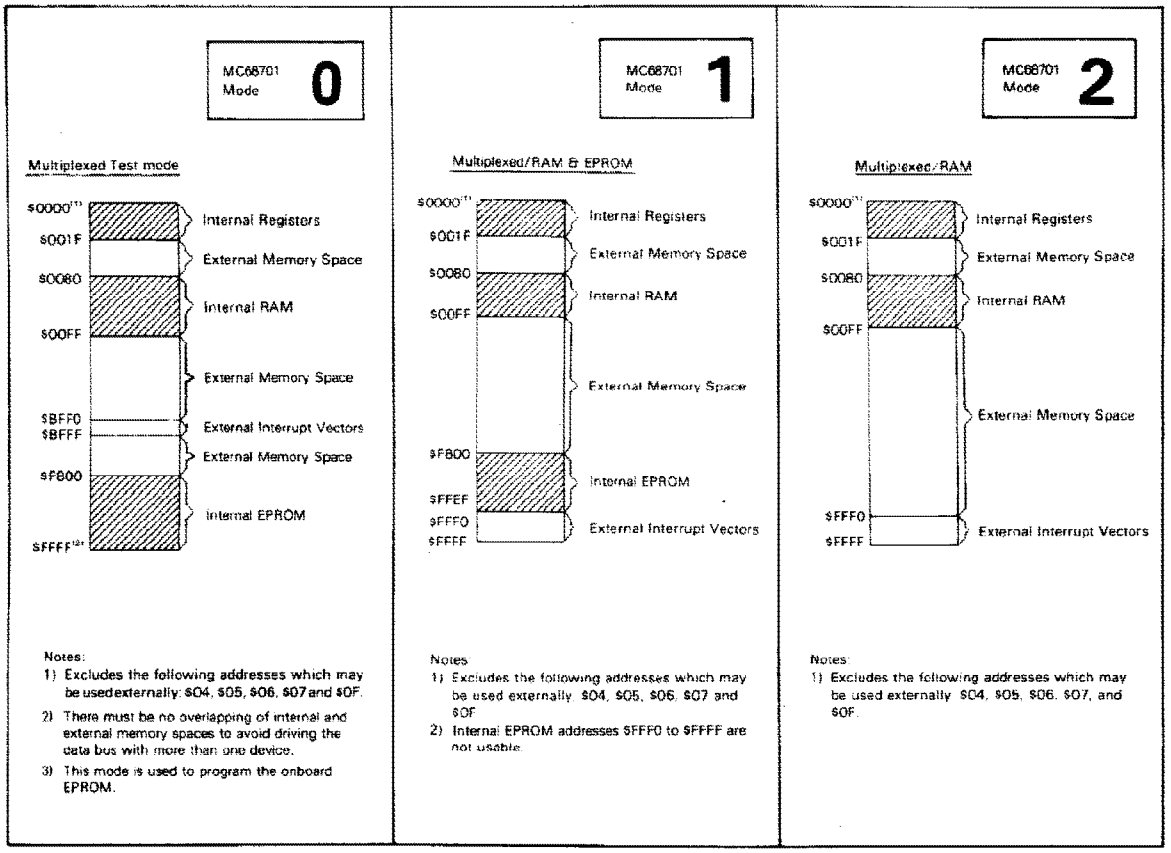
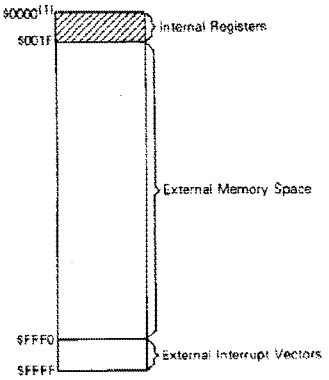


FIGURE 16 — MC68701 MEMORY MAPS (CONTINUED)

MC68701 Mode 3

Multiplexed/No RAM or EPROM

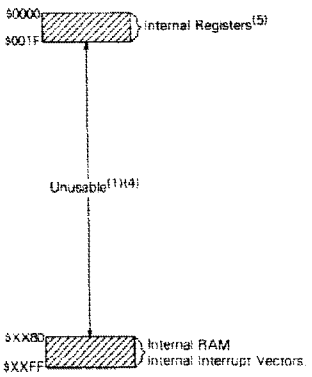


Notes:

1) Excludes the following addresses which may be used externally: $04, $05, $06, $07 and $0F.

MC68701 Mode 4

Single Chip Test



Notes

1) The internal EPROM is disabled.

2) Mode 4 may be changed to Mode 5 without having to assert $\overline{RESET}$ by writing a "1" into the PCO bit of Port 2 Data Register.

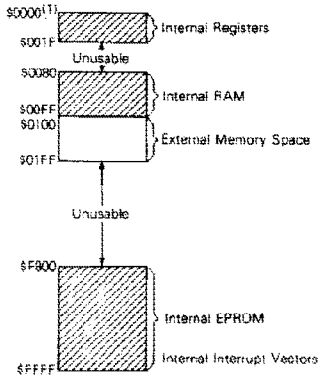
3) Addresses A8 to A15 are treated as "don't cares" to decode internal RAM.

4) Internal RAM will appear at $XX80 to $XXFF.

5) MCU read of the Port 3 Data Direction Register will access the Port 3 Data Register.

MC68701 Mode 5

Non-Multiplexed/Partial Decode



Notes.

1) Excludes the following addresses which may **NOT** be used externally: $04, $06, and $0F (No IOS).

2) This mode may be entered without going through $\overline{RESET}$ by using Mode 4 and subsequently writing a "1" into the PCO bit of Port 2 Data Register.

3) Address lines A0 to A7 will not contain addresses until the Data Direction Register for Port 4 has been written with "1's" in the appropriate bits. These address lines will assert "1's" until made outputs by writing the Data Direction Register.

FIGURE 16 — MC68701 MEMORY MAPS (CONCLUDED)

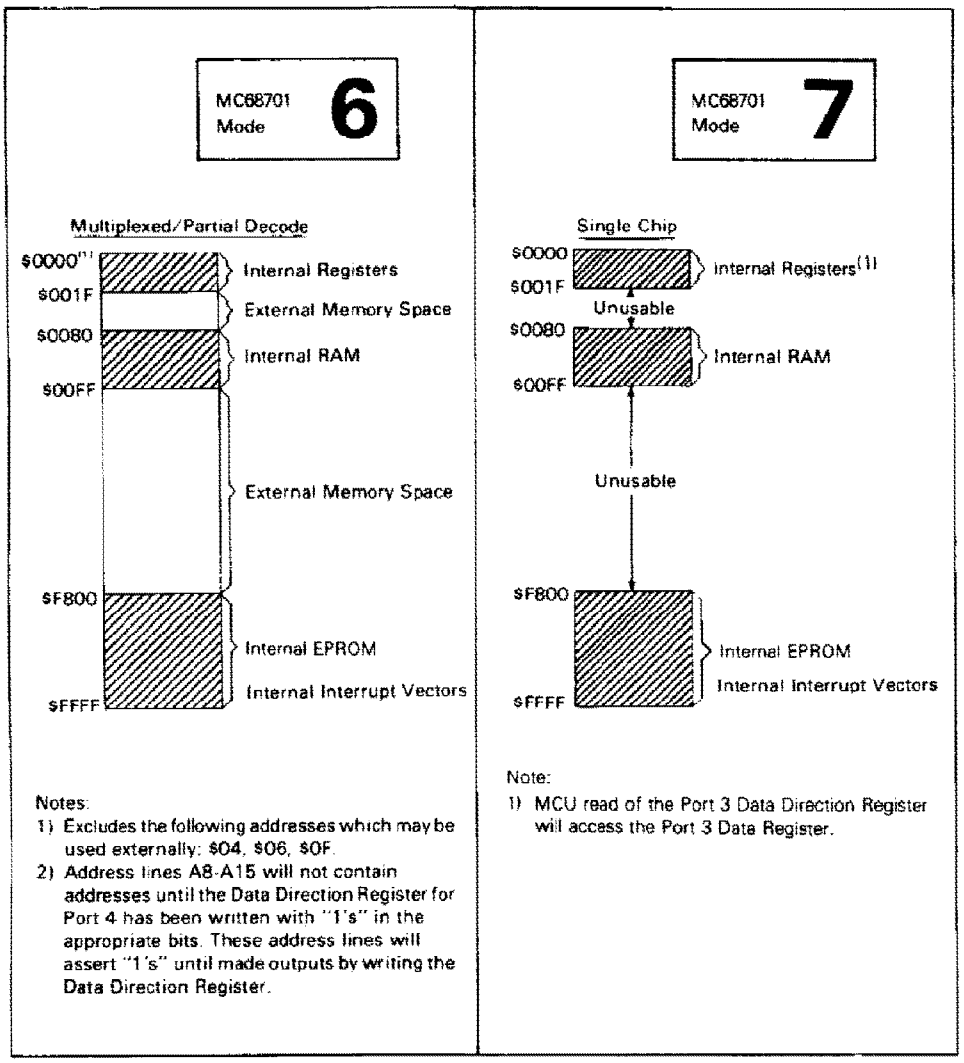


TABLE 4 — INTERNAL REGISTER AREA

Register	Address	Register	Address
Port 1 Data Direction Register* * *	00	Output Compare Register (Low Byte)	0C
Port 2 Data Direction Register* * *	01	Input Capture Register (High Byte)	0D
Port 1 Data Register	02	Input Capture Register (Low Byte)	0E
Port 2 Data Register	03	Port 3 Control and Status Register	0F*
Port 3 Data Direction Register* * *	04*	Rate and Mode Control Register	10
Port 4 Data Direction Register* * *	05* *	Transmit/Receive Control and Status Register	11
Port 3 Data Register	06*	Receive Data Register	12
Port 4 Data Register	07* *	Transmit Data Register	13
Timer Control and Status Register	08	RAM/EPROM Control Register	14
Counter (High Byte)	09	Reserved	15-1F
Counter (Low Byte)	0A		
Output Compare Register (High Byte)	0B		

* External addresses in Modes 0, 1, 2, 3, 5, 6; cannot be accessed in Mode 5 (No $\overline{IOS}$)
* * External addresses in Modes 0, 1, 2, 3
* * * 1 = output, 0 = Input
* * * 1 = Output, 0 = Input

MC68701 INTERRUPTS

The MCU supports two types of interrupt requests: maskable and non-maskable. A Non-Maskable Interrupt ($\overline{NMI}$) is always recognized and acted upon at the completion of the current instruction. Maskable interrupts are controlled by the Condition Code Register's I-bit and by individual enable bits. The I-bit controls all maskable interrupts. Of the maskable interrupts, there are two types: $\overline{IRQ1}$ and $\overline{IRQ2}$. The Programmable Timer and Serial Communications Interface use an internal $\overline{IRQ2}$ interrupt line. External devices (and $\overline{IS3}$) use $\overline{IRQ1}$. An $\overline{IRQ1}$ interrupt is serviced before $\overline{IRQ2}$ if both are pending.

All $\overline{IRQ2}$ interrupts use hardware prioritized vectors. The single SCI interrupt and three timer interrupts are serviced in a prioritized order and each is vectored to a separate location. All MCU interrupt vector locations are shown in Table 5.

TABLE 5 — MCU INTERRUPT VECTOR LOCATIONS

Mode 0		Modes 1-7		
MSB	LSB	MSB	LSB	Interrupt
BFFE	BFFF	FFFE	FFFF	$\overline{RESET}$
BFFC	BFFD	FFFC	FFFD	$\overline{NMI}$
BFFA	BFFB	FFFA	FFFB	Software Interrupt (SWI)
BFF8	BFF9	FFF8	FFF9	$\overline{IRQ1}$ (or $\overline{IS3}$)
BFF6	BFF7	FFF6	FFF7	ICF (Input Capture) *
BFF4	BFF5	FFF4	FFF5	OCF (Output Compare) *
BFF2	BFF3	FFF2	FFF3	TOF (Timer Overflow) *
BFF0	BFF1	FFF0	FFF1	SCI(RDRF+ORFE+TDRE)*

* IRQ2 Interrupt

The interrupt flowchart is depicted in Figure 17 and is common to every MCU interrupt excluding reset. During interrupt servicing the Program Counter, Index Register, A Accumulator, B Accumulator, and Condition Code Register are pushed to the stack. The I-bit is set to inhibit maskable interrupts and a vector is fetched corresponding to the current highest priority interrupt. The vector is transferred to the Program Counter and instruction execution is resumed. Interrupt and $\overline{RESET}$ timing are illustrated in Figures 18 and 19.

FUNCTIONAL PIN DESCRIPTIONS

V_{CC} AND V_{SS}

V_{CC} and V_{SS} provide power to a large portion of the MCU. The power supply should provide +5 volts (±5%) to V_{CC}, and V_{SS} should be tied to ground. Total power dissipation (including V_{CC} Standby), will not exceed P_D milliwatts.

V_{CC} STANDBY

V_{CC} Standby provides power to the standby portion ($80 through $BF) of the RAM and the STBY PWR and RAME bits of the RAM Control Register. Voltage requirements depend on whether the MCU is in a powerup or powerdown state. In the powerup state, the power supply should provide +5 volts (±5%) and must reach V_{SB} volts before $\overline{RESET}$ reaches 4.0 volts. During powerdown, V_{CC} Standby must remain above V_{SBB} (min) to sustain the standby RAM and STBY PWR bit. While in powerdown operation, the standby current will not exceed I_{SBB}.

It is typical to power both V_{CC} and V_{CC} Standby from the same source during normal operation. A diode must be used between them to prevent supplying power to V_{CC} during powerdown operation. V_{CC} Standby should be tied to ground in Mode 3.

XTAL1 AND EXTAL2

These two input pins interface either a crystal or TTL compatible clock to the MCU internal clock generator. Divide-by-four circuitry is included which allows use of the inexpensive 3.58 MHz or 4.4336 MHz Color Burst TV crystals. A 20 pF capacitor should be tied from each crystal pin to ground to ensure reliable startup and operation. Alternatively, EXTAL2 may be driven by an external TTL compatible clock at $4f_o$ with a duty cycle of 50% (±5%) with XTAL1 connected to ground.

The internal oscillator is designed to interface with an AT-cut quartz crystal resonator operated in parallel resonance mode in the frequency range specified for f_{XTAL}. The crystal should be mounted as close as possible to the input pins to minimize output distortion and startup stabilization time.** The MCU is compatible with most commercially available crystals. Nominal crystal parameters are shown in Figure 20.

$\overline{RESET}$/V_{PP}

This input is used to reset the MCU internal state and provide an orderly startup procedure. During powerup, $\overline{RESET}$ must be held below 0.4 volts: (1) at least t_{RC} after V_{CC} reaches 4.75 volts in order to provide sufficient time for the clock generator to stabilize, and (2) until V_{CC} Standby reaches V_{SB} volts. $\overline{RESET}$ must be held low at least three E-cycles if asserted during powerup operation.

This pin is also used to supply V_{PP} in Mode 0 for programming the EPROM, and supplies operating power to the EPROM during powerup operation.

E (ENABLE)

This is an output clock used primarily for bus synchronization. It is TTL compatible and is the slightly skewed divide-by-four result of the MCU input clock frequency. It will drive one Schottky TTL load and 90 pF, and all data given in cycles is referenced to this clock unless otherwise noted.

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

An $\overline{NMI}$ negative edge requests an MCU interrupt sequence, but the current instruction will be completed before it responds to the request. The MCU will then begin an interrupt sequence. Finally, a vector is fetched from $FFFC and $FFFD (or $BFFC and $BFFD in Mode 0), transferred to the Program Counter and instruction execution is resumed. $\overline{NMI}$ typically requires a 3.3 kΩ (nominal) resistor to V_{CC}. There is no internal $\overline{NMI}$ pullup resistor. $\overline{NMI}$ must be held low for at least one E-cycle to be recognized under all conditions.

$\overline{IRQ1}$ (MASKABLE INTERRUPT REQUEST 1)

$\overline{IRQ1}$ is a level-sensitive input which can be used to request an interrupt sequence. The MPU will complete the current instruction before it responds to the request. If the inter-

** Devices made with masks subsequent to T7A and C84 incorporate an advanced clock with improved startup characteritics.

FIGURE 17 — INTERRUPT FLOWCHART

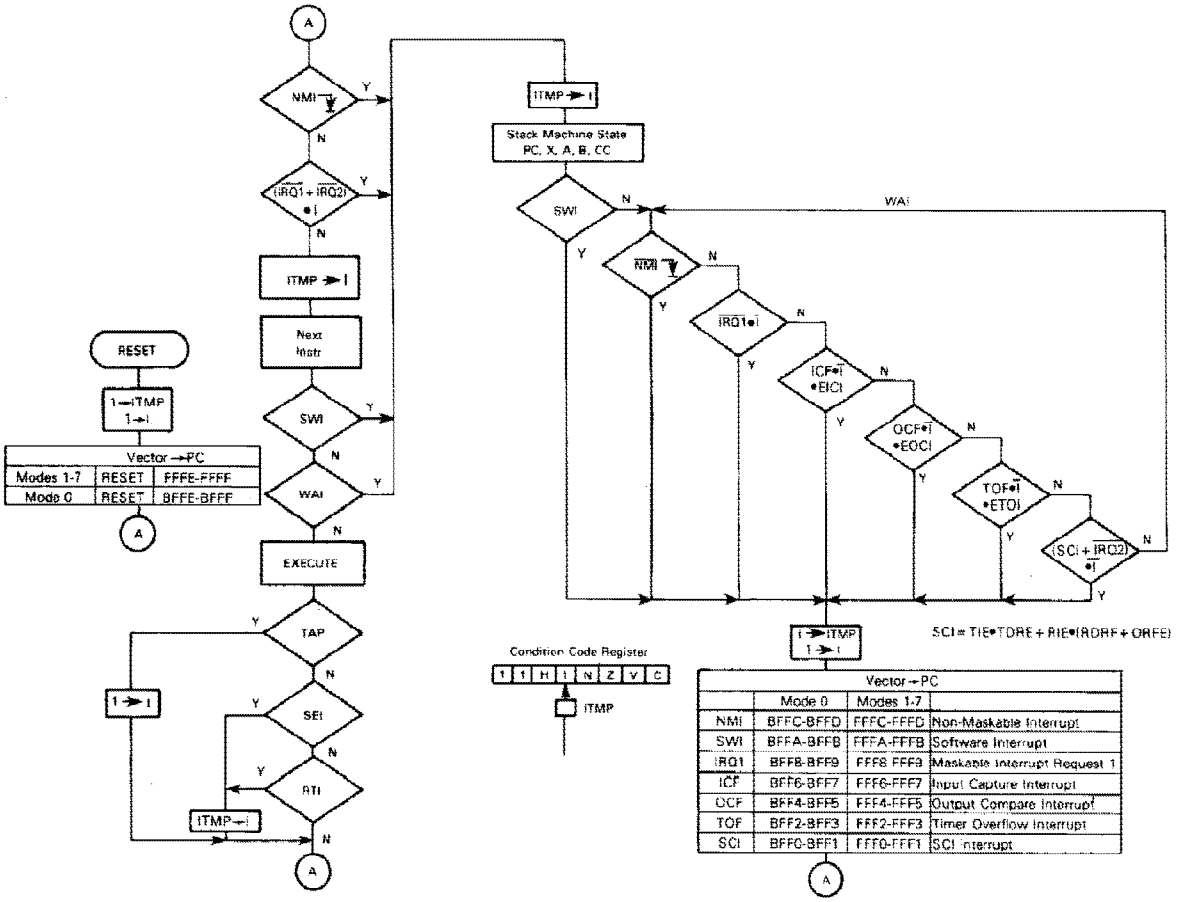


FIGURE 18 — INTERRUPT SEQUENCE

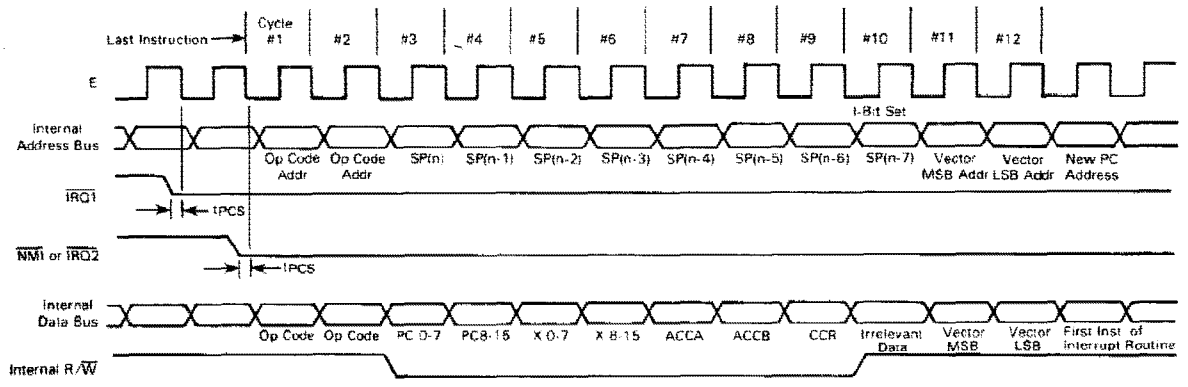
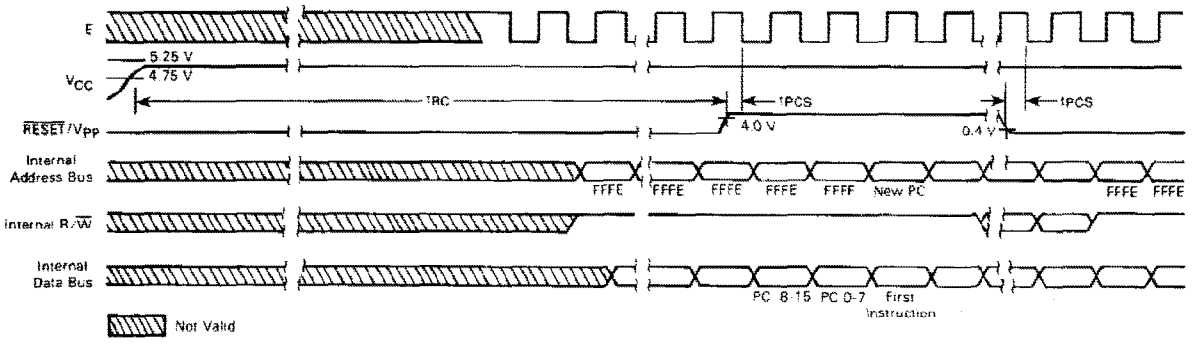


FIGURE 19 — RESET TIMING



rupt mask bit (I-bit) in the Condition Code Register is clear, the MCU will begin an interrupt sequence. A vector is fetched from $FFF8 and $FFF9 (or $BFF8 and $BFF9 in Mode 0), transferred to the Program Counter, and instruction execution is resumed.

$\overline{IRQ1}$ typically requires an external 3.3 kΩ (nominal) resistor to V_{CC} for wire-OR applications. $\overline{IRQ1}$ has no internal pullup resistor.

SC1 AND SC2 (STROBE CONTROL 1 AND 2)

The function of SC1 and SC2 depends on the operating mode. SC1 is configured as an output in all modes except single chip mode, whereas SC2 is always an output. SC1 and SC2 can drive one Schottky load and 90 pF.

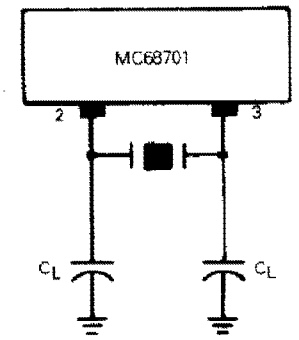
SC1 and SC2 In Single Chip Mode

In Single Chip Mode, SC1 and SC2 are configured as an input and output, respectively, and both function as Port 3 control lines. SC1 functions as $\overline{IS3}$ and can be used to indicate that Port 3 input data is ready or output data has been accepted. Three options associated with $\overline{IS3}$ are controlled by the Port 3 Control and Status Register and are discussed in the Port 3 description. If unused, $\overline{IS3}$ can remain unconnected.

SC2 is configured as $\overline{OS3}$ and can be used to strobe output data or acknowledge input data. It is controlled by Output Strobe Select (OSS) in the Port 3 Control and Status Register. The strobe is generated by a read (OSS = 0) or write (OSS = 1) to the Port 3 Data Register. $\overline{OS3}$ timing is shown in Figure 5.

FIGURE 20 — MC68701 OSCILLATOR CHARACTERISTICS

(a) Nominal Recommended Crystal Parameters



C_L = 20 pF (typical)

NOTE

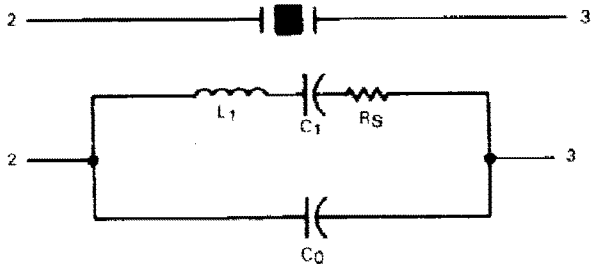
TTL-compatible oscillators may be obtained from:

Motorola Component Products
Attn: Data Clock Sales
2553 N. Edgington St.
Franklin Park, IL 60131
Tel: 312-451-1000
Telex: 433-0067

MC68701 Nominal Crystal Parameters

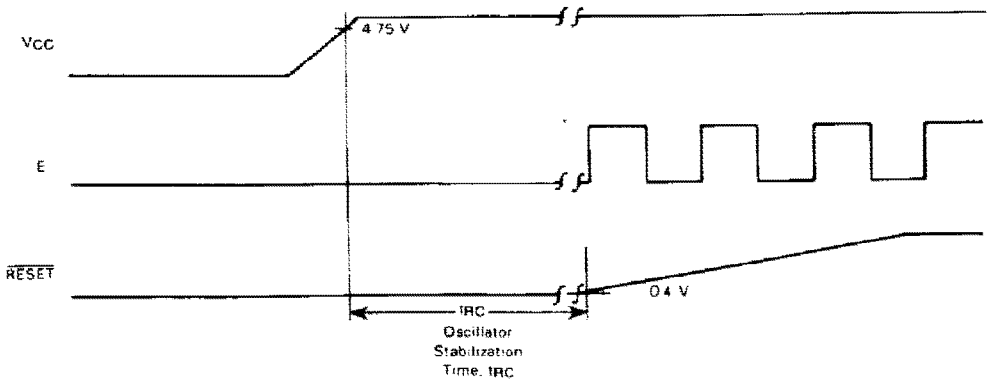
	3.58 MHz	4.00 MHz	5.0 MHz	6.0 MHz	8.0 MHz
RS	60 Ω	50 Ω	30-50 Ω	30-50 Ω	20-40 Ω
C_0	3.5 pF	6.5 pF	4.6 pF	4-6 pF	4.6 pF
C_1	0.015 pF	0.025 pF	0.01-0.02 pF	0.01-0.02 pF	0.01-0.02 pF
Q	>40 k	>30 k	>20 k	>20 k	>20 k

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



Equivalent Circuit

(b) Oscillator Stabilization Time (t_{RC})



SC1 And SC2 In Expanded Non-Multiplexed Mode

In the Expanded Non-Multiplexed Mode, both SC1 and SC2 are configured as outputs. SC1 functions as Input/Output Select ($\overline{IOS}$) and is asserted only when $0100 through $01FF is sensed on the internal address bus.

SC2 is configured as Read/Write and is used to control the direction of data bus transfers. An MPU read is enabled when Read/Write and E are high.

SC1 And SC2 In Expanded Multiplexed Mode

In the Expanded Multiplexed Modes, both SC1 and SC2 are configured as outputs. SC1 functions as Address Strobe and can be used to demultiplex the eight least significant addresses and the data bus. A latch controlled by Address Strobe captures address on the negative edge, as shown in Figure 15.

SC2 is configured as Read/Write and is used to control the direction of data bus transfers. An MPU read is enabled when Read/Write and E are high.

P10-P17 (PORT 1)

Port 1 is a mode independent 8-bit I/O port with each line an input or output as defined by the Port 1 Data Direction Register. The TTL compatible three-state output buffers can drive one Schottky TTL load and 30 pF, Darlington transistors, or CMOS devices using external pullup resistors. It is configured as a data input port by $\overline{RESET}$. Unused lines can remain unconnected.

P20-P24 (PORT 2)

Port 2 is a mode-independent, 5-bit, multipurpose I/O port. The voltage levels present on P20, P21, and P22 on the rising edge of $\overline{RESET}$ determine the operating mode of the MCU. The entire port is then configured as a data input port. The Port 2 lines can be selectively configured as data output lines by setting the appropriate bits in the Port 2 Data Direction Register. The Port 2 Data Register is used to move data through the port. However, if P21 is configured as an output, it will be tied to the timer Output Compare function and cannot be used to provide output from the Port 2 Data Register.

Port 2 can also be used to provide an interface for the Serial Communications Interface and the timer Input Edge function. These configurations are described in the appropriate SCI and Timer sections of this publication.

The Port 2 high-impedance, TTL compatible output buffers are capable of driving one Schottky TTL load and 30 pF or CMOS devices using external pullup resistors.

PORT 2 DATA REGISTER

7	6	5	4	3	2	1	0	
PC2	PC1	PC0	P24	P23	P22	P21	P20	$0003

P30-P37 (PORT 3)

Port 3 can be configured as an I/O port, a bidirectional 8-bit data bus, or a multiplexed address/data bus depending on the operating mode. The TTL compatible three-state output buffers can drive one Schottky TTL load and 90 pF. Unused lines can remain unconnected.

Port 3 In Single-Chip Mode

Port 3 is an 8-bit I/O port in the Single-Chip Mode, with each line configured by the Port 3 Data Direction Register. There are also two lines, $\overline{IS3}$ and $\overline{OS3}$, which can be used to control Port 3 data transfers.

Three Port 3 options are controlled by the Port 3 Control and Status Register and are available only in Single-Chip Mode: (1) Port 3 input data can be latched using IS3 as a control signal, (2) $\overline{OS3}$ can be generated by either an MPU read or write to the Port 3 Data Register, and (3) an $\overline{IRQ1}$ interrupt can be enabled by an $\overline{IS3}$ negative edge. Port 3 latch timing is shown in Figure 4.

PORT 3 CONTROL AND STATUS REGISTER

7	6	5	4	3	2	1	0	
IS3 Flag	IS3 IRQ1 Enable	X	OSS	Latch Enable	X	X	X	$000F

Bit 0-2	Not used.
Bit 3	LATCH ENABLE. This bit controls the input latch for Port 3. If set, input data is latched by an $\overline{IS3}$ negative edge. The latch is transparent after a read of Port 3 Data Register. LATCH ENABLE is cleared during reset.
Bit 4	OSS (Output Strobe Select). This bit determines whether $\overline{OS3}$ will be generated by a read or write of the Port 3 Data Register. When clear, the strobe is generated by a read; when set, it is generated by a write. OSS is cleared during reset.
Bit 5	Not used.
Bit 6	IS3 $\overline{IRQ1}$ ENABLE. When set, an $\overline{IRQ1}$ interrupt will be enabled whenever IS3 FLAG is set; when clear, the interrupt is inhibited. This bit is cleared during reset.
Bit 7	IS3 FLAG. This read-only status bit is set by an IS3 negative edge. It is cleared by a read of the Port 3 Control and Status Register (with IS3 FLAG set) followed by a read or write to the Port 3 Data Register or during reset.

Port 3 In Expanded Non-Multiplexed Mode

Port 3 is configured as a bidirectional data bus (D7-D0) in the Expanded Non-Multiplexed Mode. The direction of data transfers is controlled by Read/Write (SC2). Data is clocked by E (Enable).

Port 3 In Expanded Multiplexed Mode

Port 3 is configured as a time multiplexed address (A0-A7) and data bus (D7-D0) in the Expanded Multiplexed Modes where Address Strobe (AS) can be used to demultiplex the two buses. Port 3 is held in a high impedance state between valid address and data to prevent potentional bus conflicts.

P40-P47 (PORT 4)

Port 4 is configured as an 8-bit I/O port, as address outputs, or as data inputs depending on the operating mode. Port 4 can drive one Schottky TTL load and 90 pF and is the only port with internal pullup resistors. Unused lines can remain unconnected.

Port 4 In Single Chip Mode

In Single Chip Mode, Port 4 functions as an 8-bit I/O port with each line configured by the Port 4 Data Direction Register. Internal pullup resistors allow the port to directly interface with CMOS at 5 volt levels. External pullup resistors to more than 5 volts, however, cannot be used.

Port 4 In Expanded Non-Multiplexed Mode

Port 4 is configured during reset as an 8-bit input port, where the Port 4 Data Direction Register can be written to provide any or all of eight address lines A0 to A7. Internal pullup resistors pull the lines high until the Port 4 Data Direction Register is configured.

Port 4 In Expanded Multiplexed Mode

In all Expanded Multiplexed modes except Mode 6, Port 4 functions as half of the address bus and provides A8 to A15. In Mode 6, the port is configured during reset as an 8-bit parallel input port, where the Port 4 Data Direction Register can be written to provide any or all of upper address lines A8 to A15. Internal pullup resistors pull the lines high until the Port 4 Data Direction Register is configured, where bit 0 controls A8.

RESIDENT MEMORY

The MC68701 has 128 bytes of onboard RAM and 2048 bytes of onboard UV erasable EPROM. This memory is controlled by four bits in the RAM/EPROM Control Register.

One half of the RAM is powered through the V_{CC} standby pin and is maintainable during V_{CC} powerdown. This standby portion of the RAM consists of 64 bytes located from $80 through $BF.

Power must be supplied to V_{CC} standby if the internal RAM is to be used, regardless of whether standby power operation is anticipated. In Mode 3, V_{CC} standby should be tied to ground.

The RAM is controlled by the *RAM/EPROM Control* Register.

RAM/EPROM CONTROL REGISTER ($14)

The RAM/EPROM Control Register includes four bits: STBY PWR, RAME, PPC, and PLC. Two of these bits, STBY PWR and RAME, are used to control RAM access and determine the adequacy of the standby power source during power-down operation. It is intended that RAME be cleared and STBY PWR be set as part of a power-down procedure. RAME and STBY PWR are Read/Write bits.

The remaining two bits, PLC and PPC, control the operation of the EPROM. PLC and PPC are readable in all modes but can be changed only in Mode 0. The PLC bit can be written without restriction in Mode 0, but operation of the PPC bit is controlled by the state of PLC.

Associated with the EPROM are an 8-bit data latch and a 16-bit address latch. The data latch is enabled at all times, latching each data byte written to the EPROM. The address latch is controlled by the PLC bit.

A description of the RAM/EPROM Control Register follows.

MC68701 RAM/EPROM CONTROL REGISTER

7	6	5	4	3	2	1	0	
STBY PWR	RAME	X	X	X	X	PPC	PLC	$14

Bit 0 — PLC. Programming Latch Control. This bit controls (a) a latch which captures the EPROM address to be programmed and (b) whether the PPC bit can be cleared. The latch is triggered by an MPU write to a location in the EPROM. This bit is set during reset and can be cleared only in Mode 0. The PLC bit is defined as follows:

PLC = 0 EPROM address latch enabled; EPROM address is latched during MPU writes to the EPROM.

PLC = 1 EPROM address latch is transparent.

Bit 1 — PPC. Programming Power Control. This bit gates power from the $\overline{RESET}$/V_{PP} pin to the EPROM programming circuit. PPC is set during reset and whenever the PLC bit is set. It can be cleared only if (a) operating in Mode 0, and (b) if PLC has been previously cleared. The PPC bit is defined as follows:

PPC = 0 EPROM programming power (V_{PP}) applied.

PPC = 1 EPROM programming power (V_{PP}) is not applied.

Bit 2-5 — Unused.

Bit 6 RAME — RAM Enable. This Read/Write bit can be used to remove the entire RAM from the internal memory map. RAME is set (enabled) during reset provided standby power is available on the positive edge of reset. If RAME is clear, any access to a RAM address is external. If RAME is set and not in Mode 3, the RAM is included in the internal map.

Bit 7 STBY PWR — Standby Power. This bit is a read/write status bit which, when once set, remains set as long as V_{CC} standby remains above V_{SBB} (minimum). As long as this bit is set following a period of standby operation, the standby power supply has adequately preserved the data in the standby RAM. If this bit is cleared during a period of standby operation, it indicates that V_{CC} standby had fallen to a level sufficiently below V_{SBB} (minimum) to suspect that data in the standby RAM is not valid. This bit can be set only by software and is not affected during reset.

Note that if PPC and PLC are set, they cannot be simultaneously cleared with a single MPU write. The PLC bit must be cleared prior to attempting to clear PPC. If both PPC and PLC are clear, setting PLC will also set PPC. In addition,

it is assumed that V_{PP} is applied to the RESET/V_{PP} pin whenever PPC is clear. If this is not the case, the result is undefined.

ERASING THE MC68701 EPROM

Ultraviolet erasure will clear all bits of the EPROM to the "0" state. Note that this erased state differs from that of some other widely used EPROMs (such as the MCM68708) where the erased state is a "1". The MC68701 EPROM is programmed by erasing it to "0's" and entering "1's" into the desired bit locations.

The MC68701 EPROM can be erased by exposure to high intensity ultraviolet light with a wave length of 2537A for a minimum of 30 minutes. The recommended integrated dose (UV intensity X exposure time) is 15 Ws/cm. The lamps should be used without shortwave filters and the MC68701 should be positioned about one inch away from the UV tubes.

The MC68701 transparent lid should always be covered after erasing. This protects both the EPROM and light-sensitive nodes from accidental exposure to ultraviolet light.

PROGRAMMING THE MC68701 EPROM

When the MC68701 is released from Reset in Mode 0, a vector is fetched from location $BFFE:BFFF. This provides a method for an external program to obtain control of the microcomputer with access to every location in the EPROM.

To program the EPROM, it is necessary to operate the MC68701 in Mode 0 under the control of a program resident in external memory which can facilitate loading and programming of the EPROM. After the pattern has been loaded into external memory, the EPROM can be programmed as follows:

a. Apply programming power (V_{PP}) to the $\overline{\text{RESET}}$/V_{PP} pin.

b. Clear the PLC control bit and set the PPC bit by writing $FE to the RAM/EPROM Control Register.

c. Write data to the next EPROM location to be programmed. Triggered by an MPU write to the EPROM, internal latches capture both the EPROM address and the data byte.

d. Clear the PPC bit for programming time, t_{pp}, by writing $FC to the RAM/EPROM Control Register and waiting for time, t_{pp}. This step gates the programming power (V_{PP}) from the $\overline{\text{RESET}}$/V_{PP} pin to the EPROM which programs the location.

e. Repeat steps b through d for each byte to be programmed.

f. Set the PLC and PPC bits by writing $FF to the RAM/EPROM control register.

g. Remove the programming power (V_{PP}) from the $\overline{\text{RESET}}$/V_{PP} pin. The EPROM can now be read and verified.

Because of the erased state of an EPROM byte is $00, it is not necessary to program a location which is to contain $00. Finally, it should be noted that the result of inadvertently programming a location more than once is the logical OR of the data patterns.

A routine which can be used to program the MC68701 EPROM is provided at the end of this publication. This non-reentrant routine requires four double byte variables named IMBEQ, IMEND, PNTR, and WAIT to be initialized prior to entry to the routine. These variables indicate (a) the first and last memory locations which bound the data to be programmed into the EPROM, (b) the first EPROM location to be programmed, and (c) a number which is used to generate the programming time delay. The last variable, WAIT, takes into account the MCU input crystal (or TTL-compatible clock) frequency to insure the programming time, t_{pp}, is met. WAIT is defined as the number of MPU E-cycles that will occur in the real-time EPROM programming interval, t_{pp}. For example, if $t_{pp} = 50$ milliseconds and the MC68701 is being driven with a 4.00 MHz TTL-compatible clock:

$$\text{WAIT (MPU E-cycles)} = t_{pp} * (\text{MCU INPUT FREQ})/4 * 10^6$$
$$= 50000(4 * 10^6)/4 * 10^6$$
$$= 50000$$

NOTE

A monitor program called PRObug® is available from Motorola Microsystems. PRObug contains a user option for programming the on-board MC68701 EPROM.

PROGRAMMABLE TIMER

The Programmable Timer can be used to perform input waveform measurements while independently generating an output waveform. Pulse widths can vary from several microseconds to many seconds. A block diagram of the Timer is shown in Figure 21.

COUNTER ($09:0A)

The key timer element is a 16-bit free-running counter which is incremented by E (Enable). It is cleared during reset and is read-only with one exception: a write to the counter ($09) will preset it to $FFF8. This feature, intended for testing, can disturb serial operations because the counter provides the SCI internal bit rate clock. TOF is set whenever the counter contains all 1's.

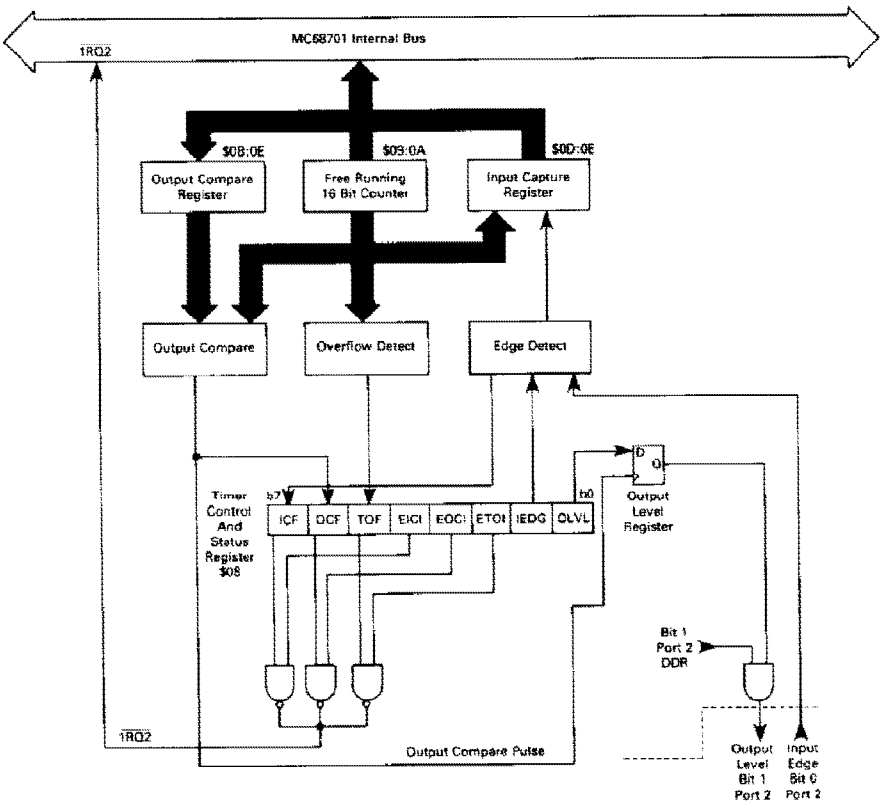
OUTPUT COMPARE REGISTER ($0B:0C)

The Output Compare Register is a 16-bit Read/Write register used to control an output waveform or provide an arbitrary timeout flag. It is compared with the free-running counter on each E-cycle. When a match occurs, OCF is set and OLVL is clocked to an output level register. If Port 2, bit 1, is configured as an output, OLVL will appear at P21 and the Output Compare Register and OLVL can then be changed for the next compare. The function is inhibited for one cycle after a write to the high byte of the Compare Register ($0B) to ensure a valid compare. The Output Compare Register is set to $FFFF during reset.

INPUT CAPTURE REGISTER ($0D:0E)

The Input Capture Register is a 16-bit read-only register used to store the free-running counter when a "proper" input transition occurs as defined by IEDG. Port 2, bit 0 should be configured as an input, but the edge detect circuit always

FIGURE 21 — BLOCK DIAGRAM OF PROGRAMMABLE TIMER



senses P20 even when configured as an output. An input capture can occur independently of ICF: the register always contains the most current value. Counter transfer is inhibited, however, between accesses of a double byte MPU read. The input pulse width must be at least two E-cycles to ensure an input capture under all conditions.

TIMER CONTROL AND STATUS REGISTER ($08)

The Timer Control and Status Register (TCSR) is an 8-bit register of which all bits are readable while bits 0-4 can be written. The three most significant bits provide the timer status and indicate if:

- a proper level transition has been detected,
- a match has occurred between the free-running counter and the output compare register, and
- the free-running counter has overflowed.

Each of the three events can generate an $\overline{IRQ2}$ interrupt and is controlled by an individual enable bit in the TCSR.

TIMER CONTROL AND STATUS REGISTER (TCSR)

7	6	5	4	3	2	1	0	
ICF	OCF	TOF	EICI	EOCI	ETOI	IEDG	OLVL	$0008

Bit 0 OLVL	Output level. OLVL is clocked to the output level register by a successful output compare and will appear at P21 if Bit 1 of the Port 2 Data Direction Register is set. It is cleared during reset.
Bit 1 EIDG	Input Edge. IEDG is cleared during reset and controls which level transition will trigger a counter transfer to the Input Capture Register: IEDG = 0 Transfer on a negative-edge IEDG = 1 Transfer on a positive-edge.
Bit 2 ETOI	Enable Timer Overflow Interrupt. When set, an $\overline{IRQ2}$ interrupt is enabled for a timer overflow; when clear, the interrupt is inhibited. It is cleared during reset.
Bit 3 EOCI	Enable Output Compare Interrupt. When set, an $\overline{IRQ2}$ interrupt is enabled for an output compare; when clear, the interrupt is inhibited. It is cleared during reset.
Bit 4 EICI	Enable Input Capture Interrupt. When set, an $\overline{IRQ2}$ interrupt is enabled for an input capture; when clear, the interrupt is inhibited. It is cleared during reset.

Bit 5 TOF — Timer Overflow Flag. TOF is set when the counter contains all 1's. It is cleared by reading the TCSR (with TOF set) then reading the counter high byte ($09), or by RESET.

Bit 6 OCF — Output Compare Flag. OCF is set when the Output Compare Register matches the free-running counter. It is cleared by reading the TCSR (with OCF set) and then writing to the Output Compare Register ($0B or $0C), or by RESET.

Bit 7 ICF — Input Capture Flag. ICF is set to indicate a proper level transition; it is cleared by reading the TCSR (with ICF set) and then the Input Capture Register High Byte ($0D), or by RESET.

SERIAL COMMUNICATIONS INTERFACE (SCI)

A full-duplex asynchronous Serial Communications Interface (SCI) is provided with two data formats and a variety of rates. The SCI transmitter and receiver are functionally independent, but use the same data format and bit rate. Serial data formats include standard mark/space (NRZ) and Bi-phase and both provide one start bit, eight data bits, and one stop bit. "Baud" and "bit rate" are used synonymously in the following description.

WAKE-UP FEATURE

In a typical serial loop multi-processor configuration, the software protocol will usually identify the addressee(s) at the beginning of the message. In order to permit uninterested MPU's to ignore the remainder of the message, a wake-up feature is included whereby all further SCI receiver flag (and interrupt) processing can be inhibited until the data line goes idle. An SCI receiver is re-enabled by an idle string of ten consecutive 1's or during reset. Software must provide for the required idle string between consecutive messages and prevent it within messages.

PROGRAMMABLE OPTIONS

The following features of the SCI are programmable:

- format: standard mark/space (NRZ) or Bi-phase
- clock: external or internal bit rate clock
- Baud : one of 4 per E-clock frequency, or external clock (X8 desired baud)
- wake-up feature: enabled or disabled
- interrupt requests: enabled individually for transmitter and receiver
- clock output: internal bit rate clock enabled or disabled to P22

SERIAL COMMUNICATIONS REGISTERS

The Serial Communications Interface includes four addressable registers as depicted in Figure 22. It is controlled by the Rate and Mode Control Register and the Transmit/Receive Control and Status Register. Data is transmitted and received utilizing a write-only Transmit Register and a read-only Receive Register. The shift registers are not accessible to software.

Rate and Mode Control Register (RMCR) ($10)

The Rate and Mode Control Register controls the SCI bit rate, format, clock source, and under certain conditions, the configuration of P22. The register consists of four write-only bits which are cleared during reset. The two least significant bits control the bit rate of the internal clock and the remaining two bits control the format and clock source.

RATE AND MODE CONTROL REGISTER (RMCR)

7	6	5	4	3	2	1	0	
X	X	X	X	CC1	CC0	SS1	SS0	$0010

Bit 1:Bit 0 — SS1:SS0 Speed Select. These two bits select the Baud rate when using the internal clock. Four rates may be selected which are a function of the MCU input frequency. Table 6 lists bit time and rates for three selected MCU frequencies.

Bit 3:Bit 2 — CC1:CC0 Clock Control and Format Select. These two bits control the format and select the serial clock source. If CC1 is set, the DDR value for P22 is forced to the complement of CC0 and cannot be altered until CC1 is cleared. If CC1 is cleared after having been set, its DDR value is unchanged. Table 7 defines the formats, clock source, and use of P22.

If both CC1 and CC0 are set, an external TTL compatible clock must be connected to P22 at eight times (8X) the desired bit rate, but not greater than E, with a duty cycle of 50% (±10%). If CC1:CC0 = 10, the internal bit rate clock is provided at P22 regardless of the values for TE or RE.

NOTE: The source of SCI internal bit rate clock is the timer free running counter. An MPU write to the counter can disturb serial operations.

Transmit/Receive Control And Status Register (TRCSR) ($11)

The Transmit/Receive Control and Status Register controls the transmitter, receiver, wake-up feature, and two individual interrupts and monitors the status of serial operations. All eight bits are readable while bits 0 to 4 are also writable. The register is initialized to $20 by RESET.

TRANSMIT/RECEIVE CONTROL AND STATUS REGISTER (TRCSR)

7	6	5	4	3	2	1	0	
RDRF	ORFE	TDRE	RIE	RE	TIE	TE	WU	$0011

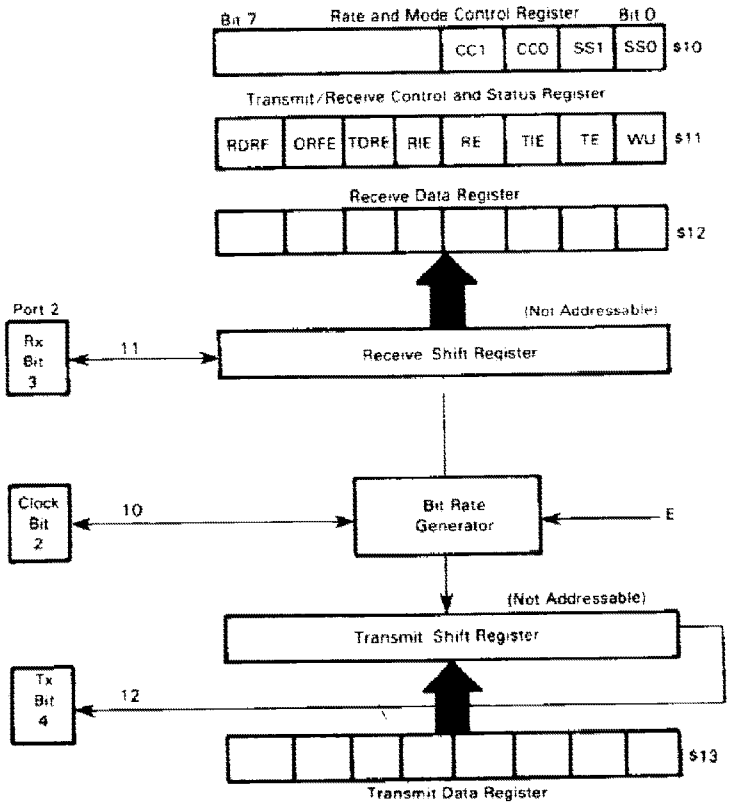
TABLE 6 — SCI BIT TIMES AND RATES

SS1:SS0		$4f_o$ —	2.4576 MHz	4.0 MHz	4.9152 MHz
		E	614.4 kHz	1.0 MHz	1.2288 MHz
0	0	÷16	26 μs/38,400 Baud	16 μs/62,500 Baud	13.0 μs/76,800 Baud
0	1	÷128	208 μs/4,800 Baud	128 μs/7812.5 Baud	104.2 μs/9,600 Baud
1	0	÷1024	1.67 ms/600 Baud	1.024 ms/976.6 Baud	833.3 μs/1,200 Baud
1	1	÷4096	6.67 ms/150 Baud	4.096 ms/244.1 Baud	3.33 ms/300 Baud
External (P22)			Up to 76,800 Baud	Up to 125,000 Baud	Up to 153,600 Baud

TABLE 7 — SCI FORMAT AND CLOCK SOURCE CONTROL

CC1:CC0		Format	Clock Source	Port 2, Bit 2
0	0	Bi-Phase	Internal	Not Used
0	1	NRZ	Internal	Not Used
1	0	NRZ	Internal	Output
1	1	NRZ	External	Input

FIGURE 22 — SCI REGISTERS



Bit 0 WU — "Wake-up" on Idle Line. When set, WU enables the wake-up function; it is cleared by ten consecutive 1's or during reset. WU will not set if the line is idle.

Bit 1 TE — Transmit Enable. When set, P24 DDR bit is set, cannot be changed, and will remain set if TE is subsequently cleared. When TE is changed from clear to set, the transmitter is connected to P24 and a preamble of nine consecutive 1's is transmitted. TE is cleared during reset.

Bit 2 TIE — Transmit Interrupt Enable. When set, an $\overline{IRQ2}$ interrupt is enabled when TDRE is set; when clear, the interrupt is inhibited. TE is cleared during reset.

Bit 3 RE — Receive Enable. When set, the P23 DDR bit is cleared, cannot be changed, and will remain clear if RE is subsequently cleared. While RE is set, the SCI receiver is enabled. RE is cleared during reset.

Bit 4 RIE — Receiver Interrupt Enable. When set, an IRQ2 interrupt is enabled when RDRF and/or ORFE is set; when clear, the interrupt is inhibited. RIE is cleared during reset.

Bit 5 TDRE — Transmit Data Register Empty. TDRE is set when the Transmit Data Register is transferred to the output serial shift register or during reset. It is cleared by reading the TRCSR (with TDRE set) and then writing to the Transmit Data Register. Additional data will be transmitted only if TDRE has been cleared.

Bit 6 ORFE — Overrun Framing Error. If set, ORFE indicates either an overrun or framing error. An overrun is a new byte ready to transfer to the Receiver Data Register with RDRF still set. A receiver framing error has occurred when the byte boundaries of the bit stream are not synchronized to the bit counter. An overrun can be distinguished from a framing error by the state of RDRF: if RDRF is set, then an overrun has occurred; otherwise a framing error has been detected. Data is not transferred to the Receive Data Register in an overrun condition. Unframed data causing a framed error is transferred to the Receive Data Register. However, subsequent data transfer is blocked until the framing error flag is cleared.* ORFE is cleared by reading the TRCSR (with ORFE set) then the Receive Data Register, or during reset.

Bit 7 RDRF — Receive Data Register Full. RDRF is set when the input serial shift register is transferred to the Receive Data Register. It is cleared by reading the TRCSR (with RDRF set), and then the Receive Data Register, or during reset.

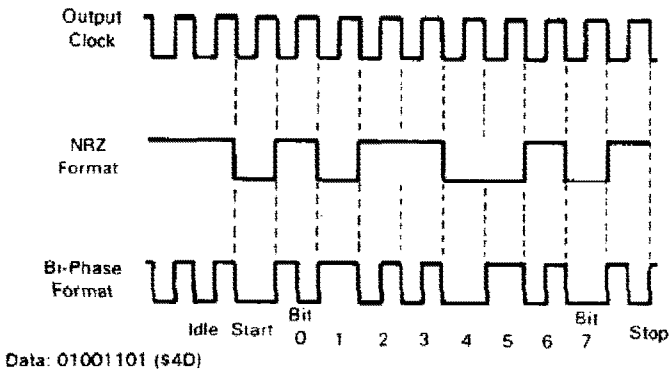
SERIAL OPERATIONS

The SCI is initialized by writing control bytes first to the Rate and Mode Control Register and then to the Transmit/Receive Control and Status Register. When TE is set, the output of the transmit serial shift register is connected to P24 and serial output is initiated by transmitting to 9-bit preamble of 1's.

At this point one of two situations exist: 1) if the Transmit Data Register is empty (TDRE = 1), a continuous string of 1's will be sent indicating an idle line, or 2) if a byte has been written to the Transmit-Data Register (TDRE = 0), it will be transferred to the output serial shift register (synchronized with the bit rate clock), TDRE will be set, and transmission will begin.

The start bit (0), eight data bits (beginning with bit 0) and a stop bit (1), will be transmitted. If TDRE is still set when the next byte transfer should occur, 1's will be sent until more data is provided. In Bi-phase format, the output toggles at the start of each bit and at half-bit time when a "1" is sent. Receive operation is controlled by RE which configures P23 as an input and enables the receiver. SCI data formats are illustrated in Figure 23.

FIGURE 23 — SCI DATA FORMATS



* Devices made with mask numbers T7A and CB4 do not transfer unframed data to the Receive Data Register.

INSTRUCTION SET

The MC68701 is upward source and object code compatible with the MC6800. Execution times of key instructions have been reduced and several new instructions have been added, including a hardware multiply. A list of new operations added to the MC6800 instruction set is shown in Table 1. In addition, two new special opcodes, 4E and 5E, are provided for test purposes. These opcodes force the program counter to increment like a 16-bit counter, causing address lines used in the expanded modes to increment until the device is reset. These opcodes have no mnemonics.

The coding of the first (or only) byte corresponding to an executable instruction is sufficient to identify the instruction and the addressing mode. The hexadecimal equivalents of the binary codes, which result from the translation of the 82 instructions in all valid modes of addressing, are shown in Table 8. There are 220 valid machine codes, 34 unassigned codes, and 2 reserved for test purposes.

PROGRAMMING MODEL

A programming model for the MC68701 is shown in Figure 9. Accumulator A can be concatenated with accumulator B and jointly referred to as accumulator D where A is the most significant byte. Any operation which modifies the double accumulator will also modify accumulator A and/or B. Other registers are defined as follows:

Program Counter — The program counter is a 16-bit register which always points to the next instruction.

Stack Pointer — The stack pointer is a 16-bit register which contains the address of the next available location in a pushdown/pullup (LIFO) queue. The stack resides in random access memory at a location defined by the programmer.

Index Register — The Index Register is a 16-bit register which can be used to store data or provide an address for the indexed mode of addressing.

Accumulators — The MCU contains two 8-bit accumulators, A and B, which are used to store operands and results from the arithmetic logic unit (ALU). They can also be concatenated and referred to as the D (double) accumulator.

Condition Code Registers — The condition code register indicates the results of an instruction and includes the Overflow (V), Carry/Borrow from MSB (C), and Half Carry following five condition bits: Negative (N), Zero (Z), from bit 3 (H). These bits are testable by the conditional branch instructions. Bit 4 is the interrupt mask (I-bit) and inhibits all maskable interrupts when set. The two unused bits, B6 and B7 are read as ones.

ADDRESSING MODES

The MC68701 provides six addressing modes which can be used to reference memory. A summary of addressing modes for all instructions is presented in Tables 9, 10, 11, and 12 where execution times are provided in E cycles. Instruction execution times are summarized in Table 13. With an input frequency of 4 MHz, E cycles are equivalent to microseconds. A cycle-by-cycle description of bus activity for each instruction is provided in Table 14 and a description of selected instructions is shown in Figure 24.

Immediate Addressing — The operand or "immediate byte(s)" is contained in the following byte(s) of the instruction where the number of bytes matches the size of the register. These are two or three byte instructions.

Direct Addressing — The least significant byte of the operand address is contained in the second byte of the instruction and the most significant byte is assumed to be $00. Direct addressing allows the user to access $00 through $FF using two byte instructions and execution time is reduced by eliminating the additional memory access. In most applications, the 256-byte area is reserved for frequently referenced data.

Extended Addressing — The second and third bytes of the instruction contain the absolute address of the operand. These are three byte instrutions.

Indexed Addressing — The unsigned offset contained in the second byte of the instruction is added with carry to the Index Register and used to reference memory without changing the Index Register. These are two byte instructions.

Inherent Addressing — The operand(s) are registers and no memory reference is required. These are single byte instructions.

Relative Addressing — Relative addressing is used only for branch instructions. If the branch condition is true, the Program Counter is overwritten with the sum of a signed single byte displacement in the second byte of the instruction and the current Program Counter. This provides a branch range of −126 to 129 bytes from the first byte of the instruction. These are two byte instructions.

TABLE 8 — CPU INSTRUCTION MAP

OP	MNEM	MODE	~	#	OP	MNEM	MODE	~	#	OP	MNEM	MODE	~	#	OP	MNEM	MODE	~	#	OP	MNEM	MODE	~	#
00	*				34	DES	INHER	3	1	68	ASL	INDXD	6	2	9C	CPX	DIR	5	2	D0	SUBB	DIR	3	2
01	NOP	INHER	2	1	35	TXS		3	1	69	ROL		6	2	9D	JSR		5	2	D1	CMPB		3	2
02	*				36	PSHA		3	1	6A	DEC		6	2	9E	LDS		4	2	D2	SBCB		3	2
03	*				37	PSHB		3	1	6B	*				9F	STS	DIR	4	2	D3	ADDD		5	2
04	LSRD		3	1	38	PULX		5	1	6C	INC		6	2	A0	SUBA	INDXD	4	2	D4	ANDB		3	2
05	ASLD		3	1	39	RTS		5	1	6D	TST		6	2	A1	CMPA		4	2	D5	BITB		3	2
06	TAP		2	1	3A	ABX		3	1	6E	JMP		3	2	A2	SBCA		4	2	D6	LDAB		3	2
07	TPA		2	1	3B	RTI		10	1	6F	CLR	INDXD	6	2	A3	SUBD		6	2	D7	STAB		3	2
08	INX		3	1	3C	PSHX		4	1	70	NEG	EXTND	6	3	A4	ANDA		4	2	D8	EORB		3	2
09	DEX		3	1	3D	MUL		10	1	71	*				A5	BITA		4	2	D9	ADCB		3	2
0A	CLV		2	1	3E	WAI		9	1	72	*				A6	LDAA		4	2	DA	ORAB		3	2
0B	SEV		2	1	3F	SWI		12	1	73	COM		6	3	A7	STAA		4	2	DB	ADDB		3	2
0C	CLC		2	1	40	NEGA		2	1	74	LSR		6	3	A8	EORA		4	2	DC	LDD		4	2
0D	SEC		2	1	41	*				75	*				A9	ADCA		4	2	DD	STD		4	2
0E	CLI		2	1	42	*				76	ROR		6	3	AA	ORAA		4	2	DE	LDX		4	2
0F	SEI		2	1	43	COMA		2	1	77	ASR		6	3	AB	ADDA		4	2	DF	STX	DIR	4	2
10	SBA		2	1	44	LSRA		2	1	78	ASL		6	3	AC	CPX		6	2	E0	SUBB	INDXD	4	2
11	CBA		2	1	45	*				79	ROL		6	3	AD	JSR		6	2	E1	CMPB		4	2
12	*				46	RORA		2	1	7A	DEC		6	3	AE	LDS		5	2	E2	SBCB		4	2
13	*				47	ASRA		2	1	7B	*				AF	STS	INDXD	5	2	E3	ADDD		6	2
14	*				48	ASLA		2	1	7C	INC		6	3	B0	SUBA	EXTND	4	3	E4	ANDB		4	2
15	*				49	ROLA		2	1	7D	TST		6	3	B1	CMPA		4	3	E5	BITB		4	2
16	TAB		2	1	4A	DECA		2	1	7E	JMP		3	3	B2	SBCA		4	3	E6	LDAB		4	2
17	TBA		2	1	4B	*				7F	CLR	EXTND	6	3	B3	SUBD		6	3	E7	STAB		4	2
18	*				4C	INCA		2	1	80	SUBA	IMMED	2	2	B4	ANDA		4	3	E8	EORB		4	2
19	DAA	INHER	2	1	4D	TSTA		2	1	81	CMPA		2	2	B5	BITA		4	3	E9	ADCB		4	2
1A	*				4E	T				82	SBCA		2	2	B6	LDAA		4	3	EA	ORAB		4	2
1B	ABA	INHER	2	1	4F	CLRA		2	1	83	SUBD		4	3	B7	STAA		4	3	EB	ADDB		4	2
1C	*				50	NEGB		2	1	84	ANDA		2	2	B8	EORA		4	3	EC	LDD		5	2
1D	*				51	*				85	BITA		2	2	B9	ADCA		4	3	ED	STD		5	2
1E	*				52	*				86	LDAA		2	2	BA	ORAA		4	3	EE	LDX		5	2
1F	*				53	COMB		2	1	87	*				BB	ADDA		4	3	EF	STX	INDXD	5	2
20	BRA	REL	3	2	54	LSRB		2	1	88	EORA		2	2	BC	CPX		6	3	F0	SUBB	EXTND	4	3
21	BRN		3	2	55	*				89	ADCA		2	2	BD	JSR		6	3	F1	CMPB		4	3
22	BHI		3	2	56	RORB		2	1	8A	ORAA		2	2	BE	LDS		5	3	F2	SBCB		4	3
23	BLS		3	2	57	ASRB		2	1	8B	ADDA		2	2	BF	STS	EXTND	5	3	F3	ADDD		6	3
24	BCC		3	2	58	ASLB		2	1	8C	CPX	IMMED	4	3	C0	SUBB	IMMED	2	2	F4	ANDB		4	3
25	BCS		3	2	59	ROLB		2	1	8D	BSR	REL	6	2	C1	CMPB		2	2	F5	BITB		4	3
26	BNE		3	2	5A	DECB		2	1	8E	LDS	IMMED	3	3	C2	SBCB		2	2	F6	LDAB		4	3
27	BEQ		3	2	5B	*				8F	*				C3	ADDD		4	3	F7	STAB		4	3
28	BVC		3	2	5C	INCB		2	1	90	SUBA	DIR	3	2	C4	ANDB		2	2	F8	EORB		4	3
29	BVS		3	2	5D	TSTB		2	1	91	CMPA		3	2	C5	BITB		2	2	F9	ADCB		4	3
2A	BPL		3	2	5E	T				92	SBCA		3	2	C6	LDAB		2	2	FA	ORAB		4	3
2B	BMI		3	2	5F	CLRB	INHER	2	1	93	SUBD		5	2	C7	*				FB	ADDB		4	3
2C	BGE		3	2	60	NEG	INDXD	6	2	94	ANDA		3	2	C8	EORB		2	2	FC	LDD		5	3
2D	BLT		3	2	61	*				95	BITA		3	2	C9	ADCB		2	2	FD	STD		5	3
2E	BGT		3	2	62	*				96	LDAA		3	2	CA	ORAB		2	2	FE	LDX		5	3
2F	BLE	REL	3	2	63	COM		6	2	97	STAA		3	2	CB	ADDB		2	2	FF	STX	EXTND	5	3
30	TSX	INHER	3	1	64	LSR		6	2	98	EORA		3	2	CC	LDD		3	3					
31	INS		3	1	65	*				99	ADCA		3	2	CD	*								
32	PULA		4	1	66	ROR		6	2	9A	ORAA		3	2	CE	LDX	IMMED	3	3					
33	PULB		4	1	67	ASR	INDXD	6	2	9B	ADDA		3	2	CF	*								

* UNDEFINED OP CODE

NOTES:

1. Addressing Modes
   INHER = Inherent INDXD = Indexed IMMED = Immediate
   REL = Relative EXTND = Extended DIR = Direct
2. Unassigned opcodes are indicated by "*" and should not be executed.
3. Codes marked by "T" force the PC to function as a 16-bit counter.

TABLE 9 — INDEX REGISTER AND STACK MANIPULATION INSTRUCTIONS

Pointer Operations	MNEM	Immed Op	Immed ~	Immed #	Direct Op	Direct ~	Direct #	Index Op	Index ~	Index #	Extend Op	Extend ~	Extend #	Inherent Op	Inherent ~	Inherent #	Boolean/ Arithmetic Operation	Condition Codes 5 H	4 I	3 N	2 Z	1 V	0 C
Compare Index Register	CPX	8C	4	3	9C	5	2	AC	6	2	BC	6	3				X − M:M + 1	•	•	↕	↕	↕	↕
Decrement Index Register	DEX													09	3	1	X − 1 → X	•	•	•	↕	•	•
Decrement Stack Pointer	DES													34	3	1	SP − 1 → SP	•	•	•	•	•	•
Increment Index Register	INX													08	3	1	X + 1 → X	•	•	•	↕	•	•
Increment Stack Pointer	INS													31	3	1	1 SP + 1 → SP	•	•	•	•	•	•
Load Index Register	LDX	CE	3	3	DE	4	2	EE	5	2	FE	5	3				M → X_H, (M + 1) → X_L	•	•	↕	↕	R	•
Load Stack Pointer	LDS	8E	3	3	9E	4	2	AE	5	2	BE	5	3				M → SP_H, (M + 1) → SP_L	•	•	↕	↕	R	•
Store Index Register	STX				DF	4	2	EF	5	2	FF	5	3				X_H → M, X_L → (M + 1)	•	•	↕	↕	R	•
Store Stack Pointer	STS				9F	4	2	AF	5	2	BF	5	3				SP_H → M, SP_L → (M + 1)	•	•	↕	↕	R	•
Index Reg → Stack Pointer	TXS													35	3	1	X − 1 → SP	•	•	•	•	•	•
Stack Pntr → Index Register	TSX													30	3	1	SP + 1 → X	•	•	•	•	•	•
Add	ABX													3A	3	1	B + X → X	•	•	•	•	•	•
Push Data	PSHX													3C	4	1	X_L → M_{SP}, SP − 1 → SP X_H → M_{SP}, SP − 1 → SP	•	•	•	•	•	•
Pull Data	PULX													38	5	1	SP + 1 → SP, M_{SP} → X_H SP + 1 → SP, M_{SP} → X_L	•	•	•	•	•	•

TABLE 10 — ACCUMULATOR AND MEMORY INSTRUCTIONS (Sheet 1 of 2)

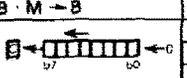
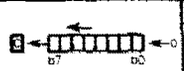
Accumulator and Memory Operations	MNE	Immed Op	Immed ~	Immed #	Direct Op	Direct ~	Direct #	Index Op	Index ~	Index #	Extend Op	Extend ~	Extend #	Inher Op	Inher ~	Inher #	Boolean Expression	Condition Codes H	I	N	Z	V	C
Add Acmltrs	ABA													1B	2	1	A + B → A	↕	•	↕	↕	↕	↕
Add B to X	ABX													3A	3	1	00:B + X → X	•	•	•	•	•	•
Add with Carry	ADCA	89	2	2	99	3	2	A9	4	2	B9	4	3				A + M + C → A	↕	•	↕	↕	↕	↕
	ADCB	C9	2	2	D9	3	2	E9	4	2	F9	4	3				B + M + C → B	↕	•	↕	↕	↕	↕
Add	ADDA	8B	2	2	9B	3	2	AB	4	2	BB	4	3				A + M → A	↕	•	↕	↕	↕	↕
	ADDB	CB	2	2	DB	3	2	EB	4	2	FB	4	3				B + M → A	↕	•	↕	↕	↕	↕
Add Double	ADDD	C3	4	3	D3	5	2	E3	6	2	F3	6	3				D + M:M + 1 → D	•	•	↕	↕	↕	↕
And	ANDA	84	2	2	94	3	2	A4	4	2	B4	4	3				A · M → A	•	•	↕	↕	R	•
	ANDB	C4	2	2	D4	3	2	E4	4	2	F4	4	3				B · M → B	•	•	↕	↕	R	•
Shift Left, Arithmetic	ASL							68	6	2	78	6	3					•	•	↕	↕	↕	↕
	ASLA													48	2	1		•	•	↕	↕	↕	↕
	ASLB													58	2	1		•	•	↕	↕	↕	↕
Shift Left Dbl	ASLD													05	3	1		•	•	↕	↕	↕	↕
Shift Right, Arithmetic	ASR							67	6	2	77	6	3					•	•	↕	↕	↕	↕
	ASRA													47	2	1		•	•	↕	↕	↕	↕
	ASRB													57	2	1		•	•	↕	↕	↕	↕
Bit Test	BITA	85	2	2	95	3	2	A5	4	2	B5	4	3				A · M	•	•	↕	↕	R	•
	BITB	C5	2	2	D5	3	2	E5	4	2	F5	4	3				B · M	•	•	↕	↕	R	•
Compare Acmltrs	CBA													11	2	1	A − B	•	•	↕	↕	↕	↕
Clear	CLR							6F	6	2	7F	6	3				00 → M	•	•	R	S	R	R
	CLRA													4F	2	1	00 → A	•	•	R	S	R	R
	CLRB													5F	2	1	00 → B	•	•	R	S	R	R
Compare	CMPA	81	2	2	91	3	2	A1	4	2	B1	4	3				A − M	•	•	↕	↕	↕	↕
	CMPB	C1	2	2	D1	3	2	E1	4	2	F1	4	3				B − M	•	•	↕	↕	↕	↕
1's Complement	COM							63	6	2	73	6	3				$\overline{M}$ → M	•	•	↕	↕	R	S
	COMA													43	2	1	$\overline{A}$ → A	•	•	↕	↕	R	S
	COMB													53	2	1	$\overline{B}$ → B	•	•	↕	↕	R	S
Decimal Adj. A	DAA													19	2	1	Adj binary sum to BCD	•	•	↕	↕	↕	↕
Decrement	DEC							6A	6	2	7A	6	3				M − 1 → M	•	•	↕	↕	↕	•
	DECA													4A	2	1	A − 1 → A	•	•	↕	↕	↕	•
	DECB													5A	2	1	B − 1 → B	•	•	↕	↕	↕	•
Exclusive OR	EORA	88	2	2	98	3	2	A8	4	2	B8	4	3				A ⊕ M → A	•	•	↕	↕	R	•
	EORB	C8	2	2	D8	3	2	E8	4	2	F8	4	3				B ⊕ M → B	•	•	↕	↕	R	•
Increment	INC							6C	6	2	7C	6	3				M + 1 → M	•	•	↕	↕	↕	•
	INCA													4C	2	1	A + 1 → A	•	•	↕	↕	↕	•
	INCB													5C	2	1	B + 1 → B	•	•	↕	↕	↕	•
Load Acmltrs	LDAA	86	2	2	96	3	2	A6	4	2	B6	4	3				M → A	•	•	↕	↕	R	•
	LDAB	C6	2	2	D6	3	2	E6	4	2	F6	4	3				M → B	•	•	↕	↕	R	•
Load Double	LDD	CC	3	3	DC	4	2	EC	5	2	FC	5	3				M:M + 1 → D	•	•	↕	↕	R	•
Logical Shift, Left	LSL							68	6	2	78	6	3					•	•	↕	↕	↕	↕
	LSLA													48	2	1		•	•	↕	↕	↕	↕
	LSLB													58	2	1		•	•	↕	↕	↕	↕
	LSLD													05	3	1		•	•	↕	↕	↕	↕

TABLE 10 — ACCUMULATOR AND MEMORY INSTRUCTIONS (Sheet 2 of 2)

Accumulator and Memory Operations	MNE	Immed			Direct			Index			Extend			Inher			Boolean Expression	Condition Codes					
		Op	~	#	Op	~	#	Op	~	#	Op	~	#	Op	~	#		H	I	N	Z	V	C
Shift Right, Logical	LSR							64	6	2	74	6	3					●	●	R	↕	↕	↕
	LSRA													44	2	1		●	●	R	↕	↕	↕
	LSRB													54	2	1		●	●	R	↕	↕	↕
	LSRD													04	3	1		●	●	R	↕	↕	↕
Multiply	MUL													3D	10	1	A X B → D	●	●	●	●	●	↕
2's Complement (Negate)	NEG							60	6	2	70	6	3				00 - M → M	●	●	↕	↕	↕	↕
	NEGA													40	2	1	00 - A → A	●	●	↕	↕	↕	↕
	NEGB													50	2	1	00 - B → B	●	●	↕	↕	↕	↕
No Operation	NOP													01	2	1	PC + 1 → PC	●	●	●	●	●	●
Inclusive OR	ORAA	8A	2	2	9A	3	2	AA	4	2	BA	4	3				A + M → A	●	●	↕	↕	R	●
	ORAB	CA	2	2	DA	3	2	EA	4	2	FA	4	3				B + M → B	●	●	↕	↕	R	●
Push Data	PSHA													36	3	1	A → Stack	●	●	●	●	●	●
	PSHB													37	3	1	B → Stack	●	●	●	●	●	●
Pull Data	PULA													32	4	1	Stack → A	●	●	●	●	●	●
	PULB													33	4	1	Stack → B	●	●	●	●	●	●
Rotate Left	ROL							69	6	2	79	6	3					●	●	↕	↕	↕	↕
	ROLA													49	2	1		●	●	↕	↕	↕	↕
	ROLB													59	2	1		●	●	↕	↕	↕	↕
Rotate Right	ROR							66	6	2	76	6	3					●	●	↕	↕	↕	↕
	RORA													46	2	1		●	●	↕	↕	↕	↕
	RORB													56	2	1		●	●	↕	↕	↕	↕
Subtract Acmltr	SBA													10	2	1	A - B → A	●	●	↕	↕	↕	↕
Subtract with Carry	SBCA	B2	2	2	92	3	2	A2	4	2	B2	4	3				A - M - C → A	●	●	↕	↕	↕	↕
	SBCB	C2	2	2	D2	3	2	E2	4	2	F2	4	3				B - M - C → B	●	●	↕	↕	↕	↕
Store Acmltrs	STAA				97	3	2	A7	4	2	B7	4	3				A → M	●	●	↕	↕	R	●
	STAB				D7	3	2	E7	4	2	F7	4	3				B → M	●	●	↕	↕	R	●
	STD				DD	4	2	ED	5	2	FD	5	3				D → M:M + 1	●	●	↕	↕	R	●
Subtract	SUBA	80	2	2	90	3	2	A0	4	2	B0	4	3				A - M → A	●	●	↕	↕	↕	↕
	SUBB	C0	2	2	D0	3	2	E0	4	2	F0	4	3				B - M → B	●	●	↕	↕	↕	↕
Subtract Double	SUBD	83	4	3	93	5	2	A3	6	2	B3	6	3				D - M:M + 1 → D	●	●	↕	↕	↕	↕
Transfer Acmltr	TAB													16	2	1	A → B	●	●	↕	↕	R	●
	TBA													17	2	1	B → A	●	●	↕	↕	R	●
Test, Zero or Minus	TST							6D	6	2	7D	6	3				M - 00	●	●	↕	↕	R	R
	TSTA													4D	2	1	A - 00	●	●	↕	↕	R	R
	TSTB													5D	2	1	B - 00	●	●	↕	↕	R	R

The condition code register notes are listed after Table 12.

TABLE 11 — JUMP AND BRANCH INSTRUCTIONS

Operations	MNEM	Direct			Relative			Index			Extend			Inherent			Branch Test	Condition Code Reg.					
		Op	~	#	Op	~	#	Op	~	#	Op	~	#	Op	~	#		5	4	3	2	1	0
																		H	I	N	Z	V	C
Branch Always	BRA				20	3	2										None	•	•	•	•	•	•
Branch Never	BRN				21	3	2										None	•	•	•	•	•	•
Branch If Carry Clear	BCC				24	3	2										C = 0	•	•	•	•	•	•
Branch If Carry Set	BCS				25	3	2										C = 1	•	•	•	•	•	•
Branch If = Zero	BEQ				27	3	2										Z = 1	•	•	•	•	•	•
Branch If ≥ Zero	BGE				2C	3	2										N ⊕ V = 0	•	•	•	•	•	•
Branch If > Zero	BGT				2E	3	2										Z + (N ⊕ V) = 0	•	•	•	•	•	•
Branch If Higher	BHI				22	3	2										C + Z = 0	•	•	•	•	•	•
Branch If Higher or Same	BHS				24	3	2										C = 0	•	•	•	•	•	•
Branch If ≤ Zero	BLE				2F	3	2										Z + (N ⊕ V) = 1	•	•	•	•	•	•
Branch If Carry Set	BLO				25	3	2										C = 1	•	•	•	•	•	•
Branch If Lower Or Same	BLS				23	3	2										C + Z = 1	•	•	•	•	•	•
Branch If < Zero	BLT				2D	3	2										N ⊕ V = 1	•	•	•	•	•	•
Branch If Minus	BMI				2B	3	2										N = 1	•	•	•	•	•	•
Branch If Not Equal Zero	BNE				26	3	2										Z = 0	•	•	•	•	•	•
Branch If Overflow Clear	BVC				28	3	2										V = 0	•	•	•	•	•	•
Branch If Overflow Set	BVS				29	3	2										V = 1	•	•	•	•	•	•
Branch If Plus	BPL				2A	3	2										N = 0	•	•	•	•	•	•
Branch To Subroutine	BSR				8D	6	2											•	•	•	•	•	•
Jump	JMP							6E	3	2	7E	3	3				See Special Operations-Figure 24	•	•	•	•	•	•
Jump To Subroutine	JSR	9D	5	2				AD	6	2	BD	6	3					•	•	•	•	•	•
No Operation	NOP													01	2	1		•	•	•	•	•	•
Return From Interrupt	RTI													3B	10	1		↕	↕	↕	↕	↕	↕
Return From Subroutine	RTS													39	5	1	See Special Operations-Figure 24	•	•	•	•	•	•
Software Interrupt	SWI													3F	12	1		•	S	•	•	•	•
Wait For Interrupt	WAI													3E	9	1		•	•	•	•	•	•

TABLE 12 — CONDITION CODE REGISTER MANIPULATION INSTRUCTIONS

Operations	Inherent				Boolean Operation	Condition Code Register					
	MNEM	Op	~	#		5	4	3	2	1	0
						H	I	N	Z	V	C
Clear Carry	CLC	0C	2	1	0 → C	•	•	•	•	•	R
Clear Interrupt Mask	CLI	0E	2	1	0 → I	•	R	•	•	•	•
Clear Overflow	CLV	0A	2	1	0 → V	•	•	•	•	R	•
Set Carry	SEC	0D	2	1	1 → C	•	•	•	•	•	S
Set Interrupt Mask	SEI	0F	2	1	1 → I	•	S	•	•	•	•
Set Overflow	SEV	0B	2	1	1 → V	•	•	•	•	S	•
Accumulator A → CCR	TAP	06	2	1	A → CCR	↕	↕	↕	↕	↕	↕
CCR → Accumulator A	TPA	07	2	1	CCR → A	•	•	•	•	•	•

LEGEND

- Op Operation Code (Hexadecimal)
- ~ Number of MPU Cycles
- M_{SP} Contents of memory location pointed to by Stack Pointer
- # Number of Program Bytes
- + Arithmetic Plus
- − Arithmetic Minus
- • Boolean AND
- X Arithmetic Multiply
- + Boolean Inclusive OR
- ⊕ Boolean Exclusive OR
- M Complement of M
- → Transfer Into
- 0 Bit = Zero
- 00 Byte = Zero

CONDITION CODE SYMBOLS

- H Half-carry from bit 3
- I Interrupt mask
- N Negative (sign bit)
- Z Zero (byte)
- V Overflow, 2's complement
- C Carry/Borrow from MSB
- R Reset Always
- S Set Always
- ↕ Affected
- • Not Affected

TABLE 13 — INSTRUCTION EXECUTION TIMES IN E CYCLES

	ADDRESSING MODE					
	Immediate	Direct	Extended	Indexed	Inherent	Relative
ABA	●	●	●	●	2	●
ABX	●	●	●	●	3	●
ADC	2	3	4	4	●	●
ADD	2	3	4	4	●	●
ADDD	4	5	6	6	●	●
AND	2	3	4	4	●	●
ASL	●	●	6	6	2	●
ASLD	●	●	●	●	3	●
ASR	●	●	6	6	2	●
BCC	●	●	●	●	●	3
BCS	●	●	●	●	●	3
BEQ	●	●	●	●	●	3
BGE	●	●	●	●	●	3
BGT	●	●	●	●	●	3
BHI	●	●	●	●	●	3
BHS	●	●	●	●	●	3
BIT	2	3	4	4	●	●
BLE	●	●	●	●	●	3
BLO	●	●	●	●	●	3
BLS	●	●	●	●	●	3
BLT	●	●	●	●	●	3
BMI	●	●	●	●	●	3
BNE	●	●	●	●	●	3
BPL	●	●	●	●	●	3
BRA	●	●	●	●	●	3
BRN	●	●	●	●	●	3
BSR	●	●	●	●	●	6
BVC	●	●	●	●	●	3
BVS	●	●	●	●	●	3
CBA	●	●	●	●	2	●
CLC	●	●	●	●	2	●
CLI	●	●	●	●	2	●
CLR	●	●	6	6	2	●
CLV	●	●	●	●	2	●
CMP	2	3	4	4	●	●
COM	●	●	6	6	2	●
CPX	4	5	6	6	●	●
DAA	●	●	●	●	2	●
DEC	●	●	6	6	2	●
DES	●	●	●	●	3	●
DEX	●	●	●	●	3	●
EOR	2	3	4	4	●	●
INC	●	●	6	6	●	●
INS	●	●	●	●	3	●

	ADDRESSING MODE					
	Immediate	Direct	Extended	Indexed	Inherent	Relative
INX	●	●	●	●	3	●
JMP	●	●	3	3	●	●
JSR	●	5	6	6	●	●
LDA	2	3	4	4	●	●
LDD	3	4	5	5	●	●
LDS	3	4	5	5	●	●
LDX	3	4	5	5	●	●
LSL	●	●	6	6	2	●
LSLD	●	●	●	●	3	●
LSR	●	●	6	6	2	●
LSRD	●	●	●	●	3	●
MUL	●	●	●	●	10	●
NEG	●	●	6	6	2	●
NOP	●	●	●	●	2	●
ORA	2	3	4	4	●	●
PSH	●	●	●	●	3	●
PSHX	●	●	●	●	4	●
PUL	●	●	●	●	4	●
PULX	●	●	●	●	5	●
ROL	●	●	6	6	2	●
ROR	●	●	6	6	2	●
RTI	●	●	●	●	10	●
RTS	●	●	●	●	5	●
SBA	●	●	●	●	2	●
SBC	2	3	4	4	●	●
SEC	●	●	●	●	2	●
SEI	●	●	●	●	2	●
SEV	●	●	●	●	2	●
STA	●	3	4	4	●	●
STD	●	4	5	5	●	●
STS	●	4	5	5	●	●
STX	●	4	5	5	●	●
SUB	2	3	4	4	●	●
SUBD	4	5	6	6	●	●
SWI	●	●	●	●	12	●
TAB	●	●	●	●	2	●
TAP	●	●	●	●	2	●
TBA	●	●	●	●	2	●
TPA	●	●	●	●	2	●
TST	●	●	6	6	2	●
TSX	●	●	●	●	3	●
TXS	●	●	●	●	3	●
WAI	●	●	●	●	9	●

SUMMARY OF CYCLE-BY-CYCLE OPERATION

Table 14 provides a detailed description of the information present on the Address Bus, Data Bus, and the Read/Write (R/W) line during each cycle of each instruction.

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

Note that during MPU reads of internal locations, the resultant value will not appear on the external Data Bus except in Mode 0. "High order" byte refers to the most significant byte of a 16-bit value.

TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 1 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/W Line	Data Bus
IMMEDIATE					
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	2	1 2	Opcode Address Opcode Address + 1	1 1	Opcode Operand Data
LDS LDX LDD	3	1 2 3	Opcode Address Opcode Address + 1 Opcode Address + 2	1 1 1	Opcode Operand Data (High Order Byte) Operand Data (Low Order Byte)
CPX SUBD ADDD	4	1 2 3 4	Opcode Address Opcode Address + 1 Opcode Address + 2 Address Bus FFFF	1 1 1 1	Opcode Operand Data (High Order Byte) Operand Data (Low Order Byte) Low Byte of Restart Vector
DIRECT					
ADC EOR ADD LDA AND ORA BIT SBC CMP SUB	3	1 2 3	Opcode Address Opcode Address + 1 Address of Operand	1 1 1	Opcode Address of Operand Operand Data
STA	3	1 2 3	Opcode Address Opcode Address + 1 Destination Address	1 1 0	Opcode Destination Address Data from Accumulator
LDS LDX LDD	4	1 2 3 4	Opcode Address Opcode Address + 1 Address of Operand Operand Address + 1	1 1 1 1	Opcode Address of Operand Operand Data (High Order Byte) Operand Data (Low Order Byte)
STS STX STD	4	1 2 3 4	Opcode Address Opcode Address + 1 Address of Operand Address of Operand + 1	1 1 0 0	Opcode Address of Operand Register Data (High Order Byte) Register Data (Low Order Byte)
CPX SUBD ADDD	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Operand Address Operand Address + 1 Address Bus FFFF	1 1 1 1 1	Opcode Address of Operand Operand Data (High Order Byte) Operand Data (Low Order Byte) Low Byte of Restart Vector
JSR	5	1 2 3 4 5	Opcode Address Opcode Address + 1 Subroutine Address Stack Pointer Stack Pointer − 1	1 1 1 0 0	Opcode Irrelevant Data First Subroutine Opcode Return Address (Low Order Byte) Return Address (High Order Byte)

TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 2 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/W Line	Data Bus
EXTENDED					
JMP	3	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Jump Address (High Order Byte)
		3	Opcode Address + 2	1	Jump Address (Low Order Byte)
ADC EOR	4	1	Opcode Address	1	Opcode
ADD LDA		2	Opcode Address + 1	1	Address of Operand
AND ORA		3	Opcode Address + 2	1	Address of Operand (Low Order Byte)
BIT SBC		4	Address of Operand	1	Operand Data
CMP SUB					
STA	4	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Destination Address (High Order Byte)
		3	Opcode Address + 2	1	Destination Address (Low Order Byte)
		4	Operand Destination Address	0	Data from Accumulator
LDS	5	1	Opcode Address	1	Opcode
LDX		2	Opcode Address + 1	1	Address of Operand (High Order Byte)
LDD		3	Opcode Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	1	Operand Data (High Order Byte)
		5	Address of Operand + 1	1	Operand Data (Low Order Byte)
STS	5	1	Opcode Address	1	Opcode
STX		2	Opcode Address + 1	1	Address of Operand (High Order Byte)
STD		3	Opcode Address + 2	1	Address of Operand (Low Order Byte)
		4	Address of Operand	0	Operand Data (High Order Byte)
		5	Address of Operand + 1	0	Operand Data (Low Order Byte)
ASL LSR	6	1	Opcode Address	1	Opcode
ASR NEG		2	Opcode Address + 1	1	Address of Operand (High Order Byte)
CLR ROL		3	Opcode Address + 2	1	Address of Operand (Low Order Byte)
COM ROR		4	Address of Operand	1	Current Operand Data
DEC TST*		5	Address Bus FFFF	1	Low Byte of Restart Vector
INC		6	Address of Operand	0	New Operand Data
CPX	6	1	Opcode Address	1	Opcode
SUBD		2	Opcode Address + 1	1	Operand Address (High Order Byte)
ADDD		3	Opcode Address + 2	1	Operand Address (Low Order Byte)
		4	Operand Address	1	Operand Data (High Order Byte)
		5	Operand Address + 1	1	Operand Data (Low Order Byte)
		6	Address Bus FFFF	1	Low Byte of Restart Vector
JSR	6	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Address of Subroutine (High Order Byte)
		3	Opcode Address + 2	1	Address of Subroutine (Low Order Byte)
		4	Subroutine Starting Address	1	Opcode of Next Instruction
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer − 1	0	Return Address (High Order Byte)

*TST does not perform the write cycle during the sixth cycle. The sixth cycle is another address bus = $FFFF.

TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 3 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
INDEXED					
JMP	3	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
ADC EOR	4	1	Opcode Address	1	Opcode
ADD LDA		2	Opcode Address + 1	1	Offset
AND ORA		3	Address Bus FFFF	1	Low Byte of Restart Vector
BIT SBC		4	Index Register Plus Offset	1	Operand Data
CMP SUB					
STA	4	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	0	Operand Data
LDS	5	1	Opcode Address	1	Opcode
LDX		2	Opcode Address + 1	1	Offset
LDD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	1	Operand Data (High Order Byte)
		5	Index Register Plus Offset + 1	1	Operand Data (Low Order Byte)
STS	5	1	Opcode Address	1	Opcode
STX		2	Opcode Address + 1	1	Offset
STD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register Plus Offset	0	Operand Data (High Order Byte)
		5	Index Register Plus Offset + 1	0	Operand Data (Low Order Byte)
ASL LSR	6	1	Opcode Address	1	Opcode
ASR NEG		2	Opcode Address + 1	1	Offset
CLR ROL		3	Address Bus FFFF	1	Low Byte of Restart Vector
COM ROR		4	Index Register Plus Offset	1	Current Operand Data
DEC TST*		5	Address Bus FFFF	1	Low Byte of Restart Vector
INC		6	Index Register Plus Offset	0	New Operand Data
CPX	6	1	Opcode Address	1	Opcode
SUBD		2	Opcode Address + 1	1	Offset
ADDD		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register + Offset	1	Operand Data (High Order Byte)
		5	Index Register + Offset + 1	1	Operand Data (Low Order Byte)
		6	Address Bus FFFF		Low Byte of Restart Vector
JSR	6	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Index Register + Offset	1	First Subroutine Opcode
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer − 1	0	Return Address (High Order Byte)

*TST does not perform the write cycle during the sixth cycle. The sixth cycle is another address bus = $FFFF.

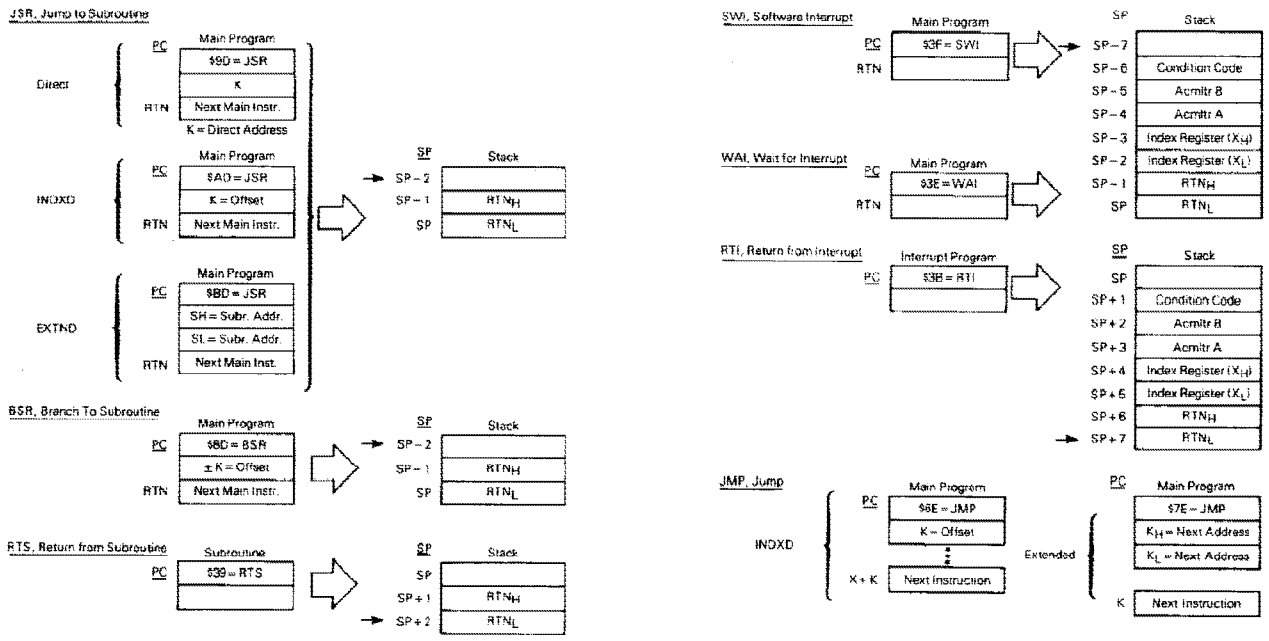
TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 4 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/$\overline{W}$ Line	Data Bus
INHERENT					
ABA DAA SEC ASL DEC SEI ASR INC SEV CBA LSR TAB CLC NEG TAP CLI NOP TBA CLR ROL TPA CLV ROR TST COM SBA	2	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Opcode of Next Instruction
ABX	3	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Irrelevant Data
		3	Address Bus FFFF	1	Low Byte of Restart Vector
ASLD LSRD	3	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Irrelevant Data
		3	Address Bus FFFF	1	Low Byte of Restart Vector
DES INS	3	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Opcode of Next Instruction
		3	Previous Stack Pointer Contents	1	Irrelevant Data
INX DEX	3	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Opcode of Next Instruction
		3	Address Bus FFFF	1	Low Byte of Restart Vector
PSHA PSHB	3	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Opcode of Next Instruction
		3	Stack Pointer	0	Accumulator Data
TSX	3	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Opcode of Next Instruction
		3	Stack Pointer	1	Irrelevant Data
TXS	3	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Opcode of Next Instruction
		3	Address Bus FFFF	1	Low Byte of Restart Vector
PULA PULB	4	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Opcode of Next Instruction
		3	Stack Pointer	1	Irrelevant Data
		4	Stack Pointer + 1	1	Operand Data from Stack
PSHX	4	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Irrelevant Data
		3	Stack Pointer	0	Index Register (Low Order Byte)
		4	Stack Pointer − 1	0	Index Register (High Order Byte)
PULX	5	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Irrelevant Data
		3	Stack Pointer	1	Irrelevant Data
		4	Stack Pointer + 1	1	Index Register (High Order Byte)
		5	Stack Pointer + 2	1	Index Register (Low Order Byte)
RTS	5	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Irrelevant Data
		3	Stack Pointer	1	Irrelevant Data
		4	Stack Pointer + 1	1	Address of Next Instruction (High Order Byte)
		5	Stack Pointer + 2	1	Address of Next Instruction (Low Order Byte)
WAI	9	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Opcode of Next Instruction
		3	Stack Pointer	0	Return Address (Low Order Byte)
		4	Stack Pointer − 1	0	Return Address (High Order Byte)
		5	Stack Pointer − 2	0	Index Register (Low Order Byte)
		6	Stack Pointer − 3	0	Index Register (High Order Byte)
		7	Stack Pointer − 4	0	Contents of Accumulator A
		8	Stack Pointer − 5	0	Contents of Accumulator B
		9	Stack Pointer − 6	0	Contents of Condition Code Register

TABLE 14 — CYCLE-BY-CYCLE OPERATION (Sheet 5 of 5)

Address Mode and Instructions	Cycles	Cycle #	Address Bus	R/W Line	Data Bus
INHERENT					
MUL	10	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Irrelevant Data
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Address Bus FFFF	1	Low Byte of Restart Vector
		5	Address Bus FFFF	1	Low Byte of Restart Vector
		6	Address Bus FFFF	1	Low Byte of Restart Vector
		7	Address Bus FFFF	1	Low Byte of Restart Vector
		8	Address Bus FFFF	1	Low Byte of Restart Vector
		9	Address Bus FFFF	1	Low Byte of Restart Vector
		10	Address Bus FFFF	1	Low Byte of Restart Vector
RTI	10	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Irrelevant Data
		3	Stack Pointer	1	Irrelevant Data
		4	Stack Pointer + 1	1	Contents of Condition Code Register from Stack
		5	Stack Pointer + 2	1	Contents of Accumulator B from Stack
		6	Stack Pointer + 3	1	Contents of Accumulator A from Stack
		7	Stack Pointer + 4	1	Index Register from Stack (High Order Byte)
		8	Stack Pointer + 5	1	Index Register from Stack (Low Order Byte)
		9	Stack Pointer + 6	1	Next Instruction Address from Stack (High Order Byte)
		10	Stack Pointer + 7	1	Next Instruction Address from Stack (Low Order Byte)
SWI	12	1	Opcode Address	1	Opcode
		2	Opcode Address + 1	1	Irrelevant Data
		3	Stack Pointer	0	Return Address (Low Order Byte)
		4	Stack Pointer − 1	0	Return Address (High Order Byte)
		5	Stack Pointer − 2	0	Index Register (Low Order Byte)
		6	Stack Pointer − 3	0	Index Register (High Order Byte)
		7	Stack Pointer − 4	0	Contents of Accumulator A
		8	Stack Pointer − 5	0	Contents of Accumulator B
		9	Stack Pointer − 6	0	Contents of Condition Code Register
		10	Stack Pointer − 7	1	Irrelevant Data
		11	Vector Address FFFA (Hex)	1	Address of Subroutine (High Order Byte)
		12	Vector Address FFFB (Hex)	1	Address of Subroutine (Low Order Byte)
RELATIVE					
BCC BHT BNE BLO BCS BLE BPL BHS BEQ BLS BRA BRN BGE BLT BVC BGT BMT BVS	3	1	Op Code Address	1	Op Code
		2	Op Code Address +1	1	Branch Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
BSR	6	1	Op Code Address	1	Op Code
		2	Op Code Address +1	1	Branch Offset
		3	Address Bus FFFF	1	Low Byte of Restart Vector
		4	Subroutine Starting Address	1	Op Code of Next Instruction
		5	Stack Pointer	0	Return Address (Low Order Byte)
		6	Stack Pointer -1	0	Return Address (High Order Byte)

FIGURE 24 — SPECIAL OPERATIONS



Legend:
RTN = Address of next instruction in Main Program to be executed upon return from subroutine
RTN_H = Most significant byte of Return Address
RTN_L = Least significant byte of Return Address
→ = Stack Pointer After Execution
K = 8-bit Unsigned Value

EPROM PROGRAMMING ROUTINE

```
PAGE  001  EPROM   .SA:1  EPROM  *** ROUTINE TO PROGRAM THE MC68701 EPROM ***

00001                         NAM    EPROM
00002                         OPT    Z01,LLEN=80
00003                         TTL    *** ROUTINE TO PROGRAM THE MC68701 EPROM **
00004
00005                   ********************************************************
00006                   *
00007                   *  E P R O M -- A  NON-REENTRANT  ROUTINE  TO  PROGRAM
00008                   *                THE MC68701 EPROM.
00009                   *
00010                   *                THE ROUTINE PROGRAMS THE MC68701 EPROM
00011                   *                STARTING  AT  ADDRESS  "PNTR"  FROM  A
00012                   *                BLOCK OF MEMORY  STARTING  AT  "IMBEG"
00013                   *                AND ENDING AT "IMEND".
00014                   *
00015                   *  CALLING CONVENTION:
00016                   *
00017                   *     JSR  EPROM
00018                   *
00019                   *  NOTES:
00020                   *
00021                   *     1.  THE ROUTINE EXPECTS FOUR DOUBLE BYTE  VALUES
00022                   *         TO  BE INITIALIZED PRIOR  TO  BEING  CALLED.
00023                   *         THESE VALUES ARE:
00024                   *
00025                   *         IMBEG = A DOUBLE BYTE ADDRESS  WHICH  POINTS
00026                   *                 TO THE FIRST BYTE TO  BE  PROGRAMMED
00027                   *                 INTO THE EPROM.
00028                   *
00029                   *         IMEND = A DOUBLE BYTE ADDRESS  WHICH  POINTS
00030                   *                 TO THE LAST BYTE TO BE PROGRAMED IN-
00031                   *                 INTO THE EPROM.
00032                   *
00033                   *         PNTR  = A DOUBLE BYTE ADDRESS  WHICH  POINTS
00034                   *                 TO THE FIRST BYTE IN THE EPROM TO BE
00035                   *                 PROGRAMMED.
00036                   *
00037                   *         WAIT  = A DOUBLE BYTE COUNTER VALUE WHICH IS
00038                   *                 A FUNCTION OF THE MCU INPUT FREQUEN-
00039                   *                 CY AND IS USED WITH THE OUTPUT  COM-
00040                   *                 PARE FUNCTION TO  GENERATE A 50 MSEC
00041                   *                 TIMEOUT.  IT IS EQUIVALENT TO
00042                   *
00043                   *                 50000 * (MCU INPUT FREQ) / 4 * 10**6
00044                   *
00045                   *                 VALUES FOR TYPICAL INPUT FREQS  ARE:
00046                   *
00047                   *                   WAIT                MCU INPUT FREQ
00048                   *               -------------           --------------
00049                   *               30615 ($7797)              2.45 MHZ
00050                   *               50000 ($C350)              4.00 MHZ
00051                   *               61375 ($EFBF)              4.91 MHZ
00052                   *
00053                   *     2.  IT IS ASSUMED THAT POWER (VPP) IS  AVAILABLE
00054                   *         TO THE RESET PIN FOR PROGRAMMING.
00055                   *
00056                   *     3.  THIS ROUTINE  PERFORMS  NO  ERROR  CHECKING.
00057                   *
00058                   *  Routine parameter initialization, such as stack pointer, etc., must be done prior to entry.
                           (Use of PRObug will ensure all needed initialization.)
```

EPROM PROGRAMMING ROUTINE

```
PAGE  002  EPROM   .SA:1  EPROM  *** ROUTINE TO PROGRAM THE MC68701 EPROM ***

00060
00061                    *  E Q U A T E S
00062
00063            0008  A TCSR   EQU    $08       TIMER CONTROL/STAT REGISTER
00064            0009  A TIMER  EQU    $09       COUNTER REGISTER
00065            000B  A OUTCMP EQU    $0B       OUTPUT COMPARE REGISTER
00066            0014  A EPMCNT EQU    $14       RAM/EPROM CONTROL REGISTER
00067
00068                    *  L O C A L    V A R I A B L E S
00069
00070A 0080                     ORG    $80
00071A 0080      0002  A IMBEG  RMB    2         START OF MEMORY BLOCK
00072A 0082      0002  A IMEND  RMB    2         LAST BYTE OF MEMORY BLOCK
00073A 0084      0002  A PNTR   RMB    2         FIRST BYTE OF EPROM TO BE PGM'D
00074A 0086      0002  A WAIT   RMB    2         COUNTER VALUE
00075
00076                    *  E P R O M    S T A R T S    H E R E
00077
00078A 3000                     ORG    $3000
00079A 3000 DE 84     A EPROM  LDX    PNTR      SAVE CALLING ARGUMENT
00080A 3002 3C                  PSHX             RESTORE WHEN DONE
00081A 3003 DE 80     A        LDX    IMBEG     USE STACK
00082
00083A 3005 3C          EPR002 PSHX              SAVE POINTER ON STACK
00084A 3006 86 FE     A        LDAA   #$FE      REMOVE VPP, SET LATCH
00085A 3008 97 14     A        STAA   EPMCNT    PPC=1, PLC=0
00086A 300A A6 00     A        LDAA   X         MOVE DATA MEMORY-TO-LATCH
00087A 300C DE 84     A        LDX    PNTR      GET WHERE TO PUT IT
00088A 300E A7 00     A        STAA   X         STASH AND LATCH
00089A 3010 08                 INX              NEXT ADDR
00090A 3011 DF 84     A        STX    PNTR      ALL SET FOR NEXT
00091A 3013 86 FC     A        LDAA   #$FC      ENABLE EPROM POWER (VPP)
00092A 3015 97 14     A        STAA   EPMCNT    PPC=0, PLC=0
00093
00094                    * NOW WAIT FOR 50 MSEC TIMEOUT USING OUTPUT COMPARE.
00095
00096A 3017 DC 86     A        LDD    WAIT      GET CYCLE COUNTER
00097A 3019 D3 09     A        ADDD   TIMER     BUMP CURRENT VALUE
00098A 301B 7F 0008   A        CLR    TCSR      CLEAR OCF
00099A 301E DD 0B     A        STD    OUTCMP    SET OUTPUT COMPARE
00100A 3020 86 40     A        LDAA   #$40      NOW WAIT FOR OCF
00101
00102A 3022 95 08     A EPR004 BITA   TCSR
00103A 3024 27 FC 3022         BEQ    EPR004    NOT YET
00104A 3026 38                 PULX             SETUP FOR NEXT ONE
00105A 3027 08                 INX              NEXT
00106A 3028 9C 82     A        CPX    IMEND     MAYBE DONE
00107A 302A 23 D9 3005         BLS    EPR002    NOT YET
00108A 302C 86 FF     A        LDAA   #$FF      REMOVE VPP, INHIBIT LATCH
00109A 302E 97 14     A        STAA   EPMCNT    EPROM CAN NOW BE READ
00110A 3030 38                 PULX             RESTORE PNTR
00111A 3031 DF 84     A        STX    PNTR
00112A 3033 39                 RTS              THAT'S ALL
00113                          END
TOTAL ERRORS 00000--00000
```

IMPORTANT NOTICE

Devices made with mask numbers T7A and C84 may generate multiple framing error flags in response to unframed data. These devices will eventually synchronize correctly after a framing error; but valid, framed data following an unframed byte may generate false framing error flags.