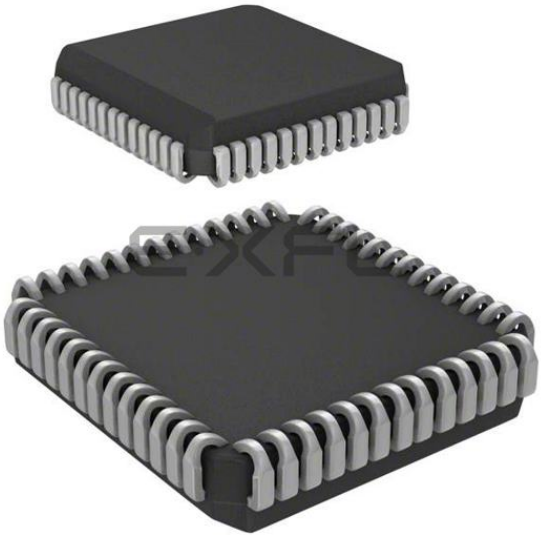


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	HC11
Core Size	8-Bit
Speed	2MHz
Connectivity	SCI, SPI
Peripherals	POR, WDT
Number of I/O	38
Program Memory Size	20KB (20K x 8)
Program Memory Type	OTP
EEPROM Size	512 x 8
RAM Size	768 x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 5.5V
Data Converters	A/D 8x8b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	52-LCC (J-Lead)
Supplier Device Package	52-PLCC (19.1x19.1)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68hc711e20cfn2

NOTE

$\overline{IRQ}$ must be configured for level-sensitive operation if there is more than one source of $\overline{IRQ}$ interrupt.

There should be a single pullup resistor near the MCU interrupt input pin (typically 4.7 kΩ). There must also be an interlock mechanism at each interrupt source so that the source holds the interrupt line low until the MCU recognizes and acknowledges the interrupt request. If one or more interrupt sources are still pending after the MCU services a request, the interrupt line will still be held low and the MCU will be interrupted again as soon as the interrupt mask bit in the MCU is cleared (normally upon return from an interrupt). Refer to Chapter 5 Resets and Interrupts.

V_{PPE} is the input for the 12-volt nominal programming voltage required for EPROM/OTPROM programming. On devices without EPROM/OTPROM, this pin is only an $\overline{XIRQ}$ input.

CAUTION

During EPROM programming of the MC68HC711E9 device, the V_{PPE} pin circuitry may latch-up and be damaged if the input current is not limited to 10 mA. For more information please refer to MC68HC711E9 8-Bit Microcontroller Unit Mask Set Errata 3 (Freescale document order number 68HC711E9MSE3.

1.4.7 MODA and MODB (MODA/$\overline{LIR}$ and MODB/V_{STBY})

During reset, MODA and MODB select one of the four operating modes:

- Single-chip mode
- Expanded mode
- Test mode
- Bootstrap mode

Refer to Chapter 2 Operating Modes and On-Chip Memory.

After the operating mode has been selected, the load instruction register ($\overline{LIR}$) pin provides an open-drain output to indicate that execution of an instruction has begun. A series of E-clock cycles occurs during execution of each instruction. The $\overline{LIR}$ signal goes low during the first E-clock cycle of each instruction (opcode fetch). This output is provided for assistance in program debugging.

The V_{STBY} pin is used to input random-access memory (RAM) standby power. When the voltage on this pin is more than one MOS threshold (about 0.7 volts) above the V_{DD} voltage, the internal RAM and part of the reset logic are powered from this signal rather than the V_{DD} input. This allows RAM contents to be retained without V_{DD} power applied to the MCU. Reset must be driven low before V_{DD} is removed and must remain low until V_{DD} has been restored to a valid level.

1.4.8 V_{RL} and V_{RH}

These two inputs provide the reference voltages for the analog-to-digital (A/D) converter circuitry:

- V_{RL} is the low reference, typically 0 Vdc.
- V_{RH} is the high reference.

For proper A/D converter operation:

- V_{RH} should be at least 3 Vdc greater than V_{RL}.
- V_{RL} and V_{RH} should be between V_{SS} and V_{DD}.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$1000	Port A Data Register (PORTA) See page 98.	Read: Write:	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
		Reset:	I	0	0	0	I	I	I	I
$1001	Reserved		R	R	R	R	R	R	R	R
$1002	Parallel I/O Control Register (PIOC) See page 102.	Read: Write:	STAF	STAI	CWOM	HNDS	OIN	PLS	EGA	INVB
		Reset:	0	0	0	0	0	U	1	1
$1003	Port C Data Register (PORTC) See page 99.	Read: Write:	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
		Reset:	Indeterminate after reset							
$1004	Port B Data Register (PORTB) See page 99.	Read: Write:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
		Reset:	0	0	0	0	0	0	0	0
$1005	Port C Latched Register (PORTCL) See page 99.	Read: Write:	PCL7	PCL6	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0
		Reset:	Indeterminate after reset							
$1006	Reserved		R	R	R	R	R	R	R	R
$1007	Port C Data Direction Register (DDRC) See page 100.	Read: Write:	DDRC7	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0
		Reset:	0	0	0	0	0	0	0	0
$1008	Port D Data Register (PORTD) See page 100.	Read: Write:	0	0	PD5	PD4	PD3	PD2	PD1	PD0
		Reset:	U	U	I	I	I	I	I	I
$1009	Port D Data Direction Register (DDRD) See page 100.	Read: Write:			DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
		Reset:	0	0	0	0	0	0	0	0
$100A	Port E Data Register (PORTE) See page 101.	Read: Write:	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
		Reset:	Indeterminate after reset							
$100B	Timer Compare Force Register (CFORC) See page 135.	Read: Write:	FOC1	FOC2	FOC3	FOC4	FOC5			
		Reset:	0	0	0	0	0	0	0	0
$100C	Output Compare 1 Mask Register (OC1M) See page 136.	Read: Write:	OC1M7	OC1M6	OC1M5	OC1M4	OC1M3			
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved U = Unaffected

I = Indeterminate after reset

Figure 2-7. Register and Control Bit Assignments (Sheet 1 of 6)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$1019	Timer Output Compare 2 Register Low (TOC2L) See page 134.	Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Reset:	1	1	1	1	1	1	1	1
$101A	Timer Output Compare 3 Register High (TOC3H) See page 135.	Read: Write:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Reset:	1	1	1	1	1	1	1	1
$101B	Timer Output Compare 3 Register Low (TOC3L) See page 135.	Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Reset:	1	1	1	1	1	1	1	1
$101C	Timer Output Compare 4 Register High (TOC4H) See page 135.	Read: Write:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Reset:	1	1	1	1	1	1	1	1
$101D	Timer Output Compare 4 Register Low (TOC4L) See page 135.	Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Reset:	1	1	1	1	1	1	1	1
$101E	Timer Input Capture 4/Output Compare 5 Register High (TI4/O5) See page 133.	Read: Write:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Reset:	1	1	1	1	1	1	1	1
$101F	Timer Input Capture 4/Output Compare 5 Register Low (TI4/O5) See page 133.	Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Reset:	1	1	1	1	1	1	1	1
$1020	Timer Control Register 1 (TCTL1) See page 137.	Read: Write:	OM2	OL2	OM3	OL3	OM4	OL4	OM5	OL5
		Reset:	0	0	0	0	0	0	0	0
$1021	Timer Control Register 2 (TCTL2) See page 131.	Read: Write:	EDG4B	EDG4A	EDG1B	EDG1A	EDG2B	EDG2A	EDG3B	EDG3A
		Reset:	0	0	0	0	0	0	0	0
$1022	Timer Interrupt Mask 1 Register (TMSK1) See page 138.	Read: Write:	OC1I	OC2I	OC3I	OC4I	I4/O5I	IC1I	IC2I	IC3I
		Reset:	0	0	0	0	0	0	0	0
$1023	Timer Interrupt Flag 1 (TFLG1) See page 138.	Read: Write:	OC1F	OC2F	OC3F	OC4F	I4/O5F	IC1F	IC2F	IC3F
		Reset:	0	0	0	0	0	0	0	0
$1024	Timer Interrupt Mask 2 Register (TMSK2) See page 139.	Read: Write:	TOI	RTII	PAOVI	PAII			PR1	PR0
		Reset:	0	0	0	0	0	0	0	0

= Unimplemented R = Reserved U = Unaffected

I = Indeterminate after reset

Figure 2-7. Register and Control Bit Assignments (Sheet 3 of 6)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$1025	Timer Interrupt Flag 2 (TFLG2) See page 142.	Read:	TOF	RTIF	PAOVF	PAIF				
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$1026	Pulse Accumulator Control Register (PACTL) See page 142.	Read:	DDRA7	PAEN	PAMOD	PEDGE	DDRA3	I4/O5	RTR1	RTR0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$1027	Pulse Accumulator Count Register (PACNT) See page 146.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$1028	Serial Peripheral Control Register (SPCR) See page 123.	Read:	SPIE	SPE	DWOM	MSTR	CPOL	CPHA	SPR1	SPR0
		Write:								
		Reset:	0	0	0	0	0	1	U	U
$1029	Serial Peripheral Status Register (SPSR) See page 124.	Read:	SPIF	WCOL		MODF				
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$102A	Serial Peripheral Data I/O Register (SPDR) See page 125.	Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Write:								
		Reset:	Indeterminate after reset							
$102B	Baud Rate Register (BAUD) See page 113.	Read:	TCLR	SCP2(1)	SCP1	SCP0	RCKB	SCR2	SCR1	SCR0
		Write:								
		Reset:	0	0	0	0	0	U	U	U
$102C	Serial Communications Control Register 1 (SCCR1) See page 110.	Read:	R8	T8		M	WAKE			
		Write:								
		Reset:	I	I	0	0	0	0	0	0
$102D	Serial Communications Control Register 2 (SCCR2) See page 111.	Read:	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$102E	Serial Communications Status Register (SCSR) See page 112.	Read:	TDRE	TC	RDRF	IDLE	OR	NF	FE	
		Write:								
		Reset:	1	1	0	0	0	0	0	0

1. SCP2 adds ÷39 to SCI prescaler and is present only in MC68HC(7)11E20.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$102F	Serial Communications Data Register (SCDR) See page 110.	Read:	R7/T7	R6/T6	R5/T5	R4/T4	R3/T3	R2/T2	R1/T1	R0/T0
		Write:								
		Reset:	Indeterminate after reset							
$1030	Analog-to-Digital Control Status Register (ADCTL) See page 62.	Read:	CCF		SCAN	MULT	CD	CC	CB	CA
		Write:								
		Reset:	0	0	Indeterminate after reset					

= Unimplemented R = Reserved U = Unaffected

I = Indeterminate after reset

Figure 2-7. Register and Control Bit Assignments (Sheet 4 of 6)

Address: $1036

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	MBE		ELAT	EXCOL	EXROW	T1	T0	PGM
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 2-15. MC68HC711E20 EPROM Programming Control Register (EPROG)

MBE — Multiple-Byte Programming Enable Bit

When multiple-byte programming is enabled, address bit 5 is considered a don't care so that bytes with address bit 5 = 0 and address bit 5 = 1 both get programmed. MBE can be read in any mode and always reads 0 in normal modes. MBE can be written only in special modes.

0 = EPROM array configured for normal programming
1 = Program two bytes with the same data

Bit 6 — Unimplemented

Always reads 0

ELAT — EPROM/OTPROM Latch Control Bit

When ELAT = 1, writes to EPROM cause address and data to be latched and the EPROM/OTPROM cannot be read. ELAT can be read any time. ELAT can be written any time except when PGM = 1; then the write to ELAT is disabled.

0 = EPROM/OTPROM address and data bus configured for normal reads
1 = EPROM/OTPROM address and data bus configured for programming

EXCOL — Select Extra Columns Bit

0 = User array selected
1 = User array is disabled and extra columns are accessed at bits [7:0]. Addresses use bits [13:5] and bits [4:0] are don't care. EXCOL can be read and written only in special modes and always returns 0 in normal modes.

EXROW — Select Extra Rows Bit

0 = User array selected
1 = User array is disabled and two extra rows are available. Addresses use bits [7:0] and bits [13:8] are don't care. EXROW can be read and written only in special modes and always returns 0 in normal modes.

T[1:0] — EPROM Test Mode Select Bits

These bits allow selection of either gate stress or drain stress test modes. They can be read and written only in special modes and always read 0 in normal modes.

T1	T0	Function Selected
0	0	Normal mode
0	1	Reserved
1	0	Gate stress
1	1	Drain stress

Any one of these interrupts can be assigned the highest maskable interrupt priority by writing the appropriate value to the PSEL bits in the HPRIO register. Otherwise, the priority arrangement remains the same. An interrupt that is assigned highest priority is still subject to global masking by the I bit in the CCR, or by any associated local bits. Interrupt vectors are not affected by priority assignment. To avoid race conditions, HPRIO can be written only while I-bit interrupts are inhibited.

5.4.1 Highest Priority Interrupt and Miscellaneous Register

Address: $103C

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	RBOOT(1)	SMOD(1)	MDA(1)	IRVNE	PSEL2	PSEL2	PSEL1	PSEL0
Reset:								
Single chip:	0	0	0	0	0	1	1	0
Expanded:	0	0	1	0	0	1	1	0
Bootstrap:	1	1	0	0	0	1	1	0
Special test:	0	1	1	1	0	1	1	0

1. The values of the RBOOT, SMOD, and MDA reset bits depend on the mode selected at the RESET pin rising edge. Refer to Table 2-1. Hardware Mode Select Summary.

Figure 5-4. Highest Priority I-Bit Interrupt and Miscellaneous Register (HPRIO)

RBOOT — Read Bootstrap ROM Bit

Has meaning only when the SMOD bit is a 1 (bootstrap mode or special test mode). At all other times this bit is clear and cannot be written. Refer to Chapter 2 Operating Modes and On-Chip Memory for more information.

SMOD — Special Mode Select Bit

This bit reflects the inverse of the MODB input pin at the rising edge of reset. Refer to Chapter 2 Operating Modes and On-Chip Memory for more information.

MDA — Mode Select A Bit

The mode select A bit reflects the status of the MODA input pin at the rising edge of reset. Refer to Chapter 2 Operating Modes and On-Chip Memory for more information.

IRVNE — Internal Read Visibility/Not E Bit

The IRVNE control bit allows internal read accesses to be available on the external data bus during operation in expanded modes. In single-chip and bootstrap modes, IRVNE determines whether the E clock is driven out an external pin. For the MC68HC811E2, this bit is IRV and only controls internal read visibility. Refer to Chapter 2 Operating Modes and On-Chip Memory for more information.

PSEL[3:0] — Priority Select Bits

These bits select one interrupt source to be elevated above all other I-bit-related sources and can be written only while the I bit in the CCR is set (interrupts disabled).

CWOM — Port C Wired-OR Mode Bit (affects all eight port C pins)

It is customary to have an external pullup resistor on lines that are driven by open-drain devices.

0 = Port C outputs are normal CMOS outputs.
1 = Port C outputs are open-drain outputs.

HNDS — Handshake Mode Bit

0 = Simple strobe mode
1 = Full input or output handshake mode

OIN — Output or Input Handshake Select Bit

HNDS must be set to 1 for this bit to have meaning.

0 = Input handshake
1 = Output handshake

PLS — Pulsed/Interlocked Handshake Operation Bit

HNDS must be set to 1 for this bit to have meaning. When interlocked handshake is selected, strobe B is active until the selected edge of strobe A is detected.

0 = Interlocked handshake
1 = Pulsed handshake (Strobe B pulses high for two E-clock cycles.)

EGA — Active Edge for Strobe A Bit

0 = STRA falling edge selected, high level activates port C outputs (output handshake)
1 = STRA rising edge selected, low level activates port C outputs (output handshake)

INVB — Invert Strobe B Bit

0 = Active level is logic 0.
1 = Active level is logic 1.

Table 7-1. Baud Rate Values

Prescaler Selects						Prescale Divide	Baud Set Divide	Crystal Frequency (MHz)					
								4.00	4.9152	8.00	10.00	12.00	16.00
								Bus Frequency (MHz)					
SCP2	SCP1	SCP0	SCR2	SCR1	SCR0			1.00	1.23	2.00	2.50	3.00	4.00
0	0	0	0	0	0	1	1	62500	76800	125000	156250	187500	250000
0	0	0	0	0	1	1	2	31250	38400	62500	78125	93750	125000
0	0	0	0	1	0	1	4	15625	19200	31250	39063	46875	62500
0	0	0	0	1	1	1	8	7813	9600	15625	19531	23438	31250
0	0	0	1	0	0	1	16	3906	4800	7813	9766	11719	15625
0	0	0	1	0	1	1	32	1953	2400	3906	4883	5859	7813
0	0	0	1	1	0	1	64	977	1200	1953	2441	2930	3906
0	0	0	1	1	1	1	128	488	600	977	1221	1465	1953
0	0	1	0	0	0	3	1	20833	25600	41667	52083	62500	83333
0	0	1	0	0	1	3	2	10417	12800	20833	26042	31250	41667
0	0	1	0	1	0	3	4	5208	6400	10417	13021	15625	20833
0	0	1	0	1	1	3	8	2604	3200	5208	6510	7813	10417
0	0	1	1	0	0	3	16	1302	1600	2604	3255	3906	5208
0	0	1	1	0	1	3	32	651	800	1302	1628	1953	2604
0	0	1	1	1	0	3	64	326	400	651	814	977	1302
0	0	1	1	1	1	3	128	163	200	326	407	488	651
0	1	0	0	0	0	4	1	15625	19200	31250	39063	46875	62500
0	1	0	0	0	1	4	2	7813	9600	15625	19531	23438	31250
0	1	0	0	1	0	4	4	3906	4800	7813	9766	11719	15625
0	1	0	0	1	1	4	8	1953	2400	3906	4883	5859	7813
0	1	0	1	0	0	4	16	977	1200	1953	2441	2930	3906
0	1	0	1	0	1	4	32	488	600	977	1221	1465	1953
0	1	0	1	1	0	4	64	244	300	488	610	732	977
0	1	0	1	1	1	4	128	122	150	244	305	366	488
0	1	1	0	0	0	13	1	4808	5908	9615	12019	14423	19231
0	1	1	0	0	1	13	2	2404	2954	4808	6010	7212	9615
0	1	1	0	1	0	13	4	1202	1477	2404	3005	3606	4808
0	1	1	0	1	1	13	8	601	738	1202	1502	1803	2404
0	1	1	1	0	0	13	16	300	369	601	751	901	1202
0	1	1	1	0	1	13	32	150	185	300	376	451	601
0	1	1	1	1	0	13	64	75	92	150	188	225	300
0	1	1	1	1	1	13	128	38	46	75	94	113	150
1	0	0	0	0	0	39	1	1603	1969	3205	4006	4808	6410
1	0	0	0	0	1	39	2	801	985	1603	2003	2404	3205
1	0	0	0	1	0	39	4	401	492	801	1002	1202	1603
1	0	0	0	1	1	39	8	200	246	401	501	601	801
1	0	0	1	0	0	39	16	100	123	200	250	300	401
1	0	0	1	0	1	39	32	50	62	100	125	150	200
1	0	0	1	1	0	39	64	25	31	50	63	75	100
1	0	0	1	1	1	39	128	13	15	25	31	38	50

Shaded areas reflect standard baud rates.
On MC68HC(7)11E20 do not set SCP1 or SCP0 when SCP2 is 1.

8.5.3 Serial Clock

SCK, an input to a slave device, is generated by the master device and synchronizes data movement in and out of the device through the MOSI and MISO lines. Master and slave devices are capable of exchanging a byte of information during a sequence of eight clock cycles.

Four possible timing relationships can be chosen by using control bits CPOL and CPHA in the serial peripheral control register (SPCR). Both master and slave devices must operate with the same timing. The SPI clock rate select bits, SPR[1:0], in the SPCR of the master device, select the clock rate. In a slave device, SPR[1:0] have no effect on the operation of the SPI.

8.5.4 Slave Select

The slave select ($\overline{SS}$) input of a slave device must be externally asserted before a master device can exchange data with the slave device. $\overline{SS}$ must be low before data transactions and must stay low for the duration of the transaction.

The $\overline{SS}$ line of the master must be held high. If it goes low, a mode fault error flag (MODF) is set in the serial peripheral status register (SPSR). To disable the mode fault circuit, write a 1 in bit 5 of the port D data direction register. This sets the $\overline{SS}$ pin to act as a general-purpose output rather than the dedicated input to the slave select circuit, thus inhibiting the mode fault flag. The other three lines are dedicated to the SPI whenever the serial peripheral interface is on.

The state of the master and slave CPHA bits affects the operation of $\overline{SS}$. CPHA settings should be identical for master and slave. When CPHA = 0, the shift clock is the OR of $\overline{SS}$ with SCK. In this clock phase mode, $\overline{SS}$ must go high between successive characters in an SPI message. When CPHA = 1, $\overline{SS}$ can be left low between successive SPI characters. In cases where there is only one SPI slave MCU, its $\overline{SS}$ line can be tied to V_{SS} as long as only CPHA = 1 clock mode is used.

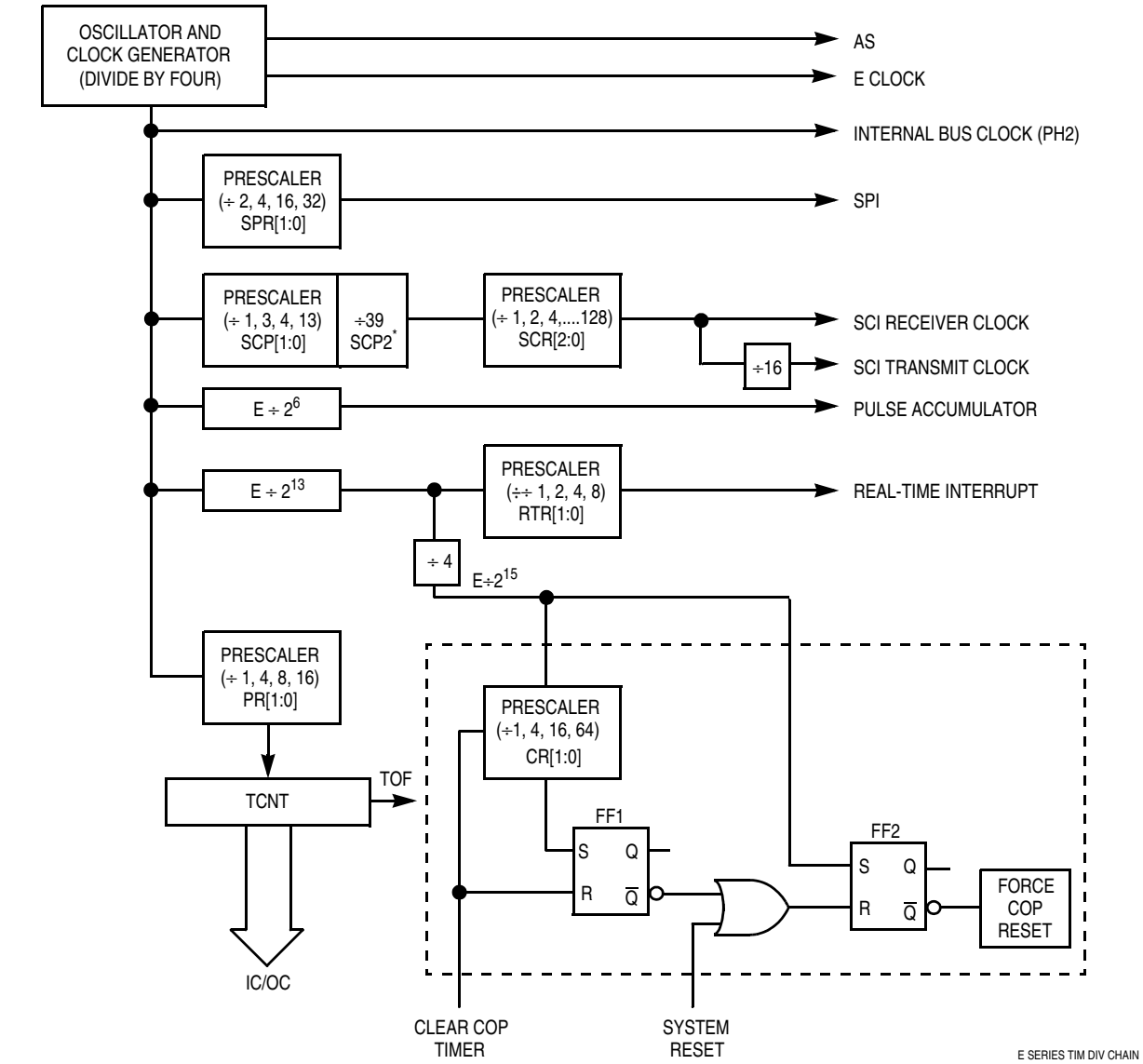
8.6 SPI System Errors

Two system errors can be detected by the SPI system. The first type of error arises in a multiple-master system when more than one SPI device simultaneously tries to be a master. This error is called a mode fault. The second type of error, write collision, indicates that an attempt was made to write data to the SPDR while a transfer was in progress.

When the SPI system is configured as a master and the $\overline{SS}$ input line goes to active low, a mode fault error has occurred — usually because two devices have attempted to act as master at the same time. In cases where more than one device is concurrently configured as a master, there is a chance of contention between two pin drivers. For push-pull CMOS drivers, this contention can cause permanent damage. The mode fault mechanism attempts to protect the device by disabling the drivers. The MSTR control bit in the SPCR and all four DDRD control bits associated with the SPI are cleared and an interrupt is generated subject to masking by the SPIE control bit and the I bit in the CCR.

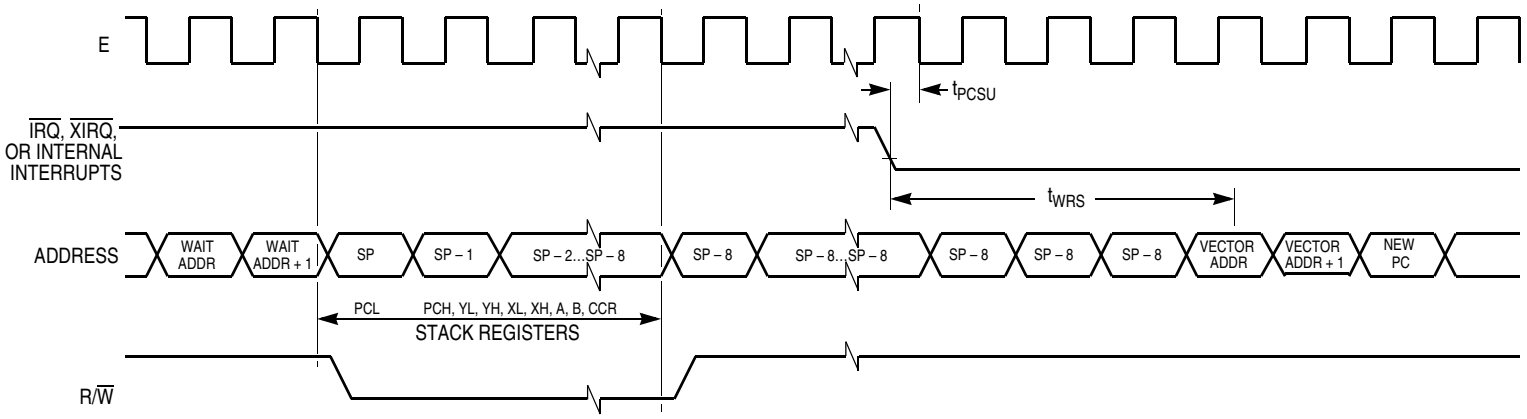
Other precautions may need to be taken to prevent driver damage. If two devices are made masters at the same time, mode fault does not help protect either one unless one of them selects the other as slave. The amount of damage possible depends on the length of time both devices attempt to act as master.

A write collision error occurs if the SPDR is written while a transfer is in progress. Because the SPDR is not double buffered in the transmit direction, writes to SPDR cause data to be written directly into the SPI shift register. Because this write corrupts any transfer in progress, a write collision error is generated. The transfer continues undisturbed, and the write data that caused the error is not written to the shifter.



* SCP2 present on MC68HC(7)11E20 only

Figure 9-1. Timer Clock Divider Chains



Note: RESET also causes recovery from WAIT.

Figure 10-5. WAIT Recovery from Interrupt Timing Diagram

Description	CONFIG	Temperature	Frequency	MC Order Number
52-pin plastic leaded chip carrier (PLCC) (Continued)				
OTPROM	$0F	–40°C to +85°C	2 MHz	MC68HC711E9CFN2
			3 MHz	MC68HC711E9CFN3
		–40°C to +105°C	2 MHz	MC68HC711E9VFN2
		–40°C to +125°C	2 MHz	MC68HC711E9MFN2
OTPROM, enhanced security feature	$0F	–40°C to +85°C	2 MHz	MC68S711E9CFN2
20 Kbytes OTPROM	$0F	0°C to +70°C	3 MHz	MC68HC711E20FN3
		–40°C to +85°C	2 MHz	MC68HC711E20CFN2
			3 MHz	MC68HC711E20CFN3
		–40°C to +105°C	2 MHz	MC68HC711E20VFN2
		–40°C to +125°C	2 MHz	MC68HC711E20MFN2
No ROM, 2 Kbytes EEPROM	$FF	0°C to +70°C	2 MHz	MC68HC811E2FN2
		–40°C to +85°C	2 MHz	MC68HC811E2CFN2
		–40°C to +105°C	2 MHz	MC68HC811E2VFN2
		–40°C to +125°C	2 MHz	MC68HC811E2MFN2
64-pin quad flat pack (QFP)				
BUFFALO ROM	$0F	–40°C to +85°C	2 MHz	MC68HC11E9BCFU2
			3 MHz	MC68HC11E9BCFU3
No ROM	$0D	–40°C to +85°C	2 MHz	MC68HC11E1CFU2
			3 MHz	MC68HC11E1CFU3
		–40°C to +105°C	2 MHz	MC68HC11E1VFU2
No ROM, no EEPROM	$0C	–40°C to +85°C	2 MHz	MC68HC11E0CFU2
		–40°C to +105°C	2 MHz	MC68HC11E0VFU2
20 Kbytes OTPROM	$0F	0°C to +70°°C	3 MHz	MC68HC711E20FU3
		–40°C to +85°C	2 MHz	MC68HC711E20CFU2
			3 MHz	MC68HC711E20CFU3
		–40°C to +105°C	2 MHz	MC68HC711E20VFU2
		–40°C to +125°C	2 MHz	MC68HC711E20MFU2
52-pin thin quad flat pack (TQFP)				
BUFFALO ROM	$0F	–40°C to +85°C	2 MHz	MC68HC11E9BCPB2
			3 MHz	MC68HC11E9BCPB3

Bootstrap mode can also be used to interactively calibrate critical analog sensors. Since this calibration is done in the final assembled system, it can compensate for any errors in discrete interface circuitry and cabling between the sensor and the analog inputs to the MCU. Note that this calibration routine is a downloaded program that does not take up space in the normal application program.

Bootstrap Mode Logic

In the M68HC11 MCUs, very little logic is dedicated to the bootstrap mode. Consequently, this mode adds almost no extra cost to the MCU system. The biggest piece of circuitry for bootstrap mode is the small boot ROM. This ROM is 192 bytes in the original MC68HC11A8, but some of the newest members of the M68HC11 Family, such as the MC68HC711K4, have as much as 448 bytes to accommodate added features. Normally, this boot ROM is present in the memory map only when the MCU is reset in bootstrap mode to prevent interference with the user's normal memory space. The enable for this ROM is controlled by the read boot ROM (RBOOT) control bit in the highest priority interrupt (HPRIO) register. The RBOOT bit can be written by software whenever the MCU is in special test or special bootstrap modes; when the MCU is in normal modes, RBOOT reverts to 0 and becomes a read-only bit. All other logic in the MCU would be present whether or not there was a bootstrap mode.

Figure 1 shows the composite memory map of the MC68HC711E9 in its four basic modes of operation, including bootstrap mode. The active mode is determined by the mode A (MDA) and special mode (SMOD) control bits in the HPRIO control register. These control bits are in turn controlled by the state of the mode A (MODA) and mode B (MODB) pins during reset. Table 1 shows the relationship between the state of these pins during reset, the selected mode, and the state of the MDA, SMOD, and RBOOT control bits. Refer to the composite memory map and information in Table 1 for the following discussion.

The MDA control bit is determined by the state of the MODA pin as the MCU leaves reset. MDA selects between single-chip and expanded operating modes. When MDA is 0, a single-chip mode is selected, either normal single-chip mode or special bootstrap mode. When MDA is 1, an expanded mode is selected, either normal expanded mode or special test mode.

The SMOD control bit is determined by the inverted state of the MODB pin as the MCU leaves reset. SMOD controls whether a normal mode or a special mode is selected. When SMOD is 0, one of the two normal modes is selected, either normal single-chip mode or normal expanded mode. When SMOD is 1, one of the two special modes is selected, either special bootstrap mode or special test mode. When either special mode is in effect (SMOD = 1), certain privileges are in effect, for instance, the ability to write to the mode control bits and fetching the reset and interrupt vectors from $BFxx rather than $FFxx.

Table 1. Mode Selection Summary

Input Pins		Mode Selected	Control Bits in HPRIO		
MODB	MODA		RBOOT	SMOD	MDA
1	0	Normal single chip	0	0	0
0	0	Normal expanded	0	0	1
0	0	Special bootstrap	1	1	0
0	1	Special test	0	1	1

The alternate vector locations are achieved by simply driving address bit A14 low during all vector fetches if SMOD = 1. For special test mode, the alternate vector locations assure that the reset vector can be fetched from external memory space so the test system can control MCU operation. In special bootstrap mode, the small boot ROM is enabled in the memory map by RBOOT = 1 so the reset vector will be fetched from this ROM and the bootloader firmware will control MCU operation.

RBOOT is reset to 1 in bootstrap mode to enable the small boot ROM. In the other three modes, RBOOT is reset to 0 to keep the boot ROM out of the memory map. While in special test mode, SMOD = 1, which allows the RBOOT control bit to be written to 1 by software to enable the boot ROM for testing purposes.

Boot ROM Firmware

The main program in the boot ROM is the bootloader, which is automatically executed as a result of resetting the MCU in bootstrap mode. Some newer versions of the M68HC11 Family have additional utility programs that can be called from a downloaded program. One utility is available to program EPROM or OTP versions of the M68HC11. A second utility allows the contents of memory locations to be uploaded to a host computer. In the MC68HC711K4 boot ROM, a section of code is used by Freescale for stress testing the on-chip EEPROM. These test and utility programs are similar to self-test ROM programs in other MCUs except that the boot ROM does not use valuable space in the normal memory map.

Bootstrap firmware is also involved in an optional EEPROM security function on some versions of the M68HC11. This EEPROM security feature prevents a software pirate from seeing what is in the on-chip EEPROM. The secured state is invoked by programming the no security (NOSEC) EEPROM bit in the CONFIG register. Once this NOSEC bit is programmed to 0, the MCU will ignore the mode A pin and always come out of reset in normal single-chip mode or special bootstrap mode, depending on the state of the mode B pin. Normal single-chip mode is the usual way a secured part would be used. Special bootstrap mode is used to disengage the security function (only after the contents of EEPROM and RAM have been erased). Refer to the *M68HC11 Reference Manual*, Freescale document order number M68HC11RM/AD, for additional information on the security mode and complete listings of the boot ROMs that support the EEPROM security functions.

Automatic Selection of Baud Rate

The bootloader program in the MC68HC711E9 accommodates either of two baud rates.

- The higher of these baud rates (7812 baud at a 2-MHz E-clock rate) is used in systems that operate from a binary frequency crystal such as 2^{23} Hz (8.389 MHz). At this crystal frequency, the baud rate is 8192 baud, which was used extensively in automotive applications.
- The second baud rate available to the M68HC11 bootloader is 1200 baud at a 2-MHz E-clock rate. Some of the newest versions of the M68HC11, including the MC68HC11F1 and MC68HC117K4, accommodate other baud rates using the same differentiation technique explained here. Refer to the reference numbers in square brackets in Figure 2 during the following explanation.

NOTE

Software can change some aspects of the memory map after reset.

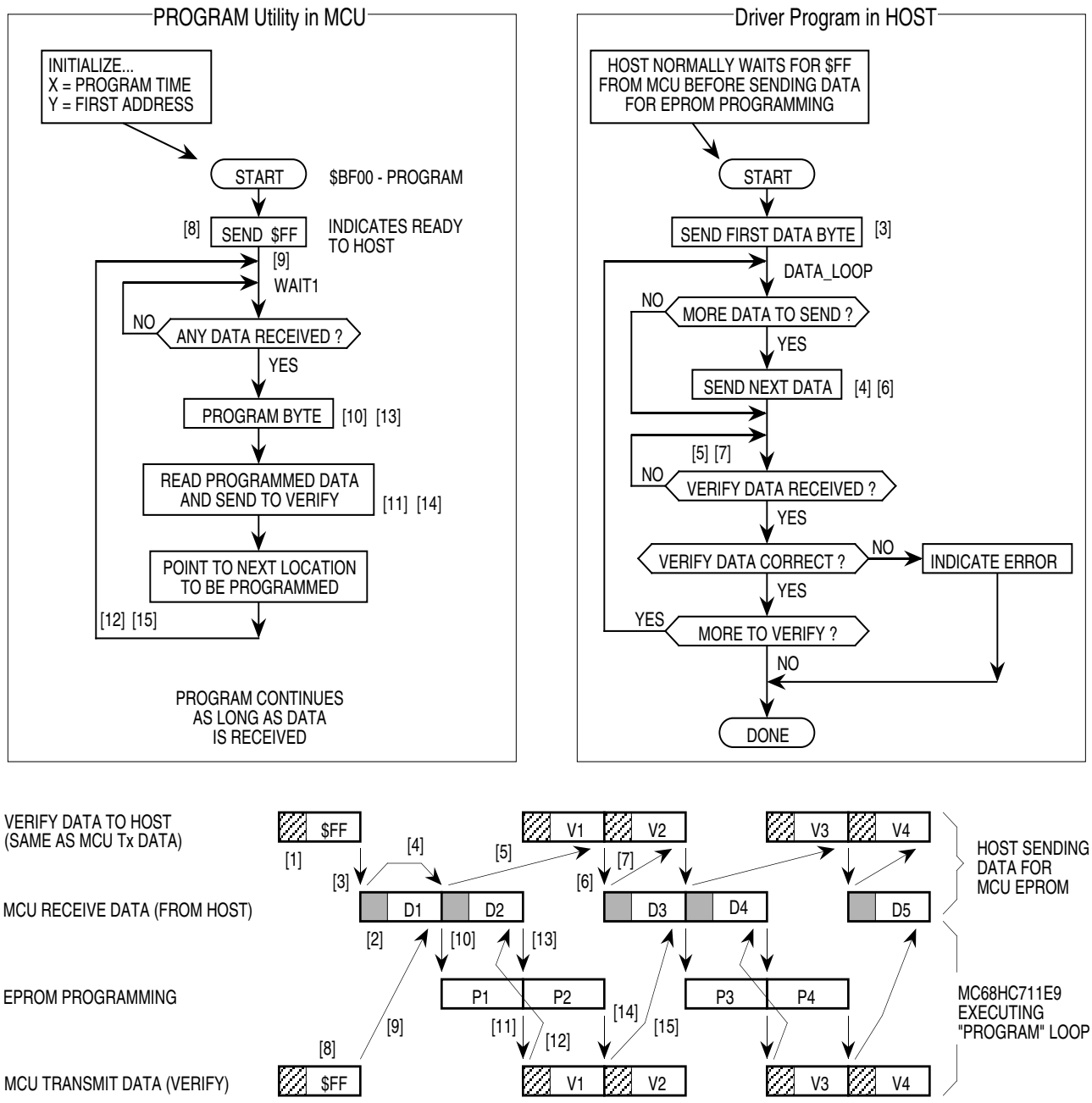


Figure 4. Host and MCU Activity during EPROM PROGRAM Utility

After the MCU sends $FF [8], it enters the WAIT1 loop [9] and waits for the first data character from the host. When this character is received [10], the MCU programs it into the address pointed to by the Y index register. When the programming time delay is over, the MCU reads the programmed data, transmits it to the host for verification [11], and returns to the top of the WAIT1 loop to wait for the next data character [12]. Because the host previously sent the second data character, it is already waiting in the SCI receiver of the MCU. Steps [13], [14], and [15] correspond to the second pass through the WAIT1 loop.

Back in the host, the first verify character is received, and the third data character is sent [6]. The host then waits for the second verify character [7] to come back from the MCU. The sequence continues as long as the host continues to send data to the MCU. Since the WAIT1 loop in the PROGRAM utility is an indefinite loop, reset is used to end the process in the MCU after the host has finished sending data to be programmed.

Allowing for Bootstrap Mode

Since bootstrap mode requires few connections to the MCU, it is easy to design systems that accommodate bootstrap mode.

Bootstrap mode is useful for diagnosing or repairing systems that have failed due to changes in the CONFIG register or failures of the expansion address/data buses, (rendering programs in external memory useless). Bootstrap mode can also be used to load information into the EPROM or EEPROM of an M68HC11 after final assembly of a module. Bootstrap mode is also useful for performing system checks and calibration routines. The following paragraphs explain system requirements for use of bootstrap mode in a product.

Mode Select Pins

It must be possible to force the MODA and MODB pins to logic 0, which implies that these two pins should be pulled up to V_{DD} through resistors rather than being tied directly to V_{DD}. If mode pins are connected directly to V_{DD}, it is not possible to force a mode other than the one the MCU is hard wired for. It is also good practice to use pulldown resistors to V_{SS} rather than connecting mode pins directly to V_{SS} because it is sometimes a useful debug aid to attempt reset in modes other than the one the system was primarily designed for. Physically, this requirement sometimes calls for the addition of a test point or a wire connected to one or both mode pins. Mode selection only uses the mode pins while $\overline{\text{RESET}}$ is active.

$\overline{\text{RESET}}$

It must be possible to initiate a reset while the mode select pins are held low. In systems where there is no provision for manual reset, it is usually possible to generate a reset by turning power off and back on.

RxD Pin

It must be possible to drive the PD0/RxD pin with serial data from a host computer (or another MCU). In many systems, this pin is already used for SCI communications; thus no changes are required.

```
              Errors: None
              Labels: 28
Last Program Address: $B67C
Last Storage Address: $0000
       Program Bytes: $007D  125
       Storage Bytes: $0000  0
```

Driving Boot Mode from a Personal Computer

In this example, a personal computer is used as the host to drive the bootloader of an MC68HC711E9. An M68HC11 EVBU is used for the target MC68HC711E9. A large program is transferred from the personal computer into the EPROM of the target MC68HC711E9.

Hardware

Figure 7 shows a small modification to the EVBU to accommodate the 12-volt (nominal) EPROM programming voltage. The $\overline{\text{XIRQ}}$ pin is connected to a pullup resistor, two jumpers, and the 60-pin connectors, P4 and P5. The object of the modification is to isolate the $\overline{\text{XIRQ}}$ pin and then connect it to the programming power supply. Carefully cut the trace on the solder side of the EVBU as indicated in Figure 7. This disconnects the pullup resistor RN1 D from $\overline{\text{XIRQ}}$ but leaves P4–18, P5–18, and jumpers J7 and J14 connected so the EVBU can still be used for other purposes after programming is done. Remove any fabricated jumpers from J7 and J14. The EVBU normally has a jumper at J7 to support the trace function

Figure 8 shows a small circuit that is added to the wire-wrap area of the EVBU. The 3-terminal jumper allows the $\overline{\text{XIRQ}}$ line to be connected to either the programming power supply or to a substitute pullup resistor for $\overline{\text{XIRQ}}$. The 100-ohm resistor is a current limiter to protect the 12-volt input of the MCU. The resistor and LED connected to P5 pin 9 (port C bit 0) is an optional indicator that lights when programming is complete.

Software

BASIC was chosen as the programming language due to its readability and availability in parallel versions on both the IBM® PC and the Macintosh®. The program demonstrates several programming techniques for use with an M68HC11 and is not necessarily intended to be a finished, commercial program. For example, there is little error checking, and the user interface is elementary. A complete listing of the BASIC program is included in Listing 2. BASIC Program for Personal Computer with moderate comments. The following paragraphs include a more detailed discussion of the program as it pertains to communicating with and programming the target MC68HC711E9. Lines 25–45 initialize and define the variables and array used in the program. Changes to this section would allow for other programs to be downloaded.

® IBM is a registered trademark of International Business Machines.
® Macintosh is a registered trademark of Apple Computers, Inc.

```
8491 '*         DECIMAL TO HEX CONVERSION
8492 '*                 INPUT:  K - INTEGER TO BE CONVERTED
8493 '*                 OUTPUT: HX$ - TWO CHARACTER STRING WITH HEX CONVERSION
8494 '*************************************************************************
8500 IF K > 255 THEN HX$="Too big":GOTO 8530
8510 HX$=MID$(H$,K\16+1,1)            'UPPER NIBBLE
8520 HX$=HX$+MID$(H$,(K MOD 16)+1,1)  'LOWER NIBBLE
8530 RETURN
9499 '******************** BOOT CODE ***************************************
9500 DATA 86, 23                 'LDAA   #$23
9510 DATA B7, 10, 02             'STAA   OPT2    make port C wire or
9520 DATA 86, FE                 'LDAA   #$FE
9530 DATA B7, 10, 03             'STAA   PORTC   light 1 LED on port C bit 0
9540 DATA C6, FF                 'LDAB   #$FF
9550 DATA F7, 10, 07             'STAB   DDRC    make port C outputs
9560 DATA CE, 0F, A0             'LDX    #4000   2msec at 2MHz
9570 DATA 18, CE, E0, 00         'LDY    #$E000  Start of BUFFALO 3.4
9580 DATA 7E, BF, 00             'JMP    $BF00   EPROM routine start address
9590 '*************************************************************************
```

Common Bootstrap Mode Problems

It is not unusual for a user to encounter problems with bootstrap mode because it is new to many users. By knowing some of the common difficulties, the user can avoid them or at least recognize and quickly correct them.

Reset Conditions vs. Conditions as Bootloaded Program Starts

It is common to confuse the reset state of systems and control bits with the state of these systems and control bits when a bootloaded program in RAM starts.

Between these times, the bootloader program is executed, which changes the states of some systems and control bits:

- The SCI system is initialized and turned on (Rx and Tx).
- The SCI system has control of the PD0 and PD1 pins.
- Port D outputs are configured for wire-OR operation.
- The stack pointer is initialized to the top of RAM.
- Time has passed (two or more SCI character times).
- Timer has advanced from its reset count value.

Users also forget that bootstrap mode is a special mode. Thus, privileged control bits are accessible, and write protection for some registers is not in effect. The bootstrap ROM is in the memory map. The DISR bit in the TEST1 control register is set, which disables resets from the COP and clock monitor systems.

Since bootstrap is a special mode, these conditions can be changed by software. The bus can even be switched from single-chip mode to expanded mode to gain access to external memories and peripherals.

Listing 3. MC68HC711E9 Bootloader ROM

```
213
214                 ***************************************************
215                 * Boot ROM revision level in ASCII
216                 *       (ORG     $BFD1)
217 BFD1 41                FCC     "A"
218                 ***************************************************
219                 * Mask set I.D. ($0000 FOR EPROM PARTS)
220                 *       (ORG     $BFD2)
221 BFD2 0000                FDB     $0000
222                 ***************************************************
223                 * '711E9 I.D. - Can be used to determine MCU type
224                 *       (ORG     $BFD4)
225 BFD4 71E9                FDB     $71E9
226
227                 ***************************************************
228                 * VECTORS - point to RAM for pseudo-vector JUMPs
229
230 BFD6 00C4                 FDB     $100-60         SCI
231 BFD8 00C7                 FDB     $100-57         SPI
232 BFDA 00CA                 FDB     $100-54         PULSE ACCUM INPUT EDGE
233 BFDC 00CD                 FDB     $100-51         PULSE ACCUM OVERFLOW
234 BFDE 00D0                 FDB     $100-48         TIMER OVERFLOW
235 BFE0 00D3                 FDB     $100-45         TIMER OUTPUT COMPARE 5
236 BFE2 00D6                 FDB     $100-42         TIMER OUTPUT COMPARE 4
237 BFE4 00D9                 FDB     $100-39         TIMER OUTPUT COMPARE 3
238 BFE6 00DC                 FDB     $100-36         TIMER OUTPUT COMPARE 2
239 BFE8 00DF                 FDB     $100-33         TIMER OUTPUT COMPARE 1
240 BFEA 00E2                 FDB     $100-30         TIMER INPUT CAPTURE 3
241 BFEC 00E5                 FDB     $100-27         TIMER INPUT CAPTURE 2
242 BFEE 00E8                 FDB     $100-24         TIMER INPUT CAPTURE 1
243 BFF0 00EB                 FDB     $100-21         REAL TIME INT
244 BFF2 00EE                 FDB     $100-18         IRQ
245 BFF4 00F1                 FDB     $100-15         XIRQ
246 BFF6 00F4                 FDB     $100-12         SWI
247 BFF8 00F7                 FDB     $100-9          ILLEGAL OP-CODE
248 BFFA 00FA                 FDB     $100-6          COP FAIL
249 BFFC 00FD                 FDB     $100-3          CLOCK MONITOR
250 BFFE BF54                 FDB     BEGIN           RESET
251 C000                      END
Symbol Table:

Symbol Name     Value  Def.#  Line Number Cross Reference

BAUD             002B *00037  00160 00180
BAUDOK           BF8A *00183  00178
BEGIN            BF54 *00155  00250
DELAYF           021B *00061  00163
DELAYS           0DB0 *00060  00181
DONEIT           BF47 *00142  00124
EEPMEND          B7FF *00050
EEPMSTR          B600 *00049  00175
ELAT             0020 *00043  00125 00128
EPGM             0001 *00044  00128
EPRMEND          FFFF *00053
EPRMSTR          D000 *00052  00206
```

Step 5

The CONFIG register defaults to hexadecimal 103F on the MC68HC711E9. PCBUG11 needs adressing parameters to allow programming of a specific block of memory so the following parameter must be given.

At the PCbug11 command prompt, type: EEPROM 0.

Then type: EEPROM 103F 103F.

Step 6

Erase the CONFIG to allow byte programming.

At the PCbug11 command prompt, type: EEPROM ERASE BULK 103F.

Step 7

You are now ready to download the program into the EEPROM and EPROM.

At the PCbug11command prompt, type: LOADSC:\MYPROG\MYPROG.S19.

For more details on programming the EPROM, read the engineering bulletin *Programming MC68HC711E9 Devices with PCbug11 and the M68HC11EVB*, Freescale document number EB187.

Step 8

You are now ready to enable the security feature on the MCHC711E9.

At the PCbug11 command prompt type: MS 103F 05.

Step 9

After the programming operation is complete, verifyng the CONFIG on the MCHC711E9 is not possible because in bootstrap mode the default value is always forced.

Step 10

The part is now in secure mode and whatever code you loaded into EEPROM will be erased if you tried to bring the microcontroller up in either expanded mode or bootstrap mode.

NOTE

It is important to note that the microcontroller will work properly in secure mode only in single chip mode.

NOTE

If the part is placed in bootstrap or expanded, the code in EEPROM and RAM will be erased and the microcontroller cannot be reused. The security software will constantly read the NOSEC bit and lock the part.