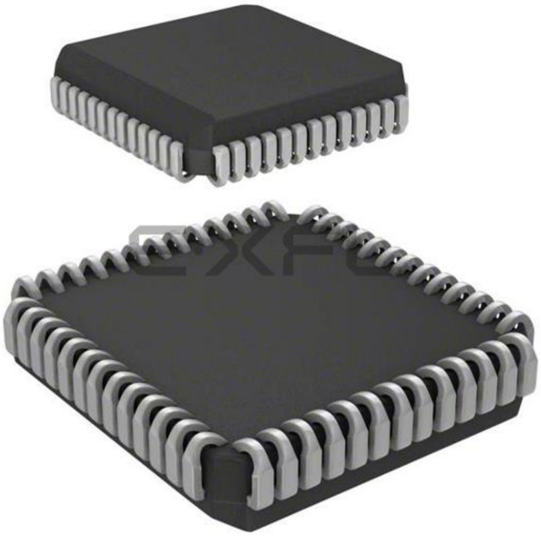
NXP USA Inc. - MC68S711E9CFNE2 Datasheet



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	12KB (12K x 8)
Program Memory Type	OTP
EEPROM Size	512 x 8
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 5.5V
Data Converters	A/D 8x8b
Oscillator Type	External
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/mc68s711e9cfne2

Chapter 5
Resets and Interrupts

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)

Address: $103F

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	EE3	EE2	EE1	EE0	NOSEC	NOCOP	▨	EEON
Resets:								
Single chip:	1	1	1	1	U	U	1	1
Bootstrap:	1	1	1	1	U	U(L)	1	1
Expanded:	U	U	U	U	1	U	1	U
Test:	U	U	U	U	1	U(L)	1	0

▨ = Unimplemented

U indicates a previously programmed bit. U(L) indicates that the bit resets to the logic level held in the latch prior to reset, but the function of COP is controlled by the DISR bit in TEST1 register.

Figure 2-11. MC68HC811E2 System Configuration Register (CONFIG)

EE[3:0] — EEPROM Mapping Bits

EE[3:0] apply only to MC68HC811E2 and allow the 2048 bytes of EEPROM to be remapped to any 4-Kbyte boundary. See Table 2-3.

Table 2-3. EEPROM Mapping

EE[3:0]	EEPROM Location
0 0 0 0	$0800–$0FFF
0 0 0 1	$1800–$1FFF
0 0 1 0	$2800–$2FFF
0 0 1 1	$3800–$3FFF
0 1 0 0	$4800–$4FFF
0 1 0 1	$5800–$5FFF
0 1 1 0	$6800–$6FFF
0 1 1 1	$7800–$7FFF
1 0 0 0	$8800–$8FFF
1 0 0 1	$9800–$9FFF
1 0 1 0	$A800–$AFFF
1 0 1 1	$B800–$BFFF
1 1 0 0	$C800–$CFFF
1 1 0 1	$D800–$DFFF
1 1 1 0	$E800–$EFFF
1 1 1 1	$F800–$FFFF

4.2.6.5 Interrupt Mask (I)

The interrupt request (IRQ) mask (I bit) is a global mask that disables all maskable interrupt sources. While the I bit is set, interrupts can become pending, but the operation of the CPU continues uninterrupted until the I bit is cleared. After any reset, the I bit is set by default and can only be cleared by a software instruction. When an interrupt is recognized, the I bit is set after the registers are stacked, but before the interrupt vector is fetched. After the interrupt has been serviced, a return-from-interrupt instruction is normally executed, restoring the registers to the values that were present before the interrupt occurred. Normally, the I bit is 0 after a return from interrupt is executed. Although the I bit can be cleared within an interrupt service routine, "nesting" interrupts in this way should only be done when there is a clear understanding of latency and of the arbitration mechanism. Refer to Chapter 5 Resets and Interrupts.

4.2.6.6 Half Carry (H)

The H bit is set when a carry occurs between bits 3 and 4 of the arithmetic logic unit during an ADD, ABA, or ADC instruction. Otherwise, the H bit is cleared. Half carry is used during BCD operations.

4.2.6.7 X Interrupt Mask (X)

The XIRQ mask (X) bit disables interrupts from the $\overline{\text{XIRQ}}$ pin. After any reset, X is set by default and must be cleared by a software instruction. When an $\overline{\text{XIRQ}}$ interrupt is recognized, the X and I bits are set after the registers are stacked, but before the interrupt vector is fetched. After the interrupt has been serviced, an RTI instruction is normally executed, causing the registers to be restored to the values that were present before the interrupt occurred. The X interrupt mask bit is set only by hardware ($\overline{\text{RESET}}$ or $\overline{\text{XIRQ}}$ acknowledge). X is cleared only by program instruction (TAP, where the associated bit of A is 0; or RTI, where bit 6 of the value loaded into the CCR from the stack has been cleared). There is no hardware action for clearing X.

4.2.6.8 STOP Disable (S)

Setting the STOP disable (S) bit prevents the STOP instruction from putting the M68HC11 into a low-power stop condition. If the STOP instruction is encountered by the CPU while the S bit is set, it is treated as a no-operation (NOP) instruction, and processing continues to the next instruction. S is set by reset; STOP is disabled by default.

4.3 Data Types

The M68HC11 CPU supports four data types:

1. Bit data
2. 8-bit and 16-bit signed and unsigned integers
3. 16-bit unsigned fractions
4. 16-bit addresses

A byte is eight bits wide and can be accessed at any byte location. A word is composed of two consecutive bytes with the most significant byte at the lower value address. Because the M68HC11 is an 8-bit CPU, there are no special requirements for alignment of instructions or operands.

Table 4-2. Instruction Set (Sheet 2 of 7)

Mnemonic	Operation	Description	Addressing Mode		Instruction				Condition Codes							
					Opcode		Operand	Cycles	S	X	H	I	N	Z	V	C
BGT (rel)	Branch if > Zero	? Z + (N ⊕ V) = 0		REL		2E	rr	3	—	—	—	—	—	—	—	—
BHI (rel)	Branch if Higher	? C + Z = 0		REL		22	rr	3	—	—	—	—	—	—	—	—
BHS (rel)	Branch if Higher or Same	? C = 0		REL		24	rr	3	—	—	—	—	—	—	—	—
BITA (opr)	Bit(s) Test A with Memory	A • M	A A A A A	IMM DIR EXT IND,X IND,Y	 18	85 95 B5 A5 A5	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	0	—
BITB (opr)	Bit(s) Test B with Memory	B • M	B B B B B	IMM DIR EXT IND,X IND,Y	 18	C5 D5 F5 E5 E5	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	0	—
BLE (rel)	Branch if Δ Zero	? Z + (N ⊕ V) = 1		REL		2F	rr	3	—	—	—	—	—	—	—	—
BLO (rel)	Branch if Lower	? C = 1		REL		25	rr	3	—	—	—	—	—	—	—	—
BLS (rel)	Branch if Lower or Same	? C + Z = 1		REL		23	rr	3	—	—	—	—	—	—	—	—
BLT (rel)	Branch if < Zero	? N ⊕ V = 1		REL		2D	rr	3	—	—	—	—	—	—	—	—
BMI (rel)	Branch if Minus	? N = 1		REL		2B	rr	3	—	—	—	—	—	—	—	—
BNE (rel)	Branch if not = Zero	? Z = 0		REL		26	rr	3	—	—	—	—	—	—	—	—
BPL (rel)	Branch if Plus	? N = 0		REL		2A	rr	3	—	—	—	—	—	—	—	—
BRA (rel)	Branch Always	? 1 = 1		REL		20	rr	3	—	—	—	—	—	—	—	—
BRCLR(opr) (msk) (rel)	Branch if Bit(s) Clear	? M • mm = 0		DIR IND,X IND,Y	 18	13 1F 1F	dd mm rr ff mm rr ff mm rr	6 7 8	—	—	—	—	—	—	—	—
BRN (rel)	Branch Never	? 1 = 0		REL		21	rr	3	—	—	—	—	—	—	—	—
BRSET(opr) (msk) (rel)	Branch if Bit(s) Set	? (M̄) • mm = 0		DIR IND,X IND,Y	 18	12 1E 1E	dd mm rr ff mm rr ff mm rr	6 7 8	—	—	—	—	—	—	—	—
BSET (opr) (msk)	Set Bit(s)	M + mm ⇒ M		DIR IND,X IND,Y	 18	14 1C 1C	dd mm ff mm ff mm	6 7 8	—	—	—	—	Δ	Δ	0	—
BSR (rel)	Branch to Subroutine	See Figure 3–2		REL		8D	rr	6	—	—	—	—	—	—	—	—
BVC (rel)	Branch if Overflow Clear	? V = 0		REL		28	rr	3	—	—	—	—	—	—	—	—
BVS (rel)	Branch if Overflow Set	? V = 1		REL		29	rr	3	—	—	—	—	—	—	—	—
CBA	Compare A to B	A – B		INH		11	—	2	—	—	—	—	Δ	Δ	Δ	Δ
CLC	Clear Carry Bit	0 ⇒ C		INH		0C	—	2	—	—	—	—	—	—	—	0
CLI	Clear Interrupt Mask	0 ⇒ I		INH		0E	—	2	—	—	—	0	—	—	—	—
CLR (opr)	Clear Memory Byte	0 ⇒ M		EXT IND,X IND,Y	 18	7F 6F 6F	hh ll ff ff	6 6 7	—	—	—	—	0	1	0	0
CLRA	Clear Accumulator A	0 ⇒ A	A	INH		4F	—	2	—	—	—	—	0	1	0	0
CLRB	Clear Accumulator B	0 ⇒ B	B	INH		5F	—	2	—	—	—	—	0	1	0	0
CLV	Clear Overflow Flag	0 ⇒ V		INH		0A	—	2	—	—	—	—	—	—	0	—
CMPA (opr)	Compare A to Memory	A – M	A A A A A	IMM DIR EXT IND,X IND,Y	 18	81 91 B1 A1 A1	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	Δ	Δ

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).

7.7.3 Serial Communications Control Register 2

The SCCR2 register provides the control bits that enable or disable individual SCI functions.

Address: $102D

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Reset:	0	0	0	0	0	0	0	0

Figure 7-5. Serial Communications Control Register 2 (SCCR2)

TIE — Transmit Interrupt Enable Bit

0 = TDRE interrupts disabled
1 = SCI interrupt requested when TDRE status flag is set

TCIE — Transmit Complete Interrupt Enable Bit

0 = TC interrupts disabled
1 = SCI interrupt requested when TC status flag is set

RIE — Receiver Interrupt Enable Bit

0 = RDRF and OR interrupts disabled
1 = SCI interrupt requested when RDRF flag or the OR status flag is set

ILIE — Idle-Line Interrupt Enable Bit

0 = IDLE interrupts disabled
1 = SCI interrupt requested when IDLE status flag is set

TE — Transmitter Enable Bit

When TE goes from 0 to 1, one unit of idle character time (logic 1) is queued as a preamble.

0 = Transmitter disabled
1 = Transmitter enabled

RE — Receiver Enable Bit

0 = Receiver disabled
1 = Receiver enabled

RWU — Receiver Wakeup Control Bit

0 = Normal SCI receiver
1 = Wakeup enabled and receiver interrupts inhibited

SBK — Send Break

At least one character time of break is queued and sent each time SBK is written to 1. As long as the SBK bit is set, break characters are queued and sent. More than one break may be sent if the transmitter is idle at the time the SBK bit is toggled on and off, as the baud rate clock edge could occur between writing the 1 and writing the 0 to SBK.

0 = Break generator off
1 = Break codes generated

7.7.4 Serial Communication Status Register

The SCSR provides inputs to the interrupt logic circuits for generation of the SCI system interrupt.

Address: $102E

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	TDRE	TC	RDRF	IDLE	OR	NF	FE	▨
Reset:	1	1	0	0	0	0	0	0

▨ = Unimplemented

Figure 7-6. Serial Communications Status Register (SCSR)

TDRE — Transmit Data Register Empty Flag

This flag is set when SCDR is empty. Clear the TDRE flag by reading SCSR with TDRE set and then writing to SCDR.

0 = SCDR busy
0 = SCDR empty

TC — Transmit Complete Flag

This flag is set when the transmitter is idle (no data, preamble, or break transmission in progress). Clear the TC flag by reading SCSR with TC set and then writing to SCDR.

0 = Transmitter busy
1 = Transmitter idle

RDRF — Receive Data Register Full Flag

This flag is set if a received character is ready to be read from SCDR. Clear the RDRF flag by reading SCSR with RDRF set and then reading SCDR.

0 = SCDR empty
1 = SCDR full

IDLE — Idle Line Detected Flag

This flag is set if the RxD line is idle. Once cleared, IDLE is not set again until the RxD line has been active and becomes idle again. The IDLE flag is inhibited when RWU = 1. Clear IDLE by reading SCSR with IDLE set and then reading SCDR.

0 = RxD line active
1 = RxD line idle

OR — Overrun Error Flag

OR is set if a new character is received before a previously received character is read from SCDR. Clear the OR flag by reading SCSR with OR set and then reading SCDR.

0 = No overrun
1 = Overrun detected

NF — Noise Error Flag

NF is set if majority sample logic detects anything other than a unanimous decision. Clear NF by reading SCSR with NF set and then reading SCDR.

0 = Unanimous decision
1 = Noise detected

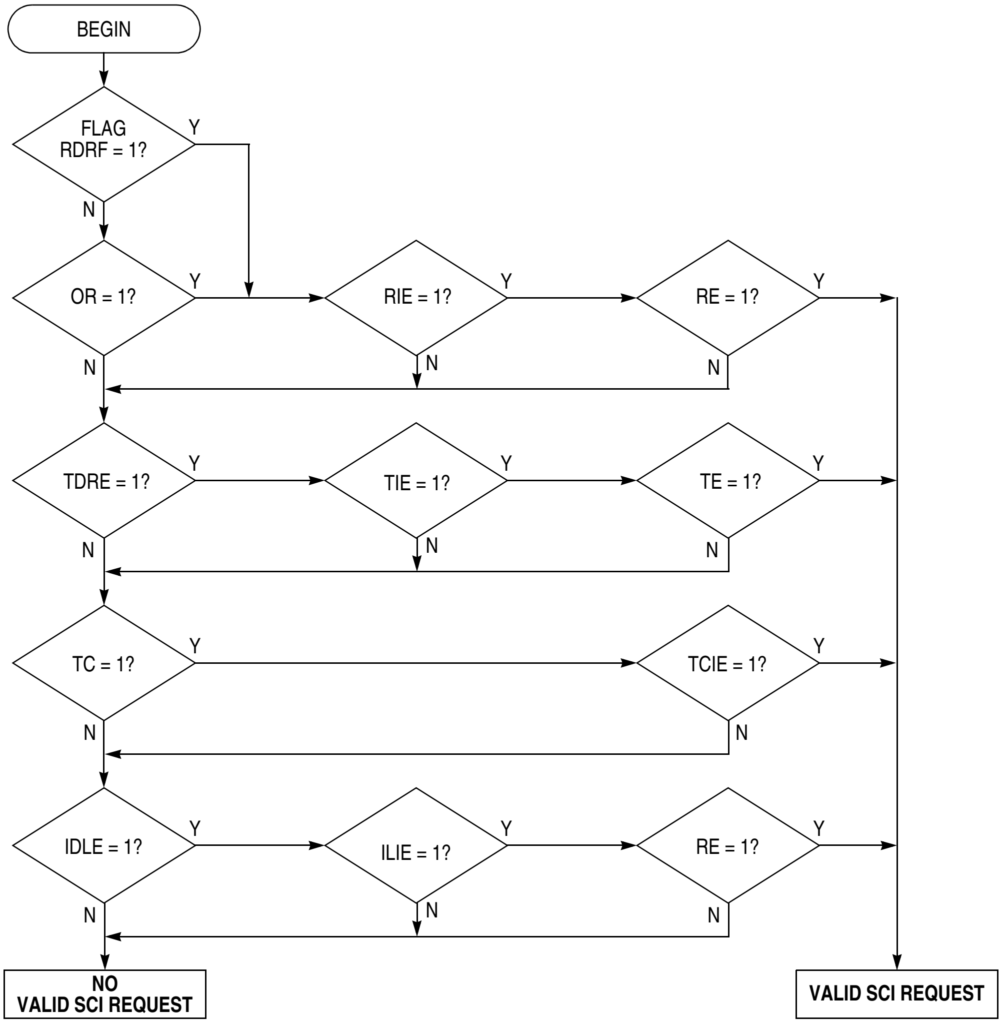


Figure 7-10. Interrupt Source Resolution Within SCI

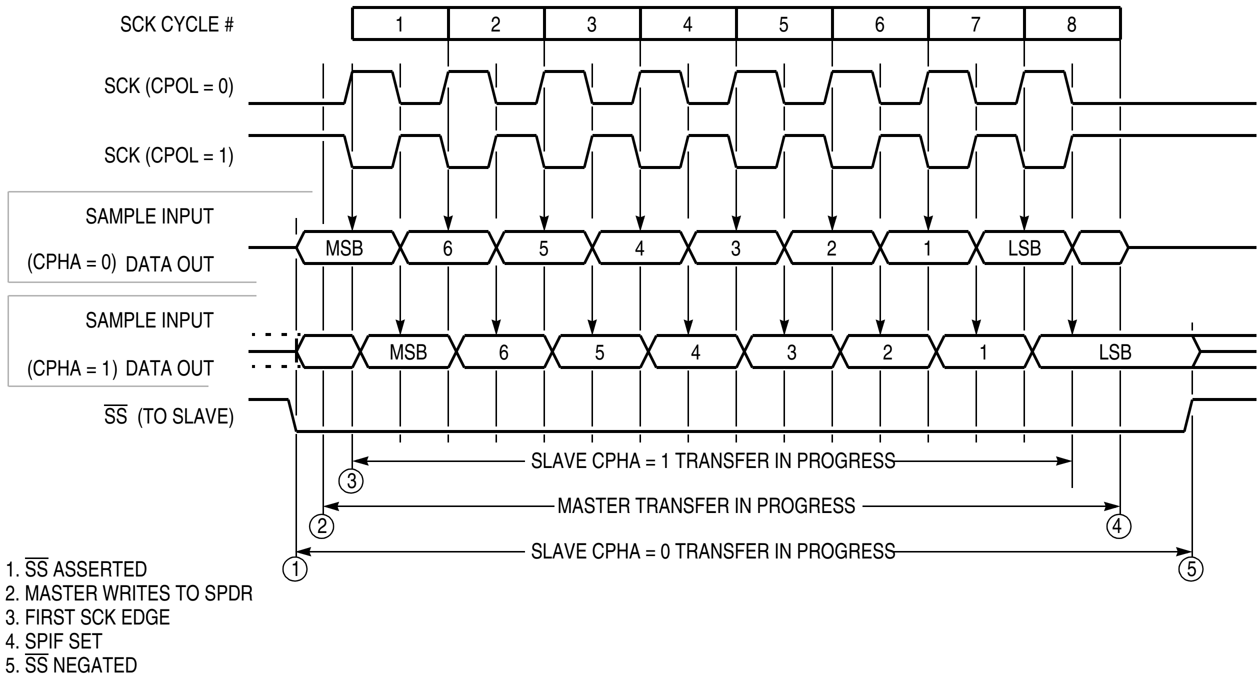


Figure 8-2. SPI Transfer Format

8.5 SPI Signals

This subsection contains descriptions of the four SPI signals:

- Master in/slave out (MISO)
- Master out/slave in (MOSI)
- Serial clock (SCK)
- Slave select ($\overline{\text{SS}}$)

Any SPI output line must have its corresponding data direction bit in DDRD register set. If the DDR bit is clear, that line is disconnected from the SPI logic and becomes a general-purpose input. All SPI input lines are forced to act as inputs regardless of the state of the corresponding DDR bits in DDRD register.

8.5.1 Master In/Slave Out

MISO is one of two unidirectional serial data signals. It is an input to a master device and an output from a slave device. The MISO line of a slave device is placed in the high-impedance state if the slave device is not selected.

8.5.2 Master Out/Slave In

The MOSI line is the second of the two unidirectional serial data signals. It is an output from a master device and an input to a slave device. The master device places data on the MOSI line a half-cycle before the clock edge that the slave device uses to latch the data.

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.

Bit 5 — Unimplemented
Always reads 0

MODF — Mode Fault Bit
To clear the MODF bit, read the SPSR (with MODF set), then write to the SPCR. Refer to 8.5.4 Slave Select and 8.6 SPI System Errors.
0 = No mode fault
1 = Mode fault

Bits [3:0] — Unimplemented
Always read 0

8.7.3 Serial Peripheral Data I/O Register

The SPDR is used when transmitting or receiving data on the serial bus. Only a write to this register initiates transmission or reception of a byte, and this only occurs in the master device. At the completion of transferring a byte of data, the SPIF status bit is set in both the master and slave devices.

A read of the SPDR is actually a read of a buffer. To prevent an overrun and the loss of the byte that caused the overrun, the first SPIF must be cleared by the time a second transfer of data from the shift register to the read buffer is initiated.

Address:	$102A							
	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	Indeterminate after reset							

Figure 8-5. Serial Peripheral Data I/O Register (SPDR)

SPI is double buffered in and single buffered out.

9.3.3 Timer Input Capture 4/Output Compare 5 Register

Use TI4/O5 as either an input capture register or an output compare register, depending on the function chosen for the PA3 pin. To enable it as an input capture pin, set the I4/O5 bit in the pulse accumulator control register (PACTL) to logic level 1. To use it as an output compare register, set the I4/O5 bit to a logic level 0. Refer to 9.7 Pulse Accumulator.

Register name: Timer Input Capture 4/Output Compare 5 (High) Address: $101E

	Bit 7	6	5	4	3	2	1	Bit 0
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

Register name: Timer Input Capture 4/Output Compare 5 (Low) Address: $101F

	Bit 7	6	5	4	3	2	1	Bit 0
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

Figure 9-7. Timer Input Capture 4/Output Compare 5 Register Pair (TI4/O5)

9.4 Output Compare

Use the output compare (OC) function to program an action to occur at a specific time — when the 16-bit counter reaches a specified value. For each of the five output compare functions, there is a separate 16-bit compare register and a dedicated 16-bit comparator. The value in the compare register is compared to the value of the free-running counter on every bus cycle. When the compare register matches the counter value, an output compare status flag is set. The flag can be used to initiate the automatic actions for that output compare function.

To produce a pulse of a specific duration, write a value to the output compare register that represents the time the leading edge of the pulse is to occur. The output compare circuit is configured to set the appropriate output either high or low, depending on the polarity of the pulse being produced. After a match occurs, the output compare register is reprogrammed to change the output pin back to its inactive level at the next match. A value representing the width of the pulse is added to the original value, and then written to the output compare register. Because the pin state changes occur at specific values of the free-running counter, the pulse width can be controlled accurately at the resolution of the free-running counter, independent of software latencies. To generate an output signal of a specific frequency and duty cycle, repeat this pulse-generating procedure.

The five 16-bit read/write output compare registers are: TOC1, TOC2, TOC3, and TOC4, and the TI4/O5. TI4/O5 functions under software control as either IC4 or OC5. Each of the OC registers is set to $FFFF on reset. A value written to an OC register is compared to the free-running counter value during each E-clock cycle. If a match is found, the particular output compare flag is set in timer interrupt flag register 1 (TFLG1). If that particular interrupt is enabled in the timer interrupt mask register 1 (TMSK1), an interrupt is generated. In addition to an interrupt, a specified action can be initiated at one or more timer output pins. For OC[5:2], the pin action is controlled by pairs of bits (OMx and OLx) in the TCTL1 register. The output action is taken on each successful compare, regardless of whether or not the OCxF flag in the TFLG1 register was previously cleared.

OC1 is different from the other output compares in that a successful OC1 compare can affect any or all five of the OC pins. The OC1 output action taken when a match is found is controlled by two 8-bit registers with three bits unimplemented: the output compare 1 mask register, OC1M, and the output compare 1 data register, OC1D. OC1M specifies which port A outputs are to be used, and OC1D specifies what data is placed on these port pins.

9.4.1 Timer Output Compare Registers

All output compare registers are 16-bit read-write. Each is initialized to $FFFF at reset. If an output compare register is not used for an output compare function, it can be used as a storage location. A write to the high-order byte of an output compare register pair inhibits the output compare function for one bus cycle. This inhibition prevents inappropriate subsequent comparisons. Coherency requires a complete 16-bit read or write. However, if coherency is not needed, byte accesses can be used.

For output compare functions, write a comparison value to output compare registers TOC1–TOC4 and TI4/O5. When TCNT value matches the comparison value, specified pin actions occur.

Register name: Timer Output Compare 1 Register (High) Address: $1016

	Bit 7	6	5	4	3	2	1	Bit 0
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

Register name: Timer Output Compare 1 Register (Low) Address: $1017

	Bit 7	6	5	4	3	2	1	Bit 0
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

Figure 9-8. Timer Output Compare 1 Register Pair (TOC1)

Register name: Timer Output Compare 2 Register (High) Address: $1018

	Bit 7	6	5	4	3	2	1	Bit 0
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

Register name: Timer Output Compare 2 Register (Low) Address: $1019

	Bit 7	6	5	4	3	2	1	Bit 0
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

Figure 9-9. Timer Output Compare 2 Register Pair (TOC2)

10.11 Peripheral Port Timing

Characteristic[1] [2]	Symbol	1.0 MHz		2.0 MHz		3.0 MHz		Unit
		Min	Max	Min	Max	Min	Max	
Frequency of operation E-clock frequency	f_o	dc	1.0	dc	2.0	dc	3.0	MHz
E-clock period	t_{CYC}	1000	—	500	—	333	—	ns
Peripheral data setup time MCU read of ports A, C, D, and E	t_{PDSU}	100	—	100	—	100	—	ns
Peripheral data hold time MCU read of ports A, C, D, and E	t_{PDH}	50	—	50	—	50	—	ns
Delay time, peripheral data write t_{PWD} = 1/4 t_{CYC}+ 100 ns MCU writes to port A MCU writes to ports B, C, and D	t_{PWD}	— —	200 350	— —	200 225	— —	200 183	ns
Port C input data setup time	t_{IS}	60	—	60	—	60	—	ns
Port C input data hold time	t_{IH}	100	—	100	—	100	—	ns
Delay time, E fall to STRB t_{DEB} = 1/4 t_{CYC}+ 100 ns	t_{DEB}	—	350	—	225	—	183	ns
Setup time, STRA asserted to E fall[3]	t_{AES}	0	—	0	—	0	—	ns
Delay time, STRA asserted to port C data output valid	t_{PCD}	—	100	—	100	—	100	ns
Hold time, STRA negated to port C data	t_{PCH}	10	—	10	—	10	—	ns
3-state hold time	t_{PCZ}	—	150	—	150	—	150	ns

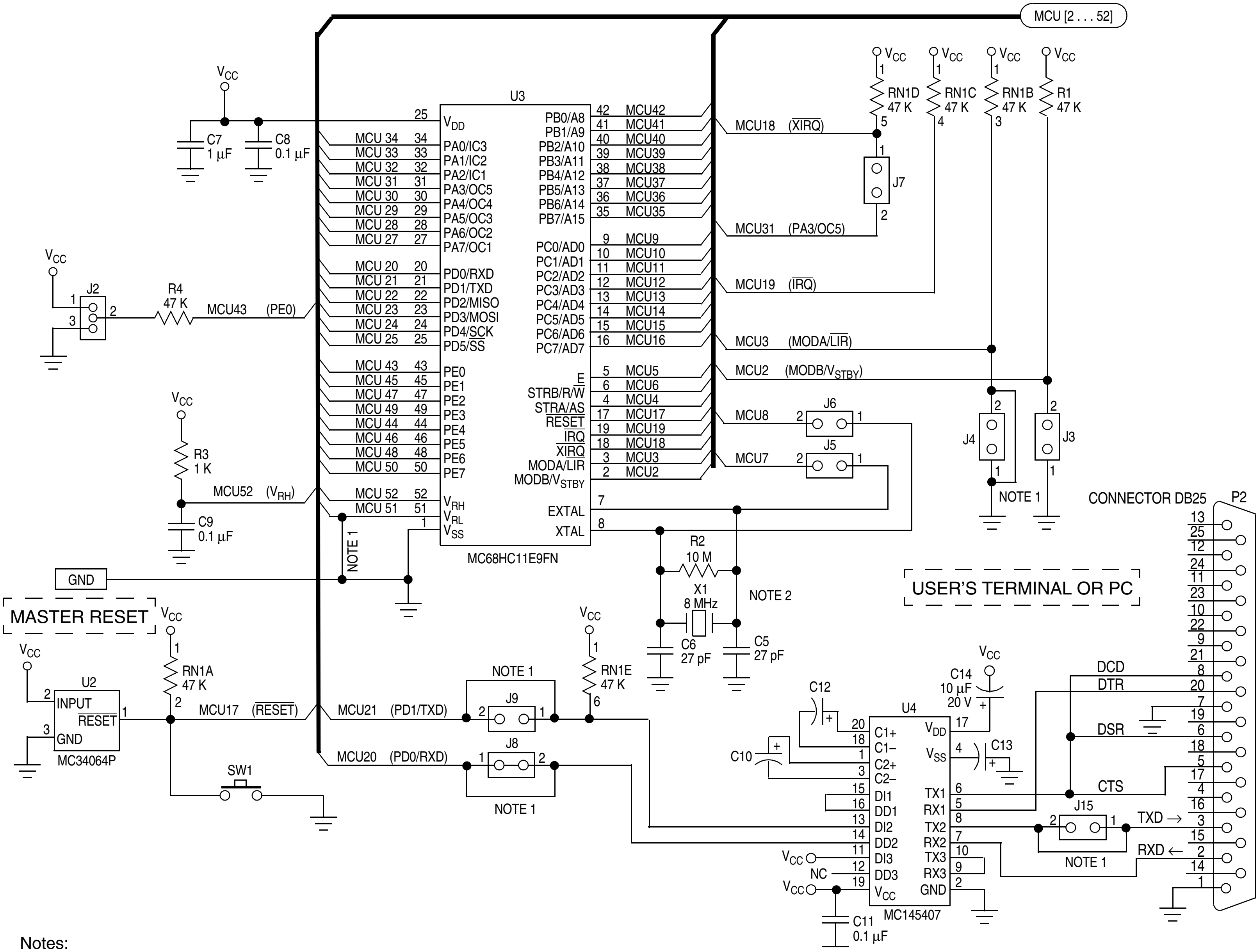
1. V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, T_A = T_L to T_H, all timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted
2. Ports C and D timing is valid for active drive. (CWOM and DWOM bits are not set in PIOC and SPCR registers, respectively.)
3. If this setup time is met, STRB acknowledges in the next cycle. If it is not met, the response may be delayed one more cycle.

10.16 MC68L11E9/E20 Expansion Bus Timing Characteristics

Num	Characteristic[1]	Symbol	1.0 MHz		2.0 MHz		Unit
			Min	Max	Min	Max	
	Frequency of operation (E-clock frequency)	f_o	dc	1.0	dc	2.0	MHz
1	Cycle time	t_{CYC}	1000	—	500	—	ns
2	Pulse width, E low, PW_{EL} = 1/2 t_{CYC}–25 ns	PW_{EL}	475	—	225	—	ns
3	Pulse width, E high, PW_{EH} = 1/2 t_{CYC}–30 ns	PW_{EH}	470	—	220	—	ns
4a	E and AS rise time	t_r	—	25	—	25	ns
4b	E and AS fall time	t_f	—	25	—	25	ns
9	Address hold time[2] [2]a, t_{AH} = 1/8 t_{CYC}–30 ns	t_{AH}	95	—	33	—	ns
12	Non-multiplexed address valid time to E rise t_{AV} = PW_{EL} –(t_{ASD} + 80 ns)[2]a	t_{AV}	275	—	88	—	ns
17	Read data setup time	t_{DSR}	30	—	30	—	ns
18	Read data hold time , max = t_{MAD}	t_{DHR}	0	150	0	88	ns
19	Write data delay time, t_{DDW} = 1/8 t_{CYC}+ 70 ns[2]a	t_{DDW}	—	195	—	133	ns
21	Write data hold time, t_{DHW} = 1/8 t_{CYC}–30 ns[2]a	t_{DHW}	95	—	33	—	ns
22	Multiplexed address valid time to E rise t_{AVM} = PW_{EL} –(t_{ASD} + 90 ns)[2]a	t_{AVM}	268	—	78	—	ns
24	Multiplexed address valid time to AS fall t_{ASL} = PW_{ASH} –70 ns	t_{ASL}	150	—	25	—	ns
25	Multiplexed address hold time, t_{AHL} = 1/8 t_{CYC}–30 ns[2]b	t_{AHL}	95	—	33	—	ns
26	Delay time, E to AS rise, t_{ASD} = 1/8 t_{CYC}–5 ns[2]a	t_{ASD}	120	—	58	—	ns
27	Pulse width, AS high, PW_{ASH} = 1/4 t_{CYC}–30 ns	PW_{ASH}	220	—	95	—	ns
28	Delay time, AS to E rise, t_{ASED} = 1/8 t_{CYC}–5 ns[2]b	t_{ASED}	120	—	58	—	ns
29	MPU address access time[3]a t_{ACCA} = t_{CYC}–(PW_{EL}–t_{AVM}) –t_{DSR}–t_f	t_{ACCA}	735	—	298	—	ns
35	MPU access time, t_{ACCE} = PW_{EH} –t_{DSR}	t_{ACCE}	—	440	—	190	ns
36	Multiplexed address delay (Previous cycle MPU read) t_{MAD} = t_{ASD} + 30 ns[2]a	t_{MAD}	150	—	88	—	ns

1. V_{DD} = 3.0 Vdc to 5.5 Vdc, V_{SS} = 0 Vdc, T_A = T_L to T_H, all timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted
2. Input clocks with duty cycles other than 50% affect bus performance. Timing parameters affected by input clock duty cycle are identified by (a) and (b). To recalculate the approximate bus timing values, substitute the following expressions in place of 1/8 t_{CYC}in the above formulas, where applicable:
(a) (1–dc) × 1/4 t_{CYC}
(b) dc × 1/4 t_{CYC}

Where:
dc is the decimal value of duty cycle percentage (high time).



Notes:
1. Default cut traces installed from factory on bottom of the board.
2. X1 is shipped as a ceramic resonator with built-in capacitors. Holes are provided for a crystal and two capacitors.

Figure B-1. EVBU Schematic Diagram

```
1640 GOSUB 8000                 'GET BYTE FOR VERIFICATION
1650 RCV = I - 1
1660 LOCATE 10,1:PRINT "Verifying byte #"; I; "    "
1664 IF CHR$(CODE%(RCV)) = B$ THEN 1670
1665 K=CODE%(RCV):GOSUB 8500
1666 LOCATE 1,1:PRINT "Byte #"; I; "    ", " - Sent "; HX$;
1668 K=ASC(B$):GOSUB 8500
1669 PRINT "  Received "; HX$;
1670 NEXT I
1680 GOSUB 8000                 'GET BYTE FOR VERIFICATION
1690 RCV = CODESIZE% - 1
1700 LOCATE 10,1:PRINT "Verifying byte #"; CODESIZE%; "    "
1710 IF CHR$(CODE%(RCV)) = B$ THEN 1720
1713 K=CODE(RCV):GOSUB 8500
1714 LOCATE 1,1:PRINT "Byte #"; CODESIZE%; "    ", " - Sent "; HX$;
1715 K=ASC(B$):GOSUB 8500
1716 PRINT "  Received "; HX$;
1720 LOCATE 8, 1: PRINT : PRINT "Done!!!!"
4900 CLOSE
4910 INPUT "Press [RETURN] to quit...", Q$
5000 END
5900 '***********************************************************************
5910 '*         SUBROUTINE TO READ IN ONE BYTE FROM A DISK FILE
5930 '*                 RETURNS BYTE IN A$
5940 '***********************************************************************
6000 FLAG = 0
6010 IF EOF(1) THEN FLAG = 1: RETURN
6020 A$ = INPUT$(1, #1)
6030 RETURN
6490 '***********************************************************************
6492 '*         SUBROUTINE TO SEND THE STRING IN A$ OUT TO THE DEVICE
6494 '*                 OPENED AS FILE #2.
6496 '***********************************************************************
6500 PRINT #2, A$;
6510 RETURN
6590 '***********************************************************************
6594 '*         SUBROUTINE THAT CONVERTS THE HEX DIGIT IN A$ TO AN INTEGER
6596 '***********************************************************************
7000 X = INSTR(H$, A$)
7010 IF X = 0 THEN FLAG = 1
7020 X = X - 1
7030 RETURN
7990 '**********************************************************************
7992 '*         SUBROUTINE TO READ IN ONE BYTE THROUGH THE COMM PORT OPENED
7994 '*                 AS FILE #2.  WAITS INDEFINITELY FOR THE BYTE TO BE
7996 '*                 RECEIVED.  SUBROUTINE WILL BE ABORTED BY ANY
7998 '*                 KEYBOARD INPUT.  RETURNS BYTE IN B$.  USES Q$.
7999 '**********************************************************************
8000 WHILE LOC(2) = 0           'WAIT FOR COMM PORT INPUT
8005 Q$ = INKEY$: IF Q$ <> "" THEN 4900 'IF ANY KEY PRESSED, THEN ABORT
8010 WEND
8020 B$ = INPUT$(1, #2)
8030 RETURN
8490 '************************************************************************
```