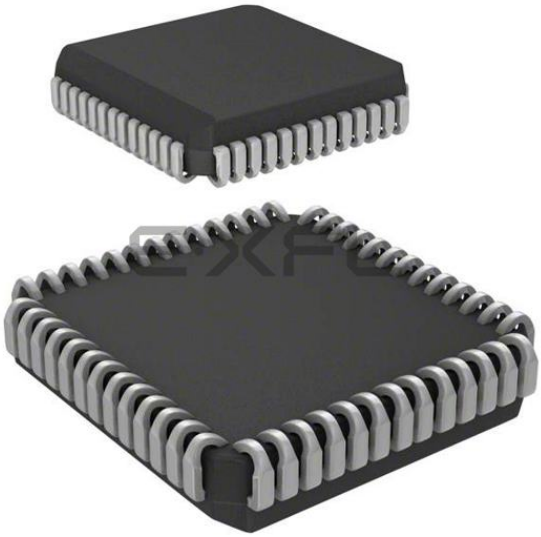


Welcome to E-XFL.COM

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	3MHz
Connectivity	SCI, SPI
Peripherals	POR, WDT
Number of I/O	38
Program Memory Size	-
Program Memory Type	ROMless
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	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/mc68hcp11e1cfne3

Chapter 1
General Description

1.1 Introduction

This document contains a detailed description of the M68HC11 E series of 8-bit microcontroller units (MCUs). These MCUs all combine the M68HC11 central processor unit (CPU) with high-performance, on-chip peripherals.

The E series is comprised of many devices with various configurations of:

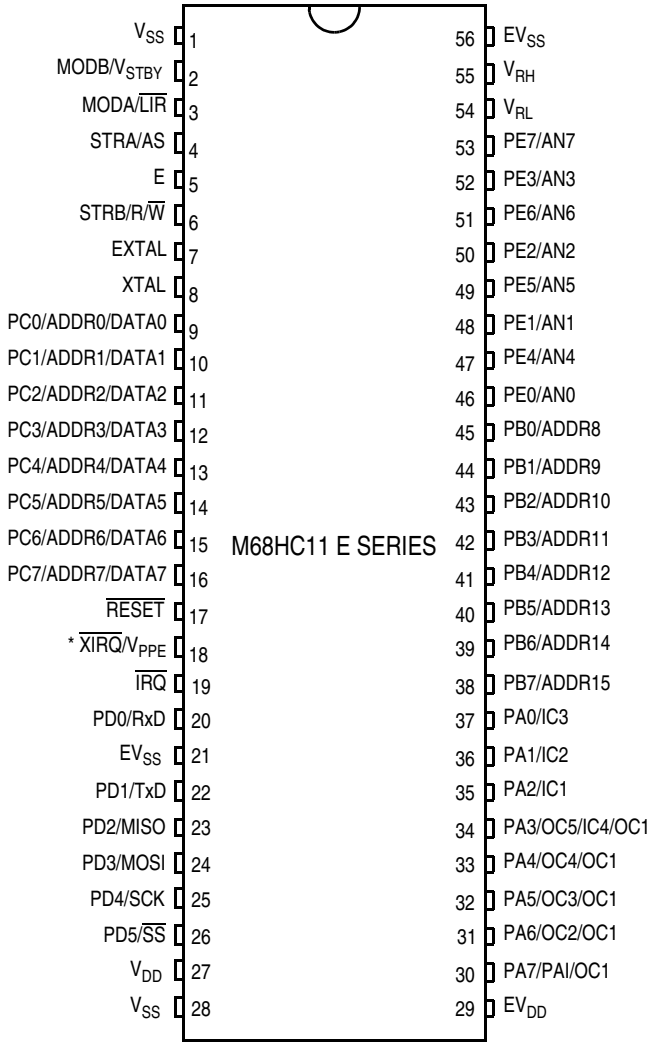
- Random-access memory (RAM)
- Read-only memory (ROM)
- Erasable programmable read-only memory (EPROM)
- Electrically erasable programmable read-only memory (EEPROM)
- Several low-voltage devices are also available.

With the exception of a few minor differences, the operation of all E-series MCUs is identical. A fully static design and high-density complementary metal-oxide semiconductor (HCMOS) fabrication process allow the E-series devices to operate at frequencies from 3 MHz to dc with very low power consumption.

1.2 Features

Features of the E-series devices include:

- M68HC11 CPU
- Power-saving stop and wait modes
- Low-voltage devices available (3.0–5.5 Vdc)
- 0, 256, 512, or 768 bytes of on-chip RAM, data retained during standby
- 0, 12, or 20 Kbytes of on-chip ROM or EPROM
- 0, 512, or 2048 bytes of on-chip EEPROM with block protect for security
- 2048 bytes of EEPROM with selectable base address in the MC68HC811E2
- Asynchronous non-return-to-zero (NRZ) serial communications interface (SCI)
- Additional baud rates available on MC68HC(7)11E20
- Synchronous serial peripheral interface (SPI)
- 8-channel, 8-bit analog-to-digital (A/D) converter
- 16-bit timer system:
  - Three input capture (IC) channels
  - Four output compare (OC) channels
  - One additional channel, selectable as fourth IC or fifth OC
- 8-bit pulse accumulator
- Real-time interrupt circuit



* V_{PPE} applies only to devices with EPROM/OTPROM.

Figure 1-5. Pin Assignments for 56-Pin SDIP

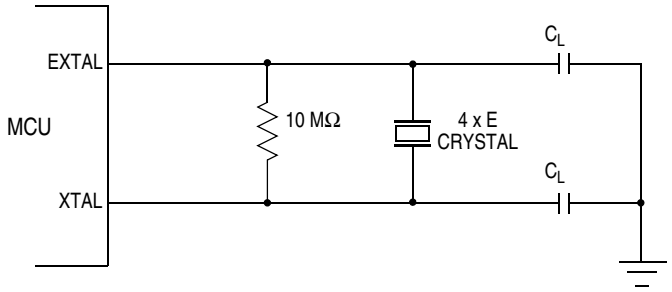


Figure 1-9. Common Parallel Resonant Crystal Connections

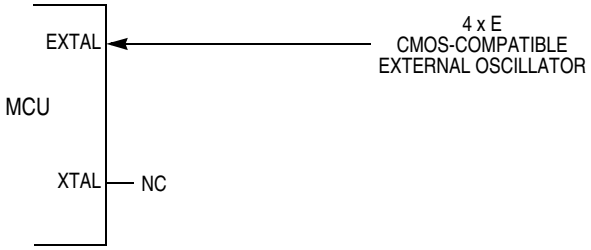


Figure 1-10. External Oscillator Connections

1.4.4 E-Clock Output (E)

E is the output connection for the internally generated E clock. The signal from E is used as a timing reference. The frequency of the E-clock output is one fourth that of the input frequency at the XTAL and EXTAL pins. When E-clock output is low, an internal process is taking place. When it is high, data is being accessed.

All clocks, including the E clock, are halted when the MCU is in stop mode. To reduce RFI emissions, the E-clock output of most E-series devices can be disabled while operating in single-chip modes.

The E-clock signal is always enabled on the MC68HC811E2.

1.4.5 Interrupt Request ($\overline{IRQ}$)

The $\overline{IRQ}$ input provides a means of applying asynchronous interrupt requests to the MCU. Either negative edge-sensitive triggering or level-sensitive triggering is program selectable (OPTION register). $\overline{IRQ}$ is always configured to level-sensitive triggering at reset. When using $\overline{IRQ}$ in a level-sensitive wired-OR configuration, connect an external pullup resistor, typically 4.7 kΩ, to V_{DD}.

1.4.6 Non-Maskable Interrupt ($\overline{XIRQ}/V_{PPE}$)

The $\overline{XIRQ}$ input provides a means of requesting a non-maskable interrupt after reset initialization. During reset, the X bit in the condition code register (CCR) is set and any interrupt is masked until MCU software enables it. Because the $\overline{XIRQ}$ input is level-sensitive, it can be connected to a multiple-source wired-OR network with an external pullup resistor to V_{DD}. $\overline{XIRQ}$ is often used as a power loss detect interrupt.

Whenever $\overline{XIRQ}$ or $\overline{IRQ}$ is used with multiple interrupt sources each source must drive the interrupt input with an open-drain type of driver to avoid contention between outputs.

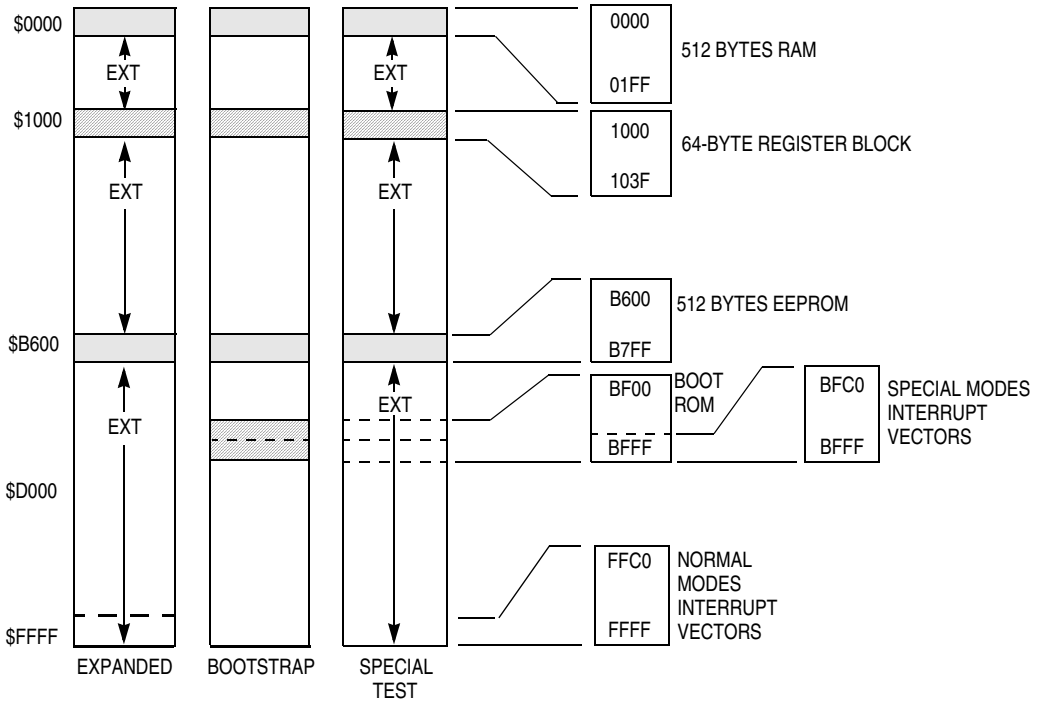


Figure 2-3. Memory Map for MC68HC11E1

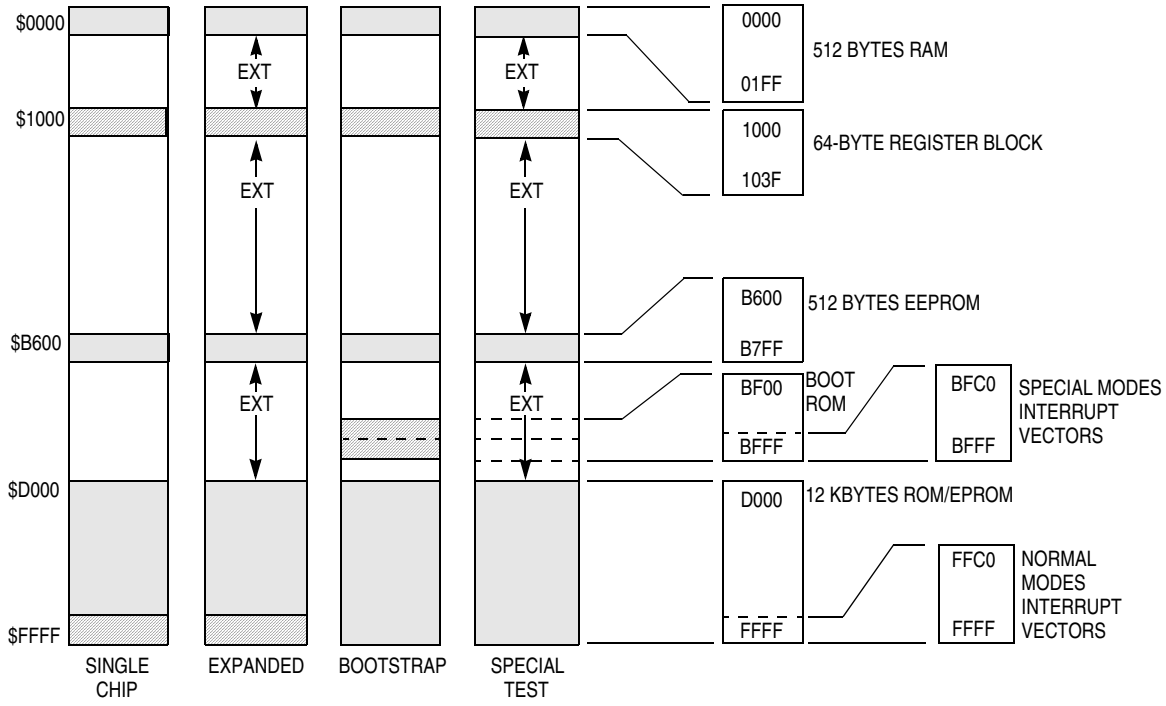


Figure 2-4. Memory Map for MC68HC(7)11E9

Table 4-2. Instruction Set (Sheet 6 of 7)

Mnemonic	Operation	Description	Addressing Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	X	H	I	N	Z	V	C
SBCA (opr)	Subtract with Carry from A	A – M – C ⇒ A	A IMM	82	ii	2	—	—	—	—	Δ	Δ	Δ	Δ
			A DIR	92	dd	3								
			A EXT	B2	hh ll	4								
			A IND,X	A2	ff	4								
			A IND,Y	18 A2	ff	5								
SBCB (opr)	Subtract with Carry from B	B – M – C ⇒ B	B IMM	C2	ii	2	—	—	—	—	Δ	Δ	Δ	Δ
			B DIR	D2	dd	3								
			B EXT	F2	hh ll	4								
			B IND,X	E2	ff	4								
			B IND,Y	18 E2	ff	5								
SEC	Set Carry	1 ⇒ C	INH	0D	—	2	—	—	—	—	—	—	—	1
SEI	Set Interrupt Mask	1 ⇒ I	INH	0F	—	2	—	—	—	1	—	—	—	—
SEV	Set Overflow Flag	1 ⇒ V	INH	0B	—	2	—	—	—	—	—	—	1	—
STAA (opr)	Store Accumulator A	A ⇒ M	A DIR	97	dd	3	—	—	—	—	Δ	Δ	0	—
			A EXT	B7	hh ll	4								
			A IND,X	A7	ff	4								
			A IND,Y	18 A7	ff	5								
STAB (opr)	Store Accumulator B	B ⇒ M	B DIR	D7	dd	3	—	—	—	—	Δ	Δ	0	—
			B EXT	F7	hh ll	4								
			B IND,X	E7	ff	4								
			B IND,Y	18 E7	ff	5								
STD (opr)	Store Accumulator D	A ⇒ M, B ⇒ M + 1	DIR	DD	dd	4	—	—	—	—	Δ	Δ	0	—
			EXT	FD	hh ll	5								
			IND,X	ED	ff	5								
			IND,Y	18 ED	ff	6								
STOP	Stop Internal Clocks	—	INH	CF	—	2	—	—	—	—	—	—	—	—
STS (opr)	Store Stack Pointer	SP ⇒ M : M + 1	DIR	9F	dd	4	—	—	—	—	Δ	Δ	0	—
			EXT	BF	hh ll	5								
			IND,X	AF	ff	5								
			IND,Y	18 AF	ff	6								
STX (opr)	Store Index Register X	IX ⇒ M : M + 1	DIR	DF	dd	4	—	—	—	—	Δ	Δ	0	—
			EXT	FF	hh ll	5								
			IND,X	EF	ff	5								
			IND,Y	CD EF	ff	6								
STY (opr)	Store Index Register Y	IY ⇒ M : M + 1	DIR	18 DF	dd	5	—	—	—	—	Δ	Δ	0	—
			EXT	18 FF	hh ll	6								
			IND,X	1A EF	ff	6								
			IND,Y	18 EF	ff	6								
SUBA (opr)	Subtract Memory from A	A – M ⇒ A	A IMM	80	ii	2	—	—	—	—	Δ	Δ	Δ	Δ
			A DIR	90	dd	3								
			A EXT	B0	hh ll	4								
			A IND,X	A0	ff	4								
			A IND,Y	18 A0	ff	5								
SUBB (opr)	Subtract Memory from B	B – M ⇒ B	A IMM	C0	ii	2	—	—	—	—	Δ	Δ	Δ	Δ
			A DIR	D0	dd	3								
			A EXT	F0	hh ll	4								
			A IND,X	E0	ff	4								
			A IND,Y	18 E0	ff	5								
SUBD (opr)	Subtract Memory from D	D – M : M + 1 ⇒ D	IMM	83	jj kk	4	—	—	—	—	Δ	Δ	Δ	Δ
			DIR	93	dd	5								
			EXT	B3	hh ll	6								
			IND,X	A3	ff	6								
			IND,Y	18 A3	ff	7								
SWI	Software Interrupt	See Figure 3–2	INH	3F	—	14	—	—	—	1	—	—	—	—
TAB	Transfer A to B	A ⇒ B	INH	16	—	2	—	—	—	—	Δ	Δ	0	—
TAP	Transfer A to CC Register	A ⇒ CCR	INH	06	—	2	Δ	↓	Δ	Δ	Δ	Δ	Δ	Δ
TBA	Transfer B to A	B ⇒ A	INH	17	—	2	—	—	—	—	Δ	Δ	0	—
TEST	TEST (Only in Test Modes)	Address Bus Counts	INH	00	—	*	—	—	—	—	—	—	—	—
TPA	Transfer CC Register to A	CCR ⇒ A	INH	07	—	2	—	—	—	—	—	—	—	—
TST (opr)	Test for Zero or Minus	M – 0	EXT	7D	hh ll	6	—	—	—	—	Δ	Δ	0	0
			IND,X	6D	ff	6								
			IND,Y	18 6D	ff	7								

Table 5-3. Highest Priority Interrupt Selection

PSEL[3:0]	Interrupt Source Promoted
0 0 0 0	Timer overflow
0 0 0 1	Pulse accumulator overflow
0 0 1 0	Pulse accumulator input edge
0 0 1 1	SPI serial transfer complete
0 1 0 0	SCI serial system
0 1 0 1	Reserved (default to $\overline{\text{IRQ}}$)
0 1 1 0	$\overline{\text{IRQ}}$ (external pin or parallel I/O)
0 1 1 1	Real-time interrupt
1 0 0 0	Timer input capture 1
1 0 0 1	Timer input capture 2
1 0 1 0	Timer input capture 3
1 0 1 1	Timer output compare 1
1 1 0 0	Timer output compare 2
1 1 0 1	Timer output compare 3
1 1 1 0	Timer output compare 4
1 1 1 1	Timer input capture 4/output compare 5

5.5 Interrupts

The MCU has 18 interrupt vectors that support 22 interrupt sources. The 15 maskable interrupts are generated by on-chip peripheral systems. These interrupts are recognized when the global interrupt mask bit (I) in the condition code register (CCR) is clear. The three non-maskable interrupt sources are illegal opcode trap, software interrupt, and $\overline{\text{XIRQ}}$ pin. Refer to Table 5-4, which shows the interrupt sources and vector assignments for each source.

For some interrupt sources, such as the SCI interrupts, the flags are automatically cleared during the normal course of responding to the interrupt requests. For example, the RDRF flag in the SCI system is cleared by the automatic clearing mechanism consisting of a read of the SCI status register while RDRF is set, followed by a read of the SCI data register. The normal response to an RDRF interrupt request would be to read the SCI status register to check for receive errors, then to read the received data from the SCI data register. These steps satisfy the automatic clearing mechanism without requiring special instructions.

6.2 Port A

Port A shares functions with the timer system and has:

- Three input-only pins
- Three output-only pins
- Two bidirectional I/O pins

Address:	$1000							
	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
Reset:	I	0	0	0	I	I	I	I
Alternate function:	PAI	OC2	OC3	OC4	IC4/OC5	IC1	IC2	IC3
And/or:	OC1	OC1	OC1	OC1	OC1	—	—	—

I = Indeterminate after reset

Figure 6-1. Port A Data Register (PORTA)

Address:	$1026							
	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRA7	PAEWN	PAMOD	PEDGE	DDRA3	I4/O5	RTR1	RTR0
Reset:	0	0	0	0	0	0	0	0

Figure 6-2. Pulse Accumulator Control Register (PACTL)

DDRA7 — Data Direction for Port A Bit 7

Overridden if an output compare function is configured to control the PA7 pin

0 = Input

1 = Output

The pulse accumulator uses port A bit 7 as the PAI input, but the pin can also be used as general-purpose I/O or as an output compare.

NOTE

Even when port A bit 7 is configured as an output, the pin still drives the input to the pulse accumulator.

PAEN — Pulse Accumulator System Enable Bit

Refer to Chapter 9 Timing Systems.

PAMOD — Pulse Accumulator Mode Bit

Refer to Chapter 9 Timing Systems.

PEDGE — Pulse Accumulator Edge Control Bit

Refer to Chapter 9 Timing Systems.

DDRA3 — Data Direction for Port A Bit 3

This bit is overridden if an output compare function is configured to control the PA3 pin.

0 = Input

1 = Output

I4/O5 — Input Capture 4/Output Compare 5 Bit

Refer to Chapter 9 Timing Systems.

RTR[1:0] — RTI Interrupt Rate Select Bits

Refer to Chapter 9 Timing Systems.

SCR[2:0] — SCI Baud Rate Select Bits

Selects receiver and transmitter bit rate based on output from baud rate prescaler stage. Refer to Figure 7-8 and Figure 7-9.

The prescaler bits, SCP[2:0], determine the highest baud rate, and the SCR[2:0] bits select an additional binary submultiple (÷1, ÷2, ÷4, through ÷128) of this highest baud rate. The result of these two dividers in series is the 16X receiver baud rate clock. The SCR[2:0] bits are not affected by reset and can be changed at any time, although they should not be changed when any SCI transfer is in progress.

Figure 7-8 and Figure 7-9 illustrate the SCI baud rate timing chain. The prescaler select bits determine the highest baud rate. The rate select bits determine additional divide by two stages to arrive at the receiver timing (RT) clock rate. The baud rate clock is the result of dividing the RT clock by 16.

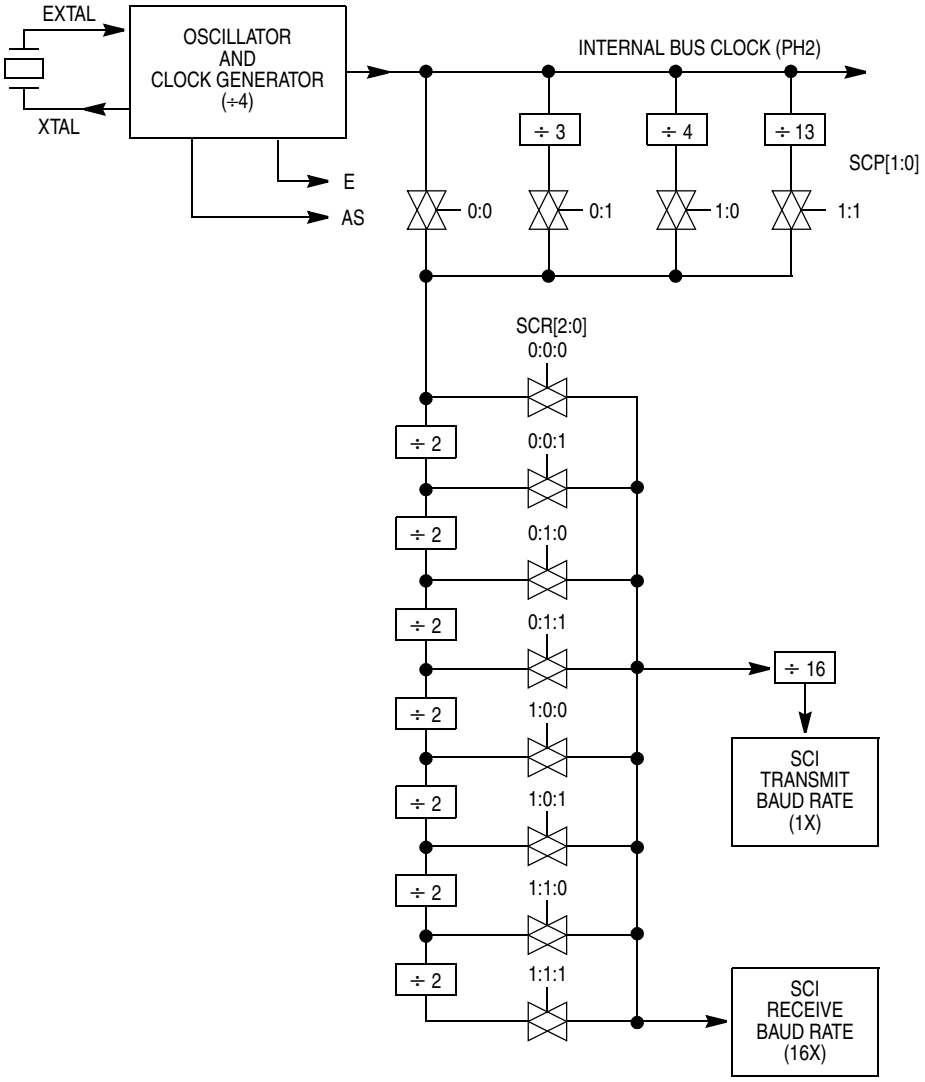


Figure 7-8. SCI Baud Rate Generator Block Diagram

A write collision is normally a slave error because a slave has no control over when a master initiates a transfer. A master knows when a transfer is in progress, so there is no reason for a master to generate a write-collision error, although the SPI logic can detect write collisions in both master and slave devices.

The SPI configuration determines the characteristics of a transfer in progress. For a master, a transfer begins when data is written to SPDR and ends when SPIF is set. For a slave with CPHA equal to 0, a transfer starts when $\overline{SS}$ goes low and ends when $\overline{SS}$ returns high. In this case, SPIF is set at the middle of the eighth SCK cycle when data is transferred from the shifter to the parallel data register, but the transfer is still in progress until $\overline{SS}$ goes high. For a slave with CPHA equal to 1, transfer begins when the SCK line goes to its active level, which is the edge at the beginning of the first SCK cycle. The transfer ends in a slave in which CPHA equals 1 when SPIF is set.

8.7 SPI Registers

The three SPI registers are:

- Serial peripheral control register (SPCR)
- Serial peripheral status register (SPSR)
- Serial peripheral data register (SPDR)

These registers provide control, status, and data storage functions.

8.7.1 Serial Peripheral Control Register

Address: $1028

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	SPIE	SPE	DWOM	MSTR	CPOL	CPHA	SPR1	SPR0
Reset:	0	0	0	0	0	1	U	U

U = Unaffected

Figure 8-3. Serial Peripheral Control Register (SPCR)

SPIE — Serial Peripheral Interrupt Enable Bit

Set the SPE bit to 1 to request a hardware interrupt sequence each time the SPIF or MODF status flag is set. SPI interrupts are inhibited if this bit is clear or if the I bit in the condition code register is 1.

0 = SPI system interrupts disabled
1 = SPI system interrupts enabled

SPE — Serial Peripheral System Enable Bit

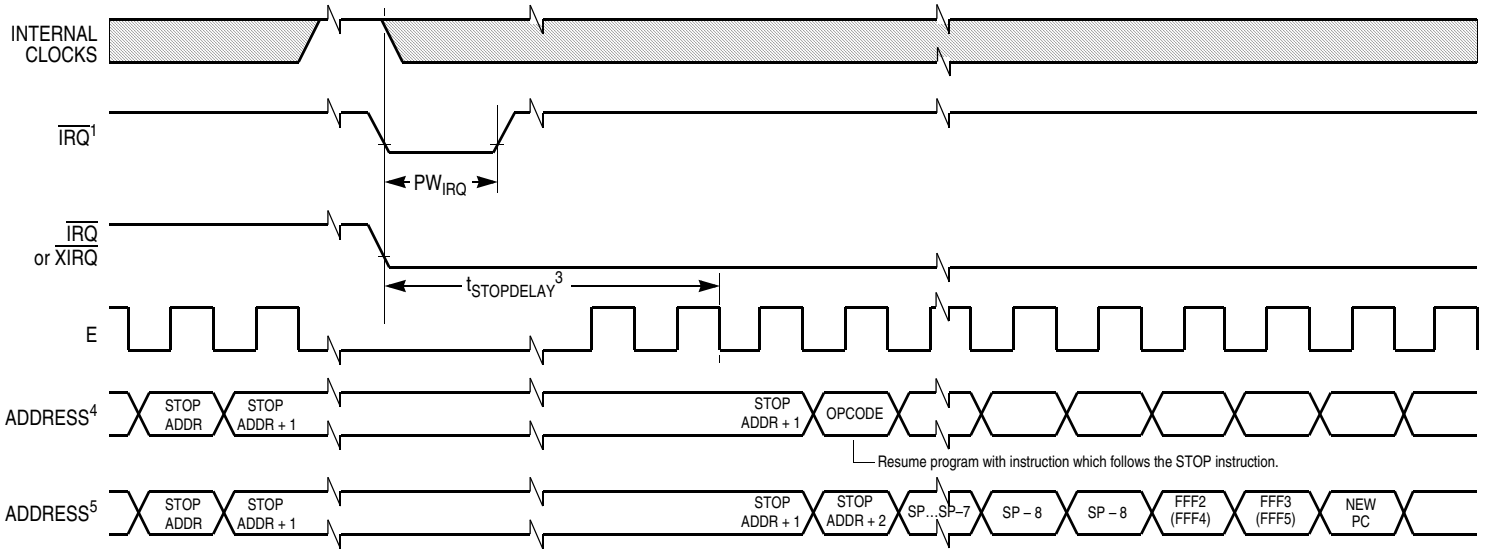
When the SPE bit is set, the port D bit 2, 3, 4, and 5 pins are dedicated to the SPI function. If the SPI is in the master mode and DDRD bit 5 is set, then the port D bit 5 pin becomes a general-purpose output instead of the $\overline{SS}$ input.

0 = SPI system disabled
1 = SPI system enabled

DWOM — Port D Wired-OR Mode Bit

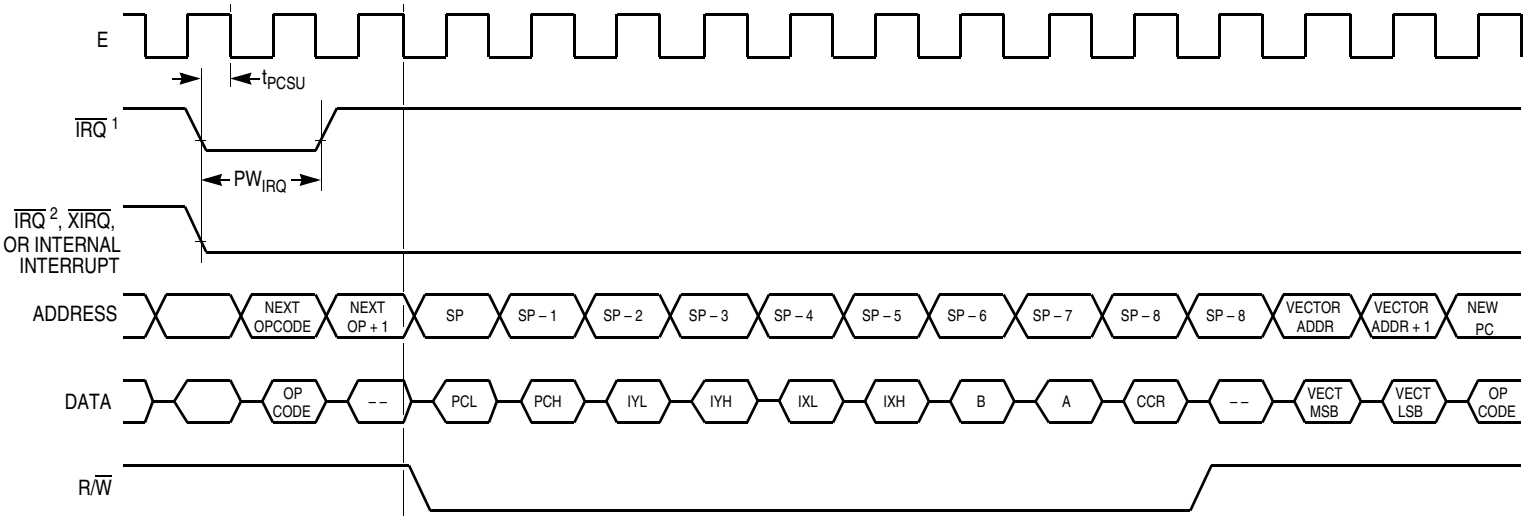
DWOM affects all port D pins.

0 = Normal CMOS outputs
1 = Open-drain outputs



Notes:
1. Edge Sensitive $\overline{\text{IRQ}}$ pin (IRQE bit = 1)
2. Level sensitive $\overline{\text{IRQ}}$ pin (IRQE bit = 0)
3. $t_{STOPDELAY}$ = 4064 t_{CYC} if DLY bit = 1 or 4 t_{CYC} if DLY = 0.
4. $\overline{\text{XIRQ}}$ with X bit in CCR = 1.
5. $\overline{\text{IRQ}}$ or ($\overline{\text{XIRQ}}$ with X bit in CCR = 0).

Figure 10-4. STOP Recovery Timing Diagram



Notes:
1. Edge sensitive $\overline{IRQ}$ pin (IRQE bit = 1)
2. Level sensitive $\overline{IRQ}$ pin (IRQE bit = 0)

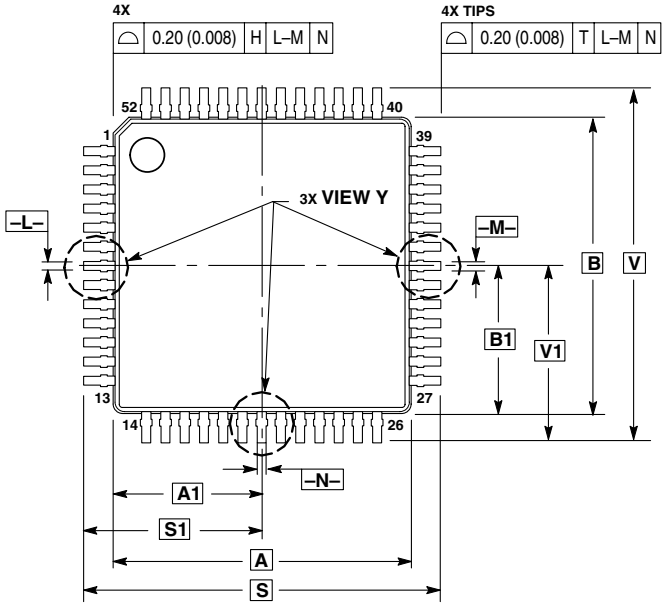
Figure 10-6. Interrupt Timing Diagram

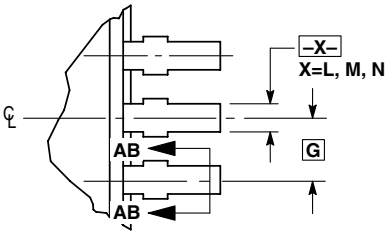
10.13 Analog-to-Digital Converter Characteristics

Characteristic[1]	Parameter[2]	Min	Absolute	2.0 MHz Max	3.0 MHz Max	Unit
Resolution	Number of bits resolved by A/D converter	—	8	—	—	Bits
Non-linearity	Maximum deviation from the ideal A/D transfer characteristics	—	—	±1/2	±1	LSB
Zero error	Difference between the output of an ideal and an actual for 0 input voltage	—	—	±1/2	±1	LSB
Full scale error	Difference between the output of an ideal and an actual A/D for full-scale input voltage	—	—	±1/2	±1	LSB
Total unadjusted error	Maximum sum of non-linearity, zero error, and full-scale error	—	—	±1/2	±1/2	LSB
Quantization error	Uncertainty because of converter resolution	—	—	±1/2	±1/2	LSB
Absolute accuracy	Difference between the actual input voltage and the full-scale weighted equivalent of the binary output code, all error sources included	—	—	±1	±2	LSB
Conversion range	Analog input voltage range	V_{RL}	—	V_{RH}	V_{RH}	V
V_{RH}	Maximum analog reference voltage[3]	V_{RL}	—	V_{DD} +0.1	V_{DD} +0.1	V
V_{RL}	Minimum analog reference voltage[2]	V_{SS} –0.1	—	V_{RH}	V_{RH}	V
ΔV_R	Minimum difference between V_{RH} and V_{RL}[2]	3	—	—	—	V
Conversion time	Total time to perform a single A/D conversion: E clock Internal RC oscillator	 — —	 32 —	 — t_{CYC}+32	 — t_{CYC}+32	 t_{CYC} µs
Monotonicity	Conversion result never decreases with an increase in input voltage; has no missing codes	—	Guaranteed	—	—	—
Zero input reading	Conversion result when $V_{In} = V_{RL}$	00	—	—	—	Hex
Full scale reading	Conversion result when $V_{In} = V_{RH}$	—	—	FF	FF	Hex
Sample acquisition time	Analog input acquisition sampling time: E clock Internal RC oscillator	 — —	 12 —	 — 12	 — 12	 t_{CYC} µs
Sample/hold capacitance	Input capacitance during sample PE[7:0]	—	20 typical	—	—	pF
Input leakage	Input leakage on A/D pins PE[7:0] V_{RL}, V_{RH}	 — —	 — —	 400 1.0	 400 1.0	 nA µA

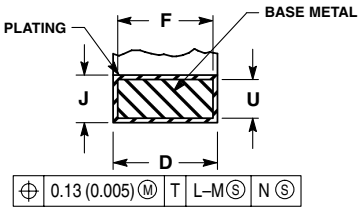
1. V_{DD} = 5.0 Vdc ±10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, 750 kHz ≤ E ≤ 3.0 MHz, unless otherwise noted
2. Source impedances greater than 10 kΩ affect accuracy adversely because of input leakage.
3. Performance verified down to 2.5 V ΔV_R, but accuracy is tested and guaranteed at ΔV_R = 5 V ±10%.

11.8 52-Pin Thin Quad Flat Pack (Case 848D)

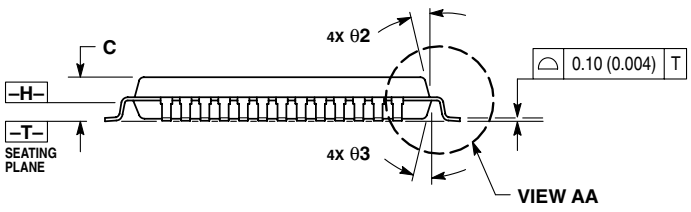


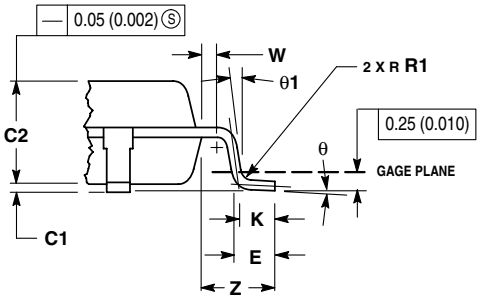


VIEW Y



SECTION AB–AB
ROTATED 90° CLOCKWISE





VIEW AA

NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.
3. DATUM PLANE –H– IS LOCATED AT BOTTOM OF LEAD AND IS COINCIDENT WITH THE LEAD WHERE THE LEAD EXITS THE PLASTIC BODY AT THE BOTTOM OF THE PARTING LINE.
4. DATUMS –L–, –M– AND –N– TO BE DETERMINED AT DATUM PLANE –H–.
5. DIMENSIONS S AND V TO BE DETERMINED AT SEATING PLANE –T–.
6. DIMENSIONS A AND B DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25 (0.010) PER SIDE. DIMENSIONS A AND B DO INCLUDE MOLD MISMATCH AND ARE DETERMINED AT DATUM PLANE -H-.
7. DIMENSION D DOES NOT INCLUDE DAMBAR PROTRUSION. DAMBAR PROTRUSION SHALL NOT CAUSE THE LEAD WIDTH TO EXCEED 0.46 (0.018). MINIMUM SPACE BETWEEN PROTRUSION AND ADJACENT LEAD OR PROTRUSION 0.07 (0.003).

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	10.00 BSC		0.394 BSC	
A1	5.00 BSC		0.197 BSC	
B	10.00 BSC		0.394 BSC	
B1	5.00 BSC		0.197 BSC	
C	—	1.70	—	0.067
C1	0.05	0.20	0.002	0.008
C2	1.30	1.50	0.051	0.059
D	0.20	0.40	0.008	0.016
E	0.45	0.75	0.018	0.030
F	0.22	0.35	0.009	0.014
G	0.65 BSC		0.026 BSC	
J	0.07	0.20	0.003	0.008
K	0.50 REF		0.020 REF	
R1	0.08	0.20	0.003	0.008
S	12.00 BSC		0.472 BSC	
S1	6.00 BSC		0.236 BSC	
U	0.09	0.16	0.004	0.006
V	12.00 BSC		0.472 BSC	
V1	6.00 BSC		0.236 BSC	
W	0.20 REF		0.008 REF	
Z	1.00 REF		0.039 REF	
θ	0°	7°	0°	7°
θ1	0°	—	0°	—
θ2	12° REF		12° REF	
θ3	5°	13°	5°	13°

A.4 Modular Development System (MMDS11)

The M68MMDS11 modular development system (MMDS11) is an emulator system for developing embedded systems based on an M68HC11 microcontroller unit (MCU). The MMDS11 provides a bus state analyzer (BSA) and real-time memory windows. The unit's integrated development environment includes an editor, an assembler, user interface, and source-level debug. These features significantly reduce the time necessary to develop and debug an embedded MCU system. The unit's compact size requires a minimum of desk space.

The MMDS11 is one component of Freescale's modular approach to MCU-based product development. This modular approach allows easy configuration of the MMDS11 to fit a wide range of requirements. It also reduces development system cost by allowing the user to purchase only the modular components necessary to support the particular MCU derivative.

MMDS11 features include:

- Real-time, non-intrusive, in-circuit emulation at the MCU’s operating frequency
- Real-time bus state analyzer
  - 8 K x 64 real-time trace buffer
  - Display of real-time trace data as raw data, disassembled instructions, raw data and disassembled instructions, or assembly-language source code
  - Four hardware triggers for commencing trace and to provide breakpoints
  - Nine triggering modes
  - As many as 8190 pre- or post-trigger points for trace data
  - 16 general-purpose logic clips, four of which can be used to trigger the bus state analyzer sequencer
  - 16-bit time tag or an optional 24-bit time tag that reduces the logic clips traced from 16 to eight
- Four data breakpoints (hardware breakpoints)
- Hardware instruction breakpoints over either the 64-Kbyte M68HC11 memory map or over a 1-Mbyte bank switched memory map
- 32 real-time variables, nine of which can be displayed in the variables window. These variables may be read or written while the MCU is running
- 32 bytes of real-time memory can be displayed in the memory window. This memory may be read or written while the MCU is running
- 64 Kbytes of fast emulation memory (SRAM)
- Current-limited target input/output connections
- Six software-selectable oscillator clock sources: five internally generated frequencies and an external frequency via a bus analyzer logic clip
- Command and response logging to MS-DOS® disk files to save session history
- SCRIPT command for automatic execution of a sequence of MMDS11 commands
- Assembly or C-language source-level debugging with global variable viewing
- Host/emulator communications speeds as high as 57,600 baud for quick program loading

® MS-DOS is a registered trademark of Microsoft Corporation.

```
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 '************************************************************************
```

```
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.

```
162 BF62 E72D              STAB   SCCR2,X        Rx and Tx Enabled
163 BF64 CC021B            LDD    #DELAYF        Delay for fast baud rate
164 BF67 ED16              STD    TOC1,X         Set as default delay
165
166                * Send BREAK to signal ready for download
167 BF69 1C2D01            BSET   SCCR2,X $01    Set send break bit
168 BF6C 1E0801FC          BRSET  PORTD,X $01 *  Wait for RxD pin to go low
169 BF70 1D2D01            BCLR   SCCR2,X $01    Clear send break bit
170 BF73
171 BF73 1F2E20FC          BRCLR  SCSR,X $20 *   Wait for RDRF
172 BF77 A62F              LDAA   SCDAT,X        Read data
173                * Data will be $00 if BREAK OR $00 received
174 BF79 2603              BNE    NOTZERO        Bypass JMP if not 0
175 BF7B 7EB600            JMP    EEPMSTR        Jump to EEPROM if it was 0
176 BF7E           NOTZERO EQU    *
177 BF7E 81FF              CMPA   #$FF           $FF will be seen as $FF
178 BF80 2708              BEQ    BAUDOK         If baud was correct
179                * Or else change to ÷104 (÷13 & ÷8) 1200 @ 2MHZ
180 BF82 1C2B33            BSET   BAUD,X $33     Works because $22 -> $33
181 BF85 CC0DB0            LDD    #DELAYS        And switch to slower...
182 BF88 ED16              STD    TOC1,X         delay constant
183 BF8A           BAUDOK  EQU    *
184 BF8A 18CE0000          LDY    #RAMSTR        Point at start of RAM
185
186 BF8E           WAIT    EQU    *
187 BF8E EC16              LDD    TOC1,X         Move delay constant to D
188 BF90           WTLOOP  EQU    *
189 BF90 1E2E2007          BRSET  SCSR,X $20 NEWONE  Exit loop if RDRF set
190 BF94 8F                XGDX                  Swap delay count to X
191 BF95 09                DEX                   Decrement count
192 BF96 8F                XGDX                  Swap back to D
193 BF97 26F7              BNE    WTLOOP         Loop if not timed out
194 BF99 200F              BRA    STAR           Quit download on timeout
195
196 BF9B           NEWONE  EQU    *
197 BF9B A62F              LDAA   SCDAT,X        Get received data
198 BF9D 18A700            STAA   $00,Y          Store to next RAM location
199 BFA0 A72F              STAA   SCDAT,X        Transmit it for handshake
200 BFA2 1808              INY                   Point at next RAM location
201 BFA4 188C0200          CPY    #RAMEND+1      See if past end
202 BFA8 26E4              BNE    WAIT           If not, Get another
203
204 BFAA           STAR    EQU    *
205 BFAA CE1068            LDX    #PROGDEL       Init X with programming delay
206 BFAD 18CED000          LDY    #EPRMSTR       Init Y with EPROM start addr
207 BFB1 7E0000            JMP    RAMSTR         ** EXIT to start of RAM **
208 BFB4
209                ************************************************
210                * Block fill unused bytes with zeros
211
212 BFB4 000000000000        BSZ    $BFD1-*
       000000000000
       000000000000
       000000000000
       0000000000
```

Freescale Semiconductor
Engineering Bulletin

EB188
Rev. 0.1, 07/2005

Enabling the Security Feature on M68HC811E2 Devices with PCbug11 on the M68HC711E9PGMR

By Edgar Saenz
Austin, Texas

Introduction

The PCbug11 software, needed along with the M68HC711E9PGMR to program MC68HC811E2 devices, is available from the download section of the Microcontroller Worldwide Web site

http://www.freescale.com

Retrieve the file pcbug342.exe (a self-extracting archive) from the MCU11 directory.

Some Freescale evaluation board products also are shipped with PCbug11.

NOTE

For specific information about any of the PCbug11 commands, see the appropriate sections in the PCbug11 User's Manual (part number M68PCBUG11/D2), which is available from the Freescale Literature http://www.freescale.com. The file is also on the software download system and is called pcbug11.pdf.

Freescale Semiconductor
Engineering Bulletin

EB296
Rev. 0.1, 07/2005

Programming MC68HC711E9 Devices with PCbug11 and the M68HC11EVBU

By John Bodnar
Austin, Texas

Introduction

The PCbug1software, needed along with the M68HC11EVBU to program MC68HC711E9 devices, is available from the download section of the Microcontroller Worldwide Web site

http://www.freescale.com

Retrieve the file pcbug342.exe (a self-extracting archive) from the MCU11 directory.

Some Freescale evaluation board products also are shipped with PCbug11.

NOTE

For specific information about any of the PCbug11 commands, see the appropriate sections in the PCbug11 User's Manual (part number M68PCBUG11/D2), which is available from the Freescale Literature Distribution Center, as well as the Worldwide Web at http://www.freescale.com. The file is also on the software download system and is called pcbug11.pdf.