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	4MHz
Connectivity	SCI, SPI
Peripherals	POR, WDT
Number of I/O	30
Program Memory Size	-
Program Memory Type	ROMless
EEPROM Size	512 x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	4.75V ~ 5.25V
Data Converters	A/D 8x8b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	68-LCC (J-Lead)
Supplier Device Package	68-PLCC (24.21x24.21)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mchc11f1cfne4

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

Paragraph Title Page

SECTION 9 TIMING SYSTEM

SECTION 10 ANALOG-TO-DIGITAL CONVERTER

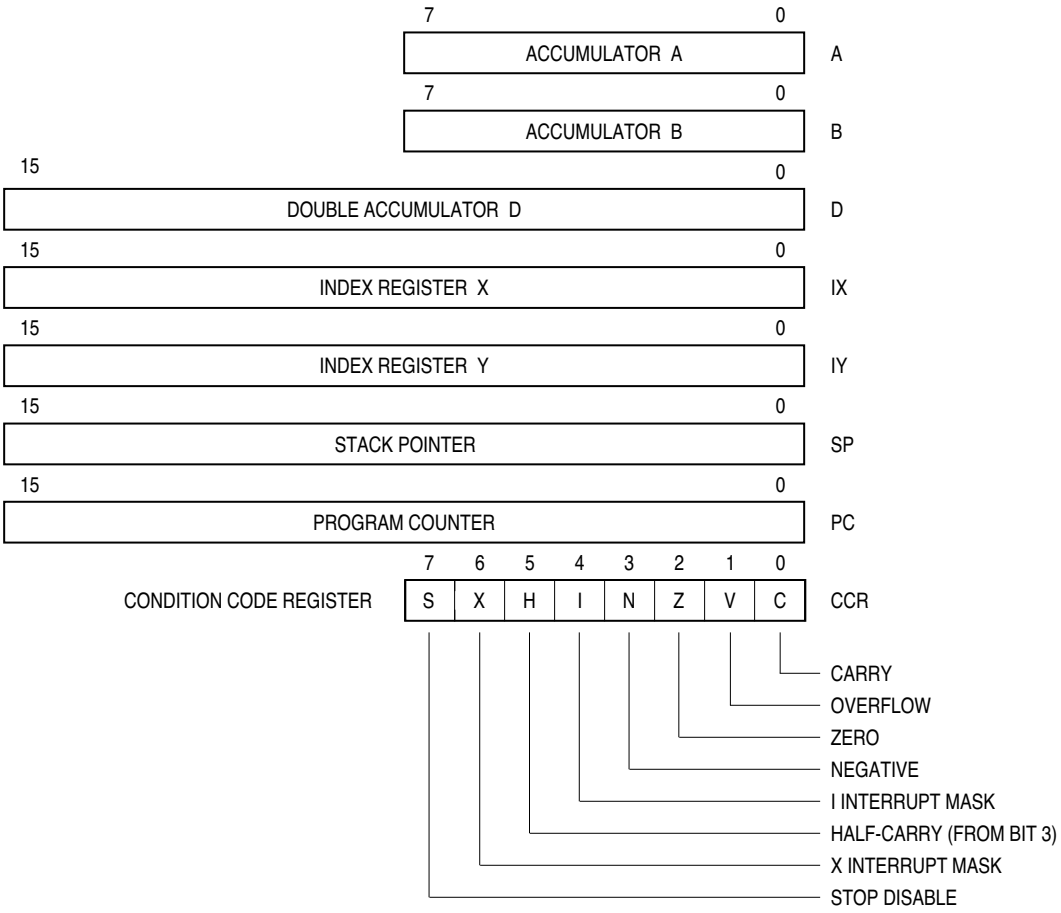


Figure 3-1 Programming Model

3.1.1 Accumulators A, B, and D

Accumulators A and B are general-purpose 8-bit registers that hold operands and results of arithmetic calculations or data manipulations. For some instructions, these two accumulators are treated as a single double-byte (16-bit) accumulator called accumulator D. Although most instructions can use accumulators A or B interchangeably, the following exceptions apply:

The ABX and ABY instructions add the contents of 8-bit accumulator B to the contents of 16-bit register X or Y, but there are no equivalent instructions that use A instead of B.

The TAP and TPA instructions transfer data from accumulator A to the condition code register, or from the condition code register to accumulator A, however, there are no equivalent instructions that use B rather than A.

The decimal adjust accumulator A (DAA) instruction is used after binary-coded decimal (BCD) arithmetic operations, but there is no equivalent BCD instruction to adjust accumulator B.

When an interrupt is recognized, the current instruction finishes normally, the return address (the current value in the program counter) is pushed onto the stack, all of the CPU registers are pushed onto the stack, and execution continues at the address specified by the vector for the interrupt. At the end of the interrupt service routine, an RTI instruction is executed. The RTI instruction causes the saved registers to be pulled off the stack in reverse order. Program execution resumes at the return address.

There are instructions that push and pull the A and B accumulators and the X and Y index registers. These instructions are often used to preserve program context. For example, pushing accumulator A onto the stack when entering a subroutine that uses accumulator A, and then pulling accumulator A off the stack just before leaving the subroutine, ensures that the contents of a register will be the same after returning from the subroutine as it was before starting the subroutine.

3.1.5 Program Counter (PC)

The program counter, a 16-bit register, contains the address of the next instruction to be executed. After reset, the program counter is initialized from one of six possible vectors, depending on operating mode and the cause of reset.

Table 3-1 Reset Vector Comparison

	POR or $\overline{\text{RESET}}$ Pin	**Clock Monitor**	**COP Watchdog**
Normal	$FFFE, F	$FFFC, D	$FFFA, B
Test or Boot	$BFFE, F	$BFFC, D	$BFFA, B

3.1.6 Condition Code Register (CCR)

This 8-bit register contains five condition code indicators (C, V, Z, N, and H), two interrupt masking bits, (I and X) and a stop disable bit (S). In the M68HC11 CPU, condition codes are automatically updated by most instructions. For example, load accumulator A (LDAA) and store accumulator A (STAA) instructions automatically set or clear the N, Z, and V condition code flags. Pushes, pulls, add B to X (ABX), add B to Y (ABY), and transfer/exchange instructions do not affect the condition codes. Refer to **Table 3-2**, which shows what condition codes are affected by a particular instruction.

3.1.6.1 Carry/Borrow (C)

The C bit is set if the arithmetic logic unit (ALU) performs a carry or borrow during an arithmetic operation. The C bit also acts as an error flag for multiply and divide operations. Shift and rotate instructions operate with and through the carry bit to facilitate multiple-word shift operations.

3.1.6.2 Overflow (V)

The overflow bit is set if an operation causes an arithmetic overflow. Otherwise, the V bit is cleared.

Table 3-2 Instruction Set (Sheet 6 of 6)

Mnemonic	Operation	Description	Addressing Mode	Instruction Opcode	Instruction Operand	Instruction Cycles	Condition Codes S	X	H	I	N	Z	V	C
STAB (opr)	Store Accumulator B	B ⇒ M	B DIR B EXT B IND,X B IND,Y	D7 F7 E7 18 E7	dd hh ll ff ff	3 4 4 5	—	—	—	—	Δ	Δ	0	—
STD (opr)	Store Accumulator D	A ⇒ M, B ⇒ M + 1	DIR EXT IND,X IND,Y	DD FD ED 18 ED	dd hh ll ff ff	4 5 5 6	—	—	—	—	Δ	Δ	0	—
STOP	Stop Internal Clocks	—	INH	CF	—	2	—	—	—	—	—	—	—	—
STS (opr)	Store Stack Pointer	SP ⇒ M : M + 1	DIR EXT IND,X IND,Y	9F BF AF 18 AF	dd hh ll ff ff	4 5 5 6	—	—	—	—	Δ	Δ	0	—
STX (opr)	Store Index Register X	IX ⇒ M : M + 1	DIR EXT IND,X IND,Y	DF FF EF CD EF	dd hh ll ff ff	4 5 5 6	—	—	—	—	Δ	Δ	0	—
STY (opr)	Store Index Register Y	IY ⇒ M : M + 1	DIR EXT IND,X IND,Y	18 DF 18 FF 1A EF 18 EF	dd hh ll ff ff	5 6 6 6	—	—	—	—	Δ	Δ	0	—
SUBA (opr)	Subtract Memory from A	A – M ⇒ A	A IMM A DIR A EXT A IND,X A IND,Y	80 90 B0 A0 18 A0	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	Δ	Δ
SUBB (opr)	Subtract Memory from B	B – M ⇒ B	A IMM A DIR A EXT A IND,X A IND,Y	C0 D0 F0 E0 18 E0	ii dd hh ll ff ff	2 3 4 4 5	—	—	—	—	Δ	Δ	Δ	Δ
SUBD (opr)	Subtract Memory from D	D – M : M + 1 ⇒ D	IMM DIR EXT IND,X IND,Y	83 93 B3 A3 18 A3	jj kk dd hh ll ff ff	4 5 6 6 7	—	—	—	—	Δ	Δ	Δ	Δ
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 IND,X IND,Y	7D 6D 18 6D	hh ll ff ff	6 6 7	—	—	—	—	Δ	Δ	0	0
TSTA	Test A for Zero or Minus	A – 0	A INH	4D	—	2	—	—	—	—	Δ	Δ	0	0
TSTB	Test B for Zero or Minus	B – 0	B INH	5D	—	2	—	—	—	—	Δ	Δ	0	0
TSX	Transfer Stack Pointer to X	SP + 1 ⇒ IX	INH	30	—	3	—	—	—	—	—	—	—	—
TSY	Transfer Stack Pointer to Y	SP + 1 ⇒ IY	INH	18 30	—	4	—	—	—	—	—	—	—	—
TXS	Transfer X to Stack Pointer	IX – 1 ⇒ SP	INH	35	—	3	—	—	—	—	—	—	—	—
TYS	Transfer Y to Stack Pointer	IY – 1 ⇒ SP	INH	18 35	—	4	—	—	—	—	—	—	—	—
WAI	Wait for Interrupt	Stack Regs & WAIT	INH	3E	—	**	—	—	—	—	—	—	—	—
XGDX	Exchange D with X	IX ⇒ D, D ⇒ IX	INH	8F	—	3	—	—	—	—	—	—	—	—
XGDY	Exchange D with Y	IY ⇒ D, D ⇒ IY	INH	18 8F	—	4	—	—	—	—	—	—	—	—

4.3.2 Initialization

Because bits in the following registers control the basic configuration of the MCU, an accidental change of their values could cause serious system problems. The protection mechanism, overridden in special operating modes, requires a write to the protected bits only within the first 64 bus cycles after any reset, or only once after each reset. **Table 4-2** summarizes the write access limited registers.

4.3.2.1 CONFIG Register

CONFIG controls the presence and position of the EEPROM in the memory map. CONFIG also enables the COP watchdog timer.

CONFIG — System Configuration Register **$103F**

	Bit 7	6	5	4	3	2	1	Bit 0	
	EE3	EE2	EE1	EE0	—	NOCOP	—	EEON	
RESET:	1	1	1	1	1	P	1	1	Single Chip
	1	1	1	1	1	P(L)	1	1	Bootstrap
	P	P	P	P	1	P	1	P	Expanded
	P	P	P	P	1	P(L)	1	0	Special Test

P indicates a previously programmed bit. P(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 DISR bit in TEST1 register.

The CONFIG register consists of an EEPROM byte and static latches that control the start-up configuration of the MCU. The contents of the EEPROM byte are transferred into static working latches during reset sequences. The operation of the MCU is controlled directly by these latches and not by CONFIG itself. In normal modes, changes to CONFIG do not affect operation of the MCU until after the next reset sequence. When programming, the CONFIG register itself is accessed. When the CONFIG register is read, the static latches are accessed.

These bits can be read at any time. The value read is the one latched into the register from the EEPROM cells during the last reset sequence. A new value programmed into this register cannot be read until after a subsequent reset sequence. Unused bits always read as ones.

In special test mode, the static latches can be written directly at any time. In all modes, CONFIG bits can only be programmed using the EEPROM programming sequence, and are neither readable nor active until latched via the next reset. Refer to **4.4.3 CONFIG Register Programming**.

EE[3:0] — EEPROM Mapping Control

EE[3:0] select the upper four bits of the EEPROM base address. In single-chip and bootstrap modes, EEPROM is forced to $FE00–$FFFF regardless of the value of EE[3:0].

Table 4-12 Chip Select Control Parameter Summary

CSIO1	Enable	IO1EN in CSCTL —	1 = On, off at reset (0)
	Valid	IO1AV in CSGSIZ —	1 = Address valid, 0 = E valid
	Polarity	IO1PL in CSCTL —	1 = Active high, 0 = Active low
	Size	Fixed —	($x060–$x7FF)
	Start Address	$x060 —	“x” is determined by REG[3:0] in INIT
	Stretch	IO1SA–IO1SB in CSSTRH —	0, 1, 2, or 3 E clocks

CSIO2	Enable	IO2EN in CSCTL —	1 = On, off at reset (0)
	Valid	IO2AV in CSGSIZ —	1 = Address valid, 0 = E valid
	Polarity	IO2PL in CSCTL —	1 = Active high, 0 = Active low
	Size	Fixed —	($x800–$xFFF)
	Start Address	$x800 —	“x” is determined by REG[3:0] in INIT
	Stretch	IO2SA–IO2SB in CSSTRH —	0, 1, 2, or 3 E clocks

$\overline{\text{CSPROG}}$	Enable	PCSEN in CSCTL —	1 = On, on after reset in expanded modes off after reset in single-chip modes
	Valid	Fixed (Address valid)	
	Polarity	Fixed (Active low)	
	Size	PSIZA–PSIZB — in CSCTL	0:0 = 64K ($0000–$FFFF) 0:1 = 32K ($8000–$FFFF) 1:0 = 16K ($C000–$FFFF) 1:1 = 8K ($E000–$FFFF)
	Start Address	Fixed (determined by size)	
	Stretch	PSTHA–PSTHB in CSSTRH —	0, 1, 2, or 3 E clocks 1 cycle after reset in expanded mode no delay after reset in all other modes
	Priority	GCSPR in CSCTL —	1 = CSGEN above $\overline{\text{CSPROG}}$ 0 = $\overline{\text{CSPROG}}$ above CSGEN

CSGEN	Enable	Set size to 0K to disable —	1 = CSGEN above $\overline{\text{CSPROG}}$ 0 = $\overline{\text{CSPROG}}$ above CSGEN
	Valid	GAVLD in CSGSIZ —	Address valid or E valid
	Polarity	GNPOL in CSGSIZ —	Active high or low
	Size	GSIZA–GSIZC in CSGSIZ —	Refer to Table 4–12
	Start Address	GA[15:10] in CSGADR	
	Stretch	GSTHA–GSTHB in CSSTRH —	0, 1, 2, or 3 E clocks

1. POR or $\overline{\text{RESET}}$ pin
2. Clock monitor reset
3. COP watchdog reset
4. $\overline{\text{XIRQ}}$ interrupt
5. Illegal opcode interrupt
6. Software interrupt (SWI)

The maskable interrupt sources have the following priority arrangement:

1. $\overline{\text{IRQ}}$
2. Real-time interrupt
3. Timer input capture 1
4. Timer input capture 2
5. Timer input capture 3
6. Timer output compare 1
7. Timer output compare 2
8. Timer output compare 3
9. Timer output compare 4
10. Timer input capture 4/output compare 5
11. Timer overflow
12. Pulse accumulator overflow
13. Pulse accumulator input edge
14. SPI transfer complete
15. SCI system (refer to **Figure 5-5**)

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 only be written while I-bit interrupts are inhibited.

5.3.1 Highest Priority Interrupt and Miscellaneous Register

HPRIO — Highest Priority I-Bit Interrupt and Miscellaneous **$103C**

	Bit 7	6	5	4	3	2	1	Bit 0	
	RBOOT*	SMOD*	MDA*	IRV	PSEL3	PSEL2	PSEL1	PSEL0	
RESET:	0	0	0	0	0	1	0	1	Single Chip
	0	0	1	1	0	1	0	1	Expanded
	1	1	0	0	0	1	0	1	Bootstrap
	0	1	1	1	0	1	0	1	Special Test

*The values of the RBOOT, SMOD, MDA, and IRV reset bits depend on the operating mode selected during power-up. Refer to Table 4–3.

RBOOT — Read Bootstrap ROM

Set to one out of reset in bootstrap mode. Valid while in special modes only. Can be read any time. Can only be written in special modes. Refer to **SECTION 4 OPERATING MODES AND ON-CHIP MEMORY** for more information.

8.5.2 Serial Peripheral Status

SPSR — Serial Peripheral Status Register **$1029**

	Bit 7	6	5	4	3	2	1	Bit 0
	SPIF	WCOL	—	MODF	—	—	—	—
RESET:	0	0	0	0	0	0	0	0

SPIF — SPI Interrupt Complete Flag

SPIF is set upon completion of data transfer between the processor and the external device. If SPIF goes high, and if SPIE is set, a serial peripheral interrupt is generated. To clear the SPIF bit, read the SPSR then access the SPDR. Unless SPSR is read (with SPIF set) first, attempts to write SPDR are inhibited.

WCOL — Write Collision

Clearing the WCOL bit is accomplished by reading the SPSR followed by an access of SPDR. Refer to **8.3.4 Slave Select** and **8.4 SPI System Errors**.

0 = No write collision
1 = Write collision

Bit 5 — Not implemented

Always reads zero

MODF — Mode Fault

To clear the MODF bit, read the SPSR then write to the SPCR. Refer to **8.3.4 Slave Select** and **8.4 SPI System Errors**.

0 = No mode fault
1 = Mode fault

Bits [3:0] — Not implemented

Always read zero

8.5.3 Serial Peripheral Data 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.

SPDR — SPI Data Register **$102A**

Bit 7	6	5	4	3	2	1	Bit 0
Bit 7	6	5	4	3	2	1	Bit 0

SPI is double buffered in and single buffered out.

SECTION 9 TIMING SYSTEM

The M68HC11 timing system is composed of five clock divider chains. The main clock divider chain includes a 16-bit free-running counter, which is driven by a programmable prescaler. The main timer's programmable prescaler provides one of the four clocking rates to drive the 16-bit counter. Two prescaler control bits select the prescale rate.

The prescaler output divides the system clock by 1, 4, 8, or 16. Taps off of this main clocking chain drive circuitry that generates the slower clocks used by the pulse accumulator, the real-time interrupt (RTI), and the computer operating properly (COP) watchdog subsystems, also described in this section. Refer to **Figure 9-1**.

All main timer system activities are referenced to this free-running counter. The counter begins incrementing from $0000 as the MCU comes out of reset, and continues to the maximum count, $FFFF. At the maximum count, the counter rolls over to $0000, sets an overflow flag, and continues to increment. As long as the MCU is running in a normal operating mode, there is no way to reset, change, or interrupt the counting. The capture/compare subsystem features three input capture channels, four output compare channels, and one channel that can be selected to perform either input capture or output compare. Each of the three input capture functions has its own 16-bit input capture register (time capture latch) and each of the output compare functions has its own 16-bit compare register. All timer functions, including the timer overflow and RTI have their own interrupt controls and separate interrupt vectors.

The pulse accumulator contains an 8-bit counter and edge select logic. The pulse accumulator can operate in either event counting mode or gated time accumulation mode. During event counting mode, the pulse accumulator's 8-bit counter increments when a specified edge is detected on an input signal. During gated time accumulation mode, an internal clock source increments the 8-bit counter while an input signal has a predetermined logic level.

The real-time interrupt (RTI) is a programmable periodic interrupt circuit that permits pacing the execution of software routines by selecting one of four interrupt rates.

The COP watchdog clock input ($E \div 2^{15}$) is tapped off of the free-running counter chain. The COP automatically times out unless it is serviced within a specific time by a program reset sequence. If the COP is allowed to time out, a reset is generated, which drives the $\overline{\text{RESET}}$ pin low to reset the MCU and the external system. Refer to **Table 9-1** for crystal related frequencies and periods.

9.2.2 Timer Input Capture Registers

When an edge has been detected and synchronized, the 16-bit free-running counter value is transferred into the input capture register pair as a single 16-bit parallel transfer. Timer counter value captures and timer counter incrementing occur on opposite half-cycles of the phase 2 clock so that the count value is stable whenever a capture occurs. The TICx registers are not affected by reset. Input capture values can be read from a pair of 8-bit read-only registers. A read of the high-order byte of an input capture register pair inhibits a new capture transfer for one bus cycle. If a double-byte read instruction, such as LDD, is used to read the captured value, coherency is assured. When a new input capture occurs immediately after a high-order byte read, transfer is delayed for an additional cycle but the value is not lost.

TIC1–TIC3 — Timer Input Capture **$1010–$1015**

$1010	Bit 15	14	13	12	11	10	9	Bit 8	TIC1 (High)
$1011	Bit 7	6	5	4	3	2	1	Bit 0	TIC1 (Low)
$1012	Bit 15	14	13	12	11	10	9	Bit 8	TIC2 (High)
$1013	Bit 7	6	5	4	3	2	1	Bit 0	TIC2 (Low)
$1014	Bit 15	14	13	12	11	10	9	Bit 8	TIC3 (High)
$1015	Bit 7	6	5	4	3	2	1	Bit 0	TIC3 (Low)

TICx not affected by reset.

9.2.3 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 one. To use it as an output compare register, set the I4/O5 bit to a logic level zero. Refer to **9.6 Pulse Accumulator**.

TI4/O5 — Timer Input Capture 4/Output Compare 5 **$101E, $101F**

$101E	Bit 15	14	13	12	11	10	9	Bit 8	TI4/O5 (High)
$101F	Bit 7	6	5	4	3	2	1	Bit 0	TI4/O5 (Low)

The TI4/O5 register pair resets to ones ($FFFF).

9.3 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.

TCTL1 — Timer Control 1 **$1020**

	Bit 7	6	5	4	3	2	1	Bit 0
	OM2	OL2	OM3	OL3	OM4	OL4	OM5	OL5
RESET:	0	0	0	0	0	0	0	0

OM[2:5] — Output Mode

OL[2:5] — Output Level

These control bit pairs are encoded to specify the action taken after a successful OCx compare. OC5 functions only if the I4/O5 bit in the PACTL register is clear. Refer to **Table 9-3** for the coding.

Table 9-2 Timer Output Compare Configuration

OMx	OLx	Action Taken on Successful Compare
0	0	Timer disconnected from output pin logic
0	1	Toggle OCx output line
1	0	Clear OCx output line to zero
1	1	Set OCx output line to one

9.3.7 Timer Interrupt Mask Register 1

Use this 8-bit register to enable or inhibit the timer input capture and output compare interrupts.

TMSK1 — Timer Interrupt Mask 1 **$1022**

	Bit 7	6	5	4	3	2	1	Bit 0
	OC1I	OC2I	OC3I	OC4I	I4/O5I	IC1I	IC2I	IC3I
RESET:	0	0	0	0	0	0	0	0

OC1I–OC4I — Output Compare x Interrupt Enable

If the OCxI enable bit is set when the OCxF flag bit is set, a hardware interrupt sequence is requested.

I4/O5I — Input Capture 4/Output Compare 5 Interrupt Enable

When I4/O5 in PACTL is one, I4/O5I is the input capture 4 interrupt enable bit. When I4/O5 in PACTL is zero, I4/O5I is the output compare 5 interrupt enable bit.

IC1I–IC3I — Input Capture x Interrupt Enable

If the ICxI enable bit is set when the ICxF flag bit is set, a hardware interrupt sequence is requested.

NOTE

Bits in TMSK1 correspond bit for bit with flag bits in TFLG1. Ones in TMSK1 enable the corresponding interrupt sources.

9.3.8 Timer Interrupt Flag Register 1

Bits in this register indicate when timer system events have occurred. Coupled with the bits of TMSK1, the bits of TFLG1 allow the timer subsystem to operate in either a polled or interrupt driven system. Each bit of TFLG1 corresponds to a bit in TMSK1 in the same position.

TFLG1 — Timer Interrupt Flag 1 **$1023**

	Bit 7	6	5	4	3	2	1	Bit 0
	OC1F	OC2F	OC3F	OC4F	I4/O5F	IC1F	IC2F	IC3F
RESET:	0	0	0	0	0	0	0	0

Clear flags by writing a one to the corresponding bit position(s).

OC1F–OC4F — Output Compare x Flag
Set each time the counter matches output compare x value

I4/O5F — Input Capture 4/Output Compare 5 Flag
Set by IC4 or OC5, depending on the function enabled by I4/O5 bit in PACTL

IC1F–IC3F — Input Capture x Flag
Set each time a selected active edge is detected on the ICx input line

9.3.9 Timer Interrupt Mask Register 2

Use this 8-bit register to enable or inhibit timer overflow and real-time interrupts. The timer prescaler control bits are included in this register.

TMSK2 — Timer Interrupt Mask 2 **$1024**

	Bit 7	6	5	4	3	2	1	Bit 0
	TOI	RTII	PAOVI	PAII	—	—	PR1	PR0
RESET:	0	0	0	0	0	0	0	0

TOI — Timer Overflow Interrupt Enable
0 = TOF interrupts disabled
1 = Interrupt requested when TOF is set to one

RTII — Real-Time Interrupt Enable
Refer to **9.4 Real-Time Interrupt**.

PAOVI — Pulse Accumulator Overflow Interrupt Enable
Refer to **9.6.3 Pulse Accumulator Status and Interrupt Bits**.

PAII — Pulse Accumulator Input Edge Interrupt Enable
Refer to **9.6.3 Pulse Accumulator Status and Interrupt Bits**.

PR[1:0] — Timer Prescaler Select
These bits are used to select the prescaler divide-by ratio. In normal modes, PR[1:0] can only be written once, and the write must be within 64 cycles after reset. Refer to **Table 9-1** and **Table 9-4** for specific timing values.

APPENDIX A ELECTRICAL CHARACTERISTICS

This appendix contains electrical parameters for the MC68HC11F1 microcontroller.

Table A-1 Maximum Ratings

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	– 0.3 to + 7.0	V
Input Voltage	V_{in}	– 0.3 to + 7.0	V
Operating Temperature Range MC68HC11F1 MC68HC11F1C MC68HC11F1V MC68HC11F1M	T_A	T_L to T_H 0 to + 70 – 40 to + 85 – 40 to + 105 – 40 to + 125	°C
Storage Temperature Range	T_{stg}	– 55 to + 150	°C
Current Drain per Pin* Excluding V_{DD}, V_{SS}, AV_{DD}, V_{RH}, and V_{RL}	I_D	25	mA

*One pin at a time, observing maximum power dissipation limits.

Internal circuitry protects the inputs against damage caused by high static voltages or electric fields; however, normal precautions are necessary to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. Extended operation at the maximum ratings can adversely affect device reliability. Tying unused inputs to an appropriate logic voltage level (either GND or V_{DD}) enhances reliability of operation.

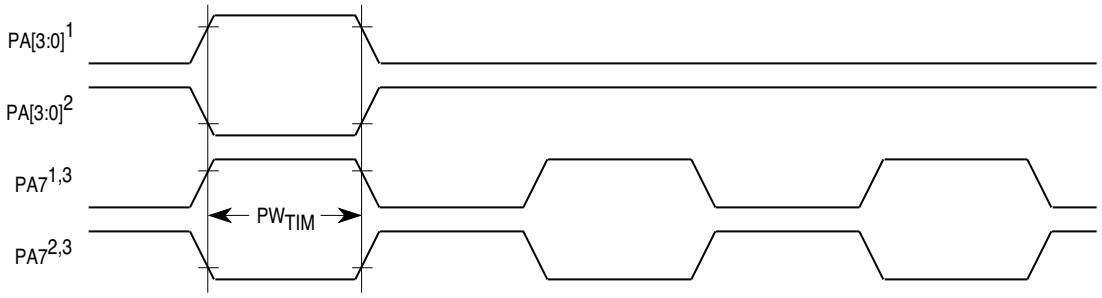
Table A-4 Control Timing

V_{DD} = 5.0 Vdc ± 5%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H

Characteristic	Symbol	2.0 MHz		3.0 MHz		4.0 MHz		Unit
		Min	Max	Min	Max	Min	Max	
Frequency of Operation	f_o	dc	2.0	dc	3.0	dc	4.0	MHz
E-Clock Period	t_{cyc}	500	—	333	—	250	—	ns
Crystal Frequency	f_{XTAL}	—	8.0	—	12.0	—	16.0	MHz
External Oscillator Frequency	4 f_o	dc	8.0	dc	12.0	dc	16.0	MHz
Processor Control Setup Time t_{PCSU} = 1/4 t_{cyc} + 50 ns	t_{PCSU}	175	—	133	—	113	—	ns
Reset Input Pulse Width (Notes 2, 3) (To Guarantee External Reset Vector) (Minimum Input Time; Can Be Preempted by Internal Reset)	PW_{RSTL}	16 1	— —	16 1	— —	16 1	— —	t_{cyc} t_{cyc}
Mode Programming Setup Time	t_{MPS}	2	—	2	—	2	—	t_{cyc}
Mode Programming Hold Time	t_{MPH}	10	—	10	—	10	—	ns
Interrupt Pulse Width, $\overline{IRQ}$ Edge-Sensitive Mode PW_{IRQ} = t_{cyc} + 20 ns	PW_{IRQ}	520	—	353	—	270	—	ns
Wait Recovery Startup Time	t_{WRS}	—	4	—	4	—	4	t_{cyc}
Timer Pulse Width, Input Capture Pulse Accumulator Input PW_{TIM} = t_{cyc} + 20 ns	PW_{TIM}	520	—	353	—	270	—	ns

NOTES:

1. All timing is shown with respect to 20% V_{DD} and 70% V_{DD}, unless otherwise noted.
2. $\overline{RESET}$ is recognized during the first clock cycle it is held low. Internal circuitry then drives the pin low for four clock cycles, releases the pin, and samples the pin level two cycles later to determine the source of the interrupt. Refer to **SECTION 5 RESETS AND INTERRUPTS** for further detail.
3. PW_{RSTL} = 8 t_{cyc} minimum on mask set C94R only.



NOTES:

1. Rising edge sensitive input.
2. Falling edge sensitive input.
3. Maximum pulse accumulator clocking rate is E-clock frequency divided by 2.

Figure A-2 Timer Inputs

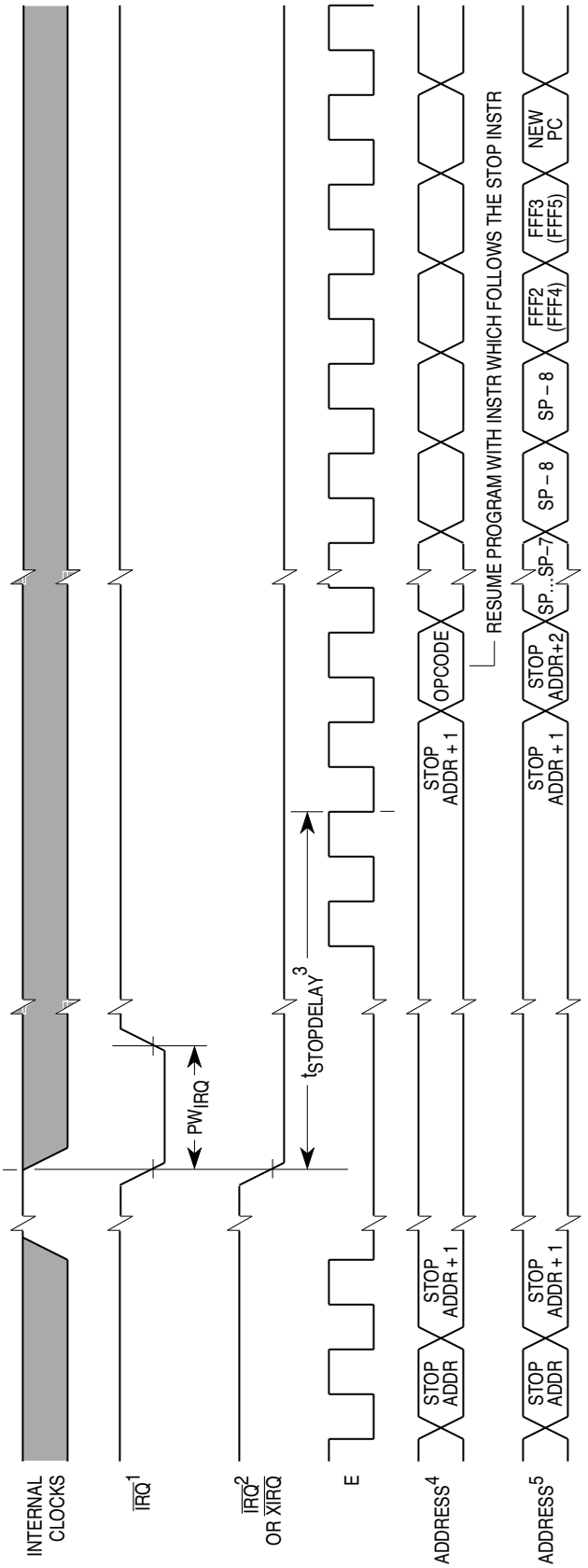


Figure A-4 STOP Recovery Timing Diagram

Table A-9 EEPROM Characteristics

V_{DD} = 5.0 Vdc ± 10%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H

Characteristic		Temperature Range			Unit
		0 to 70, – 40 to 85	**–40 to 105**	**–40 to 125**	**° C**
Programming Time (Note 1)	<1.0 MHz, RCO Enabled	10	15	20	ms
	1.0 to 2.0 MHz, RCO Disabled	20	Must use RCO	Must use RCO	
	≥ 2.0 MHz (or Anytime RCO Enabled)	10	15	20	
Erase Time (Note 1)	Byte, Row and Bulk	10	10	10	ms
Write/Erase Endurance (Note 2)		10,000	10,000	10,000	Cycles
Data Retention (Note 2)		10	10	10	Years

NOTES:

1. The RC oscillator (RCO) must be enabled (by setting the CSEL bit in the OPTION register) for EEPROM programming and erasure when the E-clock frequency is below 1.0 MHz.
2. Refer to Reliability Monitor Report (current quarterly issue) for current failure rate information.

APPENDIX CDEVELOPMENT SUPPORT

C.1 MC68HC11F1 Development Tools

The following table and text provide a reference to development tools for the MC68HC11F1 microcontrollers. For more complete information refer to the appropriate manual for each system.

Table C-1 MC68HC11F1 Development Tools

Device	Evaluation Systems	Modular Development Systems
MC68HC11F1	M68HC11F1EVS	MMDS11*

* For MC68HC11F1 support, the MMDS11 must be used with an MC68HC11F1 emulator module.

C.2 MC68HC11EVS — Evaluation System

The EVS is an economical tool for designing, debugging, and evaluating target systems based on the MC68HC11F1 MCU. The two printed circuit boards that comprise the EVS are the MC68HC11F1EM emulator module (EM) and the M68HC11PFB platform board (PFB).

- Monitor/debugger firmware
- One-line assembler/disassembler
- Host computer download capability
- Dual memory maps:
  - 64 Kbyte monitor map that includes 16 Kbytes of monitor EPROM
  - MC68HC11F1 user map that includes 64 Kbytes of emulation RAM
- OTPROM, EPROM, and EEPROM MCU programmer
- MCU extension I/O port for single-chip, expanded, and special-test operation modes
- RS-232C terminal and host I/O ports
- Logic analyzer connector

C.3 M68MMDS11 — Modular Development System for M68HC11 Devices

The M68MMDS11 Freescale Modular Development System (MMDS11) is a tool for developing embedded systems based on M68HC11 MCUs. The MMDS11 is an emulator system that provides an on-screen bus state analyzer and real-time memory monitoring windows. An integrated design environment includes an editor, an assembler, the user interface, and source-level debug capability. These features significantly reduce the time necessary to develop and debug an embedded MCU system. The compact unit requires minimum space.

- Real-time, non-intrusive, in-circuit emulation
- Assembly-language source-level debugging