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	Active
Core Processor	CPU16
Core Size	16-Bit
Speed	16MHz
Connectivity	EBI/EMI, SCI, SPI
Peripherals	POR, PWM, WDT
Number of I/O	16
Program Memory Size	-
Program Memory Type	ROMless
EEPROM Size	-
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2.7V ~ 3.6V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	144-LQFP
Supplier Device Package	144-LQFP (20x20)
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=mc68hc16z1cag16

TABLE OF CONTENTS
(Continued)

Paragraph Title Page

SECTION 8
ANALOG-TO-DIGITAL CONVERTER

SECTION 4
CENTRAL PROCESSOR UNIT

This section is an overview of the central processor unit (CPU16). For detailed information, refer to the *CPU16 Reference Manual* (CPU16RM/AD).

4.1 General

The CPU16 provides compatibility with the M68HC11 CPU and also provides additional capabilities associated with 16- and 32-bit data sizes, 20-bit addressing, and digital signal processing. CPU16 registers are an integral part of the CPU and are not addressed as memory locations.

The CPU16 treats all peripheral, I/O, and memory locations as parts of a linear one Megabyte address space. There are no special instructions for I/O that are separate from instructions for addressing memory. Address space is made up of sixteen 64-Kbyte banks. Specialized bank addressing techniques and support registers provide transparent access across bank boundaries.

The CPU16 interacts with external devices and with other modules within the microcontroller via a standardized bus and bus interface. There are bus protocols used for memory and peripheral accesses, as well as for managing a hierarchy of interrupt priorities.

4.2 Register Model

Figure 4-1 shows the CPU16 register model. Refer to the paragraphs that follow for a detailed description of each register.

4.6.8 Use of CPU16 Indexed Mode to Replace M68HC11 Direct Mode

In M68HC11 systems, the direct addressing mode can be used to perform rapid accesses to RAM or I/O mapped from $0000 to $00FF. The CPU16 uses the first 512 bytes of bank 0 for exception vectors. To provide an enhanced replacement for the M68HC11's direct addressing mode, the ZK field and index register Z have been assigned reset initialization vectors. By resetting the ZK field to a chosen page and using indexed mode addressing, a programmer can access useful data structures anywhere in the address map.

4.7 Instruction Set

The CPU16 instruction set is based on the M68HC11 instruction set, but the opcode map has been rearranged to maximize performance with a 16-bit data bus. Most M68HC11 code can run on the CPU16 following reassembly. The user must take into account changed instruction times, the interrupt mask, and the changed interrupt stack frame (refer to *Transporting M68HC11 Code to M68HC16 Devices*,Freescale Programming Note M68HC16PN01/D, for more information).

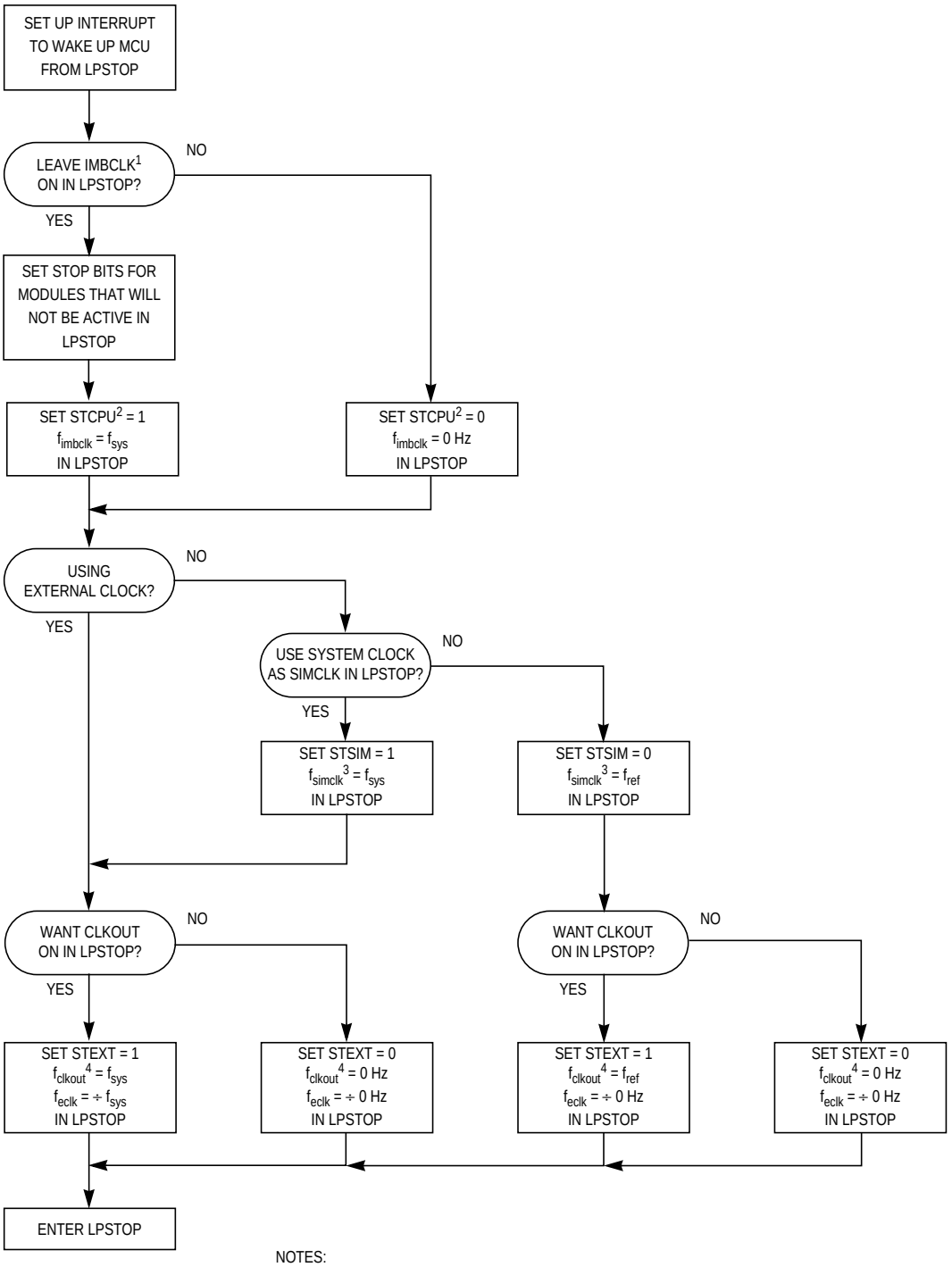
4.7.1 Instruction Set Summary

Table 4-2 is a quick reference to the entire CPU16 instruction set. Refer to the *CPU16 Reference Manual* (CPU16RM/AD) for detailed information about each instruction, assembler syntax, and condition code evaluation. **Table 4-3** provides a key to the table nomenclature.

Table 4-4 CPU16 Implementation of M68HC11 CPU Instructions

M68HC11 Instruction	CPU16 Implementation
BHS	BCC only
BLO	BCS only
BSR	Generates a different stack frame
CLC	Replaced by ANDP
CLI	Replaced by ANDP
CLV	Replaced by ANDP
DES	Replaced by AIS
DEX	Replaced by AIX
DEY	Replaced by AIY
INS	Replaced by AIS
INX	Replaced by AIX
INY	Replaced by AIY
JMP	IND8 and EXT addressing modes replaced by IND20 and EXT20 modes
JSR	IND8 and EXT addressing modes replaced by IND20 and EXT20 modes. Generates a different stack frame
LSL, LSLD	Use ASL instructions[1]
PSHX	Replaced by PSHM
PSHY	Replaced by PSHM
PULX	Replaced by PULM
PULY	Replaced by PULM
RTI	Reloads PC and CCR only
RTS	Uses two-word stack frame
SEC	Replaced by ORP
SEI	Replaced by ORP
SEV	Replaced by ORP
STOP	Replaced by LPSTOP
TAP	CPU16 CCR bits differ from M68HC11 CPU16 interrupt priority scheme differs from M68HC11
TPA	CPU16 CCR bits differ from M68HC11 CPU16 interrupt priority scheme differs from M68HC11
TSX	Adds two to SK : SP before transfer to XK : IX
TSY	Adds two to SK : SP before transfer to YK : IY
TXS	Subtracts two from XK : IX before transfer to SK : SP
TXY	Transfers XK field to YK field
TYS	Subtracts two from YK : IY before transfer to SK : SP
TYX	Transfers YK field to XK field
WAI	Waits indefinitely for interrupt or reset Generates a different stack frame

NOTES:
1. Freescale assemblers automatically translate ASL mnemonics.



NOTES:

1. IMBCLK IS THE CLOCK USED BY THE CPU16L, SIML, ADC, MCCI, AND THE GPT.
2. WHEN STCPU = 1, THE CPU16L IS SHUT DOWN IN LPSTOP. ALL OTHER MODULES WILL REMAIN ACTIVE UNLESS THE STOP BITS IN THEIR MODULE CONFIGURATION REGISTERS ARE SET PRIOR TO ENTERING LPSTOP.
3. THE SIMCLK IS USED BY THE PIT, IRQ, AND INPUT BLOCKS OF THE SIML.
4. CLKOUT CONTROL DURING LPSTOP IS OVERRIDDEN BY THE EXOFF BIT IN SIMCR. IF EXOFF = 1, THE CLKOUT PIN IS ALWAYS IN A HIGH-IMPEDANCE STATE AND STEXT HAS NO EFFECT IN LPSTOP. IF EXOFF = 0, CLKOUT IS CONTROLLED BY STEXT IN LPSTOP. WHEN STCPU = 1, THE CPU16L IS DISABLED IN LPSTOP, BUT ALL OTHER MODULES REMAIN ACTIVE OR STOPPED ACCORDING TO THE SETTING.

SIML LPSTOP FLOWCHART

Figure 5-7 SIML LPSTOP Flowchart

Table 5-15 Effect of $\overline{\text{DSACK}}$ Signals

$\overline{\text{DSACK1}}$	$\overline{\text{DSACK0}}$	Result
1	1	Insert wait states in current bus cycle
1	0	Complete cycle — Data bus port size is eight bits
0	1	Complete cycle — Data bus port size is sixteen bits
0	0	Reserved

If the CPU is executing an instruction that reads a long-word operand from a 16-bit port, the MCU latches the 16 bits of valid data and then runs another bus cycle to obtain the other 16 bits. The operation for an 8-bit port is similar, but requires four read cycles. The addressed device uses the $\overline{\text{DSACK}}$ signals to indicate the port width. For instance, a 16-bit external device always returns $\overline{\text{DSACK}}$ for a 16-bit port (regardless of whether the bus cycle is a byte or word operation).

Dynamic bus sizing requires that the portion of the data bus used for a transfer to or from a particular port size be fixed. A 16-bit port must reside on data bus bits [15:0], and an 8-bit port must reside on data bus bits [15:8]. This minimizes the number of bus cycles needed to transfer data and ensures that the MCU transfers valid data.

The MCU always attempts to transfer the maximum amount of data on all bus cycles. For a word operation, it is assumed that the port is 16 bits wide when the bus cycle begins.

Operand bytes are designated as shown in **Figure 5-11**. OP[0:3] represent the order of access. For instance, OP0 is the most significant byte of a long-word operand, and is accessed first, while OP3, the least significant byte, is accessed last. The two bytes of a word-length operand are OP0 (most significant) and OP1. The single byte of a byte-length operand is OP0.

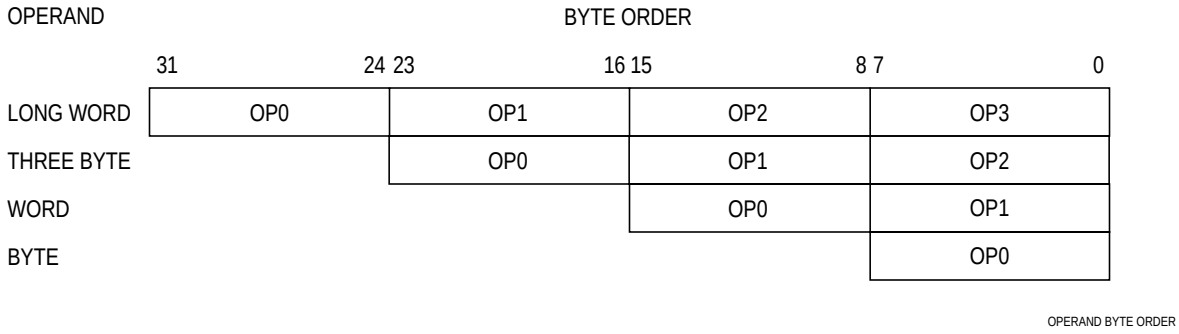


Figure 5-11 Operand Byte Order

SECTION 6
STANDBY RAM MODULE

The standby RAM (SRAM) module consists of a fixed-location control register block and an array of fast (two clock) static RAM that may be mapped to a user specified location in the system memory map. Array size depends on the M68HC16, M68CK16, and M68CM16 Z-series version. Refer to **Table 6-1** for appropriate SRAM array size. The SRAM is especially useful for system stacks and variable storage.

Table 6-1 SRAM Configuration

Z-Series Device	Array Size
MC68HC16Z1 MC68CK16Z1 MC68CM16Z1 MC68HC16Z4 MC68CK16Z4	1 Kbyte
MC68HC16Z2	2 Kbytes
MC68HC16Z3	4 Kbytes

The SRAM can be mapped to any address that is a multiple of the array size so long as SRAM boundaries do not overlap the module control registers (overlap makes the registers inaccessible). Data can be read/written in bytes, words or long words. SRAM is powered by V_{DD} in normal operation. During power-down, SRAM contents can be maintained by power from the V_{STBY} input. Power switching between sources is automatic.

6.1 SRAM Register Block

There are four SRAM control registers: the RAM module configuration register (RAMMCR), the RAM test register (RAMTST), and the RAM array base address registers (RAMBAH/RAMBAL).

The module mapping bit (MM) in the SIM configuration register (SIMCR) defines the most significant bit (ADDR23) of the IMB address for each M68HC16, M68CK16, and M68CM16 Z-series module. Because ADDR[23:20] are driven to the same value as ADDR19, MM must be set to one. If MM is cleared, IMB modules are inaccessible. For more information about how the state of MM affects the system, refer to **5.2.1 Module Mapping**.

The SRAM control register consists of eight bytes, but not all locations are implemented. Unimplemented register addresses are read as zeros, and writes have no effect. Refer to **D.3 Standby RAM Module** for the register block address map and register bit/field definitions.

SECTION 7
MASKED ROM MODULE

The masked ROM module (MRM) is only available with the MC68HC16Z2 and the MC68HC16Z3. The MRM consists of a fixed-location control register block and an 8-Kbyte mask-programmed read-only memory array that can be mapped to any 8-Kbyte boundary in the system memory map. The MRM can be programmed to insert wait states to match slower external development memory. Access time depends upon the number of wait states specified, but can be as fast as two clock cycles. The MRM can be used for program accesses only, or for program and data accesses. Data can be read in bytes, words or long words. The MRM can be configured to support system bootstrap during reset.

7.1 MRM Register Block

There are three MRM control registers: the masked ROM module configuration register (MRMCR), and the ROM array base address registers (ROMBAH and ROMBAL). In addition, the MRM register block contains the signature registers (RSIGHI and RSIGLO), and ROM bootstrap words (ROMBS[0:3]).

The module mapping bit (MM) in the SIM configuration register (SIMCR) defines the most significant bit (ADDR23) of the IMB address for each M68HC16, M68CK16, and M68CM16 Z-series module. Because ADDR[23:20] are driven to the same value as ADDR19, MM must be set to one. If MM is cleared, IMB modules are inaccessible. For more information about how the state of MM affects the system, refer to **5.2.1 Module Mapping**.

The MRM control register block consists of 32 bytes, but not all locations are implemented. Unimplemented register addresses are read as zeros, and writes have no effect. Refer to **D.4 Masked ROM Module** for the register block address map and register bit/field definitions.

7.2 MRM Array Address Mapping

Base address registers ROMBAH and ROMBAL are used to specify the ROM array base address in the memory map. Although the base address loaded into ROMBAH and ROMBAL during reset is mask-programmed as user-specified, these registers can be written after reset to change the default array address if the base address lock bit (LOCK in MRMCR) is not masked to a value of one.

NOTE

In the CPU16, ADDR[23:20] follow the logic state of ADDR19. The MRM array must not be mapped to addresses $7FF000–$7FFFFF, which are inaccessible to the CPU16. If mapped to these addresses, the array remains inaccessible until a reset occurs, or it is remapped outside of this range.

8.6.4 Comparator

The comparator indicates whether each approximation output from the RC DAC array during resolution is higher or lower than the sampled input voltage. Comparator output is fed to the digital control logic, which sets or clears each bit in the successive approximation register in sequence, MSB first.

8.7 Digital Control Subsystem

The digital control subsystem includes control and status registers, clock and prescaler control logic, channel and reference select logic, conversion sequence control logic, and the successive approximation register.

The subsystem controls the multiplexer and the output of the RC array during sample and conversion periods, stores the results of comparison in the successive-approximation register, then transfers results to the result registers.

8.7.1 Control/Status Registers

There are two control registers (ADCTL0, ADCTL1) and one status register (ADCSTAT). ADCTL0 controls conversion resolution, sample time, and clock/prescaler value. ADCTL1 controls analog input selection, conversion mode, and initiation of conversion. A write to ADCTL0 aborts the current conversion sequence and halts the ADC. Conversion must be restarted by writing to ADCTL1. A write to ADCTL1 aborts the current conversion sequence and starts a new sequence with parameters altered by the write. ADCSTAT shows conversion sequence status, conversion channel status, and conversion completion status.

The following paragraphs are a general discussion of control function. **D.5 Analog-to-Digital Converter Module** shows the ADC address map and discusses register bits and fields.

8.7.2 Clock and Prescaler Control

The ADC clock is derived from the system clock by a programmable prescaler. ADC clock period is determined by the value of the PRS field in ADCTL0. The prescaler has two stages. The first stage is a 5-bit modulus counter. It divides the system clock by any value from two to 32 (PRS[4:0] = %00000 to %11111). The second stage is a divide-by-two circuit. **Table 8-3** shows prescaler output values.

In Figure 8-10, R_F and C_F comprise the user's external filter circuit. C_S is the internal sample capacitor. Each channel has its own capacitor. C_S is never precharged; it retains the value of the last sample. V_l is an internal voltage source used to precharge the DAC capacitor array (C_{DAC}) before each sample. The value of this supply is V_{DDA}/2, or 2.5 volts for 5-volt operation.

The following paragraphs provide a simplified description of the interaction between the ADC and the user's external circuitry. This circuitry is assumed to be a simple RC low-pass filter passing a signal from a source to the ADC input pin. The following simplifying assumptions are made:

- The source impedance is included with the series resistor of the RC filter.
- The external capacitor is perfect (no leakage, no significant dielectric absorption characteristics, etc.)
- All parasitic capacitance associated with the input pin is included in the value of the external capacitor.
- Inductance is ignored.
- The “on” resistance of the internal switches is zero ohms and the “off” resistance is infinite.

8.8.6.1 Settling Time for the External Circuit

The values for R_F and C_F in the user's external circuitry determine the length of time required to charge C_F to the source voltage level (V_{SRC}).

At time t = 0, S1 in Figure 8-10 closes. S2 is open, disconnecting the internal circuitry from the external circuitry. Assume that the initial voltage across CF is zero. As CF charges, the voltage across it is determined by the following equation, where t is the total charge time:

$$V_{CF} = V_{SRC}(1 - e^{-t/R_F C_F})$$

When t = 0, the voltage across C_F = 0. As t approaches infinity, V_{CF} will equal V_{SRC}. (This assumes no internal leakage.) With 10-bit resolution, 1/2 of a count is equal to 1/2048 full-scale value. Assuming worst case (V_{SRC} = full scale), Table 8-10 shows the required time for C_F to charge to within 1/2 of a count of the actual source voltage during 10-bit conversions. Table 8-10 is based on the RC network in Figure 8-10.

NOTE

The following times are completely independent of the A/D converter architecture (assuming the ADC is not affecting the charging).

Table 9-3 Bits Per Transfer

BITS[3:0]	Bits Per Transfer
0000	16
0001	Reserved
0010	Reserved
0011	Reserved
0100	Reserved
0101	Reserved
0110	Reserved
0111	Reserved
1000	8
1001	9
1010	10
1011	11
1100	12
1101	13
1110	14
1111	15

Delay after transfer can be used to provide a peripheral deselect interval. A delay can also be inserted between consecutive transfers to allow serial A/D converters to complete conversion. Writing a value to DTL[7:0] in SPCR1 specifies a delay period. The DT bit in each command RAM byte determines whether the standard delay period (DT = 0) or the user-specified delay period (DT = 1) is used. The following expression is used to calculate the delay:

$$\text{Delay after Transfer} = \frac{32 \times \text{DTL}[7:0]}{f_{sys}} \text{ if DT = 1}$$

where DTL equals {1, 2, 3,..., 255}.

A zero value for DTL[7:0] causes a delay-after-transfer value of $8192/f_{sys}$.

$$\text{Standard Delay after Transfer} = \frac{17}{f_{sys}} \text{ if DT = 0}$$

Adequate delay between transfers must be specified for long data streams because the QSPI requires time to load a transmit RAM entry for transfer. Receiving devices need at least the standard delay between successive transfers. If the system clock is operating at a slower rate, the delay between transfers must be increased proportionately.

Table A-16 16.78-MHz AC Timing (Continued)

(V_{DD} and V_{DDSYN} = 5.0 Vdc ± 10 %, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)[1]

Num	Characteristic	Symbol	Min	Max	Unit
30	CLKOUT Low to Data In Invalid (Fast Cycle Hold)[7]	t_{CLDI}	15	—	ns
30A	CLKOUT Low to Data In High Impedance[7]	t_{CLDH}	—	90	ns
31	$\overline{DSACK[1:0]}$ Asserted to Data In Valid[9]	t_{DADI}	—	50	ns
33	Clock Low to $\overline{BG}$ Asserted/Negated	t_{CLBAN}	—	29	ns
35	$\overline{BR}$ Asserted to $\overline{BG}$ Asserted[10]	t_{BRAGA}	1	—	t_{cyc}
37	$\overline{BGACK}$ Asserted to $\overline{BG}$ Negated	t_{GAGN}	1	2	t_{cyc}
39	$\overline{BG}$ Width Negated	t_{GH}	2	—	t_{cyc}
39A	$\overline{BG}$ Width Asserted	t_{GA}	1	—	t_{cyc}
46	R/$\overline{W}$ Width Asserted (Write or Read)	t_{RWA}	150	—	ns
46A	R/$\overline{W}$ Width Asserted (Fast Write or Read Cycle)	t_{RWAS}	90	—	ns
47A	Asynchronous Input Setup Time $\overline{BR}$, $\overline{BGACK}$, $\overline{DSACK[1:0]}$, $\overline{BERR}$, $\overline{AVEC}$, $\overline{HALT}$	t_{AIST}	5	—	ns
47B	Asynchronous Input Hold Time	t_{AIHT}	15	—	ns
48	$\overline{DSACK[1:0]}$ Asserted to $\overline{BERR}$, $\overline{HALT}$ Asserted[11]	t_{DABA}	—	30	ns
53	Data Out Hold from Clock High	t_{DOCH}	0	—	ns
54	Clock High to Data Out High Impedance	t_{CHDH}	—	28	ns
55	R/$\overline{W}$ Asserted to Data Bus Impedance Change	t_{RADC}	40	—	ns
70	Clock Low to Data Bus Driven (Show Cycle)	t_{SCLDD}	0	29	ns
71	Data Setup Time to Clock Low (Show Cycle)	t_{SCLDS}	15	—	ns
72	Data Hold from Clock Low (Show Cycle)	t_{SCLDH}	10	—	ns
73	$\overline{BKPT}$ Input Setup Time	t_{BKST}	15	—	ns
74	$\overline{BKPT}$ Input Hold Time	t_{BKHT}	10	—	ns
75	Mode Select Setup Time, DATA[15:0], MODCLK, $\overline{BKPT}$ pins	t_{MSS}	20	—	t_{cyc}
76	Mode Select Hold Time, DATA[15:0], MODCLK, $\overline{BKPT}$ pins	t_{MSH}	0	—	ns
77	$\overline{RESET}$ Assertion Time[12]	t_{RSTA}	4	—	t_{cyc}
78	$\overline{RESET}$ Rise Time[13]	t_{RSTR}	—	10	t_{cyc}
100	CLKOUT High to Phase 1 Asserted[14]	t_{CHP1A}	3	40	ns
101	CLKOUT High to Phase 2 Asserted[14]	t_{CHP2A}	3	40	ns
102	Phase 1 Valid to $\overline{AS}$ or $\overline{DS}$ Asserted[14]	t_{P1VSA}	10	—	ns
103	Phase 2 Valid to $\overline{AS}$ or $\overline{DS}$ Asserted[14]	t_{P2VSN}	10	—	ns
104	$\overline{AS}$ or $\overline{DS}$ Valid to Phase 1 Negated[14]	t_{SAP1N}	10	—	ns
105	$\overline{AS}$ or $\overline{DS}$ Negated to Phase 2 Negated[14]	t_{SNP2N}	10	—	ns

NOTES:
1. Refer to notes in **Table A-18**.

Table A-21 20.97-MHz Background Debug Mode Timing

(V_{DD} and V_{DDSYN} = 5.0 Vdc ± 5%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)[1]

Num	Characteristic	Symbol	Min	Max	Unit
B0	DSI Input Setup Time	t_{DSISU}	15	—	ns
B1	DSI Input Hold Time	t_{DSIH}	10	—	ns
B2	DSCLK Setup Time	t_{DSCSU}	15	—	ns
B3	DSCLK Hold Time	t_{DSCH}	10	—	ns
B4	DSO Delay Time	t_{DSOD}	—	25	ns
B5	DSCLK Cycle Time	t_{DSCCYC}	2	—	t_{cyc}
B6	CLKOUT High to FREEZE Asserted/Negated	t_{FRZAN}	—	50	ns
B7	CLKOUT High to IPIPE1 High Impedance	t_{IFZ}	—	50	ns
B8	CLKOUT High to IPIPE1 Valid	t_{IF}	—	50	ns
B9	DSCLK Low Time	t_{DSCLO}	1	—	t_{cyc}
B10	IPIPE1 High Impedance to FREEZE Asserted	t_{IPFA}	TBD	—	t_{cyc}
B11	FREEZE Negated to IPIPE[0:1] Active	t_{FRIP}	TBD	—	t_{cyc}

NOTES:
1. All AC timing is shown with respect to V_{IH}/V_{IL} levels unless otherwise noted.

Table A-22 25.17-MHz Background Debug Mode Timing

(V_{DD} and V_{DDSYN} = 5.0 Vdc ± 5%, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)[1]

Num	Characteristic	Symbol	Min	Max	Unit
B0	DSI Input Setup Time	t_{DSISU}	10	—	ns
B1	DSI Input Hold Time	t_{DSIH}	5	—	ns
B2	DSCLK Setup Time	t_{DSCSU}	10	—	ns
B3	DSCLK Hold Time	t_{DSCH}	5	—	ns
B4	DSO Delay Time	t_{DSOD}	—	20	ns
B5	DSCLK Cycle Time	t_{DSCCYC}	2	—	t_{cyc}
B6	CLKOUT High to FREEZE Asserted/Negated	t_{FRZAN}	—	20	ns
B7	CLKOUT High to IPIPE1 High Impedance	t_{IFZ}	—	20	ns
B8	CLKOUT High to IPIPE1 Valid	t_{IF}	—	20	ns
B9	DSCLK Low Time	t_{DSCLO}	1	—	t_{cyc}
B10	IPIPE1 High Impedance to FREEZE Asserted	t_{IPFA}	TBD	—	t_{cyc}
B11	FREEZE Negated to IPIPE[0:1] Active	t_{FRIP}	TBD	—	t_{cyc}

NOTES:
1. All AC timing is shown with respect to V_{IH}/V_{IL} levels unless otherwise noted.

Table A-23 Low Voltage ECLK Bus Timing

(V_{DD} and V_{DDSYN} = 2.7 to 3.6 Vdc, V_{SS} = 0 Vdc, $T_A = T_L$ to T_H)[1]

Num	Characteristic	Symbol	Min	Max	Unit
E1	ECLK Low to Address Valid[2]	t_{EAD}	—	60	ns
E2	ECLK Low to Address Hold	t_{EAH}	10	—	ns
E3	ECLK Low to $\overline{CS}$ Valid ($\overline{CS}$ Delay)	t_{ECSD}	—	150	ns
E4	ECLK Low to $\overline{CS}$ Hold	t_{ECSH}	15	—	ns
E5	$\overline{CS}$ Negated Width	t_{ECSN}	30	—	ns
E6	Read Data Setup Time	t_{EDSR}	30	—	ns
E7	Read Data Hold Time	t_{EDHR}	15	—	ns
E8	ECLK Low to Data High Impedance	t_{EDHZ}	—	60	ns
E9	$\overline{CS}$ Negated to Data Hold (Read)	t_{ECDH}	0	—	ns
E10	$\overline{CS}$ Negated to Data High Impedance	t_{ECDZ}	—	1	t_{cyc}
E11	ECLK Low to Data Valid (Write)	t_{EDDW}	—	2	t_{cyc}
E12	ECLK Low to Data Hold (Write)	t_{EDHW}	5	—	ns
E13	$\overline{CS}$ Negated to Data Hold (Write)	t_{ECHW}	0	—	ns
E14	Address Access Time (Read)[3]	t_{EACC}	386	—	ns
E15	Chip-Select Access Time (Read)[4]	t_{EACS}	296	—	ns
E16	Address Setup Time	t_{EAS}	—	1/2	t_{cyc}

NOTES:
1. Refer to notes in **Table A-26**.

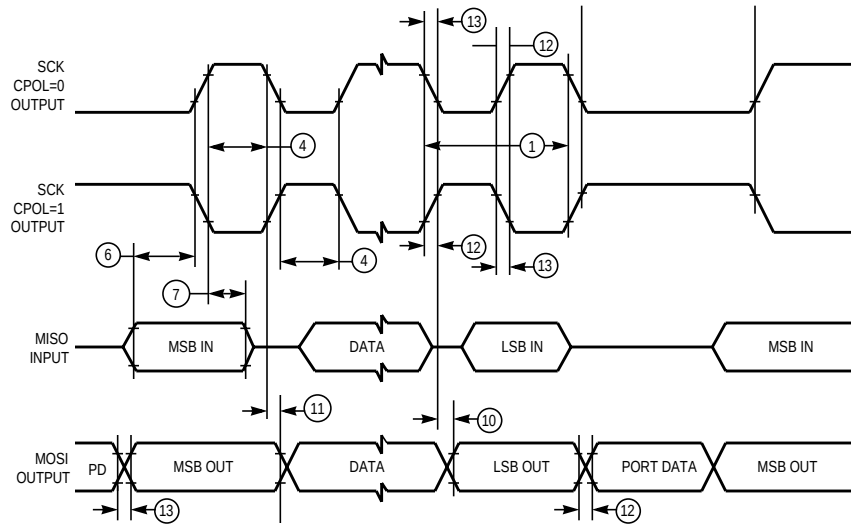


Figure A-20 SPI Timing — Master, CPHA = 0

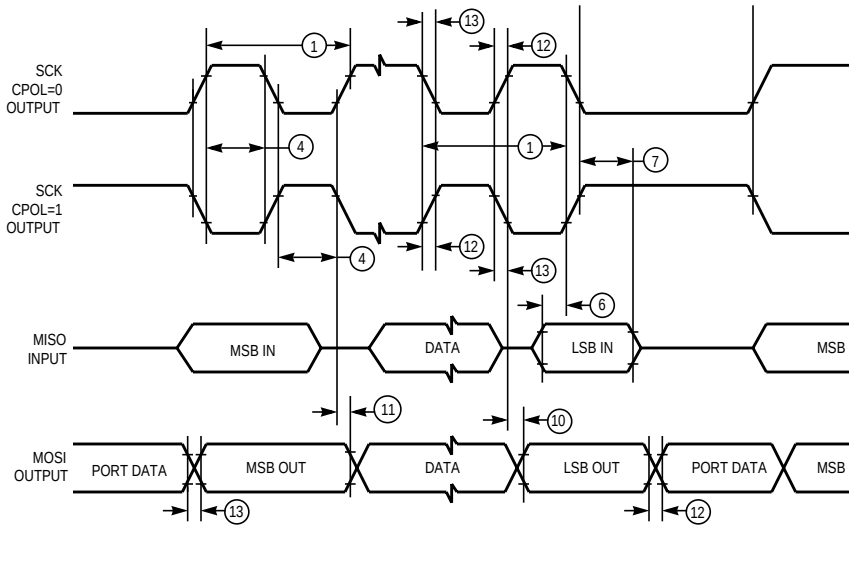


Figure A-21 SPI Timing — Master, CPHA = 1

Table A-37 Low Voltage ADC Conversion Characteristics (Operating)

(V_{DD} and V_{DDA} = 2.7 to 3.6 Vdc , V_{SS} = 0 Vdc, $T_A = T_L$ to T_H, f_{ADCLK} = 1.05 MHz)

Num	Parameter	Symbol	Min	Typical	Max	Unit
1	8-Bit Resolution[1]	1 Count	—	12	—	mV
2	8-Bit Differential Nonlinearity	DNL	–0.5	—	0.5	Counts
3	8-Bit Integral Nonlinearity	INL	–1.0	—	1.0	Counts
4	8-Bit Absolute Error[2]	AE	–1.5	—	1.5	Counts
5	10-Bit Resolution[1]	1 Count	—	3	—	mV
6	10-Bit Differential Nonlinearity[3]	DNL	–1	—	1	Counts
7	10-Bit Integral Nonlinearity[3]	INL	–2.0	—	2.0	Counts
8	10-Bit Absolute Error[3,4]	AE	–4	—	4.0	Counts
9	Source Impedance at Input[5]	R_S	—	20	—	kΩ

NOTES:

1. At $V_{RH} - V_{RL}$= 3.072 V, one 10-bit count = 3 mV and one 8-bit count = 12 mV.
2. 8-bit absolute error of 1.5 counts (18 mV) includes 1/2 count (6 mV) inherent quantization error and 1 count (12 mV) circuit (differential, integral, and offset) error.
3. Conversion accuracy varies with f_{ADCLK} rate. Reduced conversion accuracy occurs at maximum f_{ADCLK}. Assumes that minimum sample time (2 ADC clocks) is selected.
4. 10-bit absolute error of 4.0 counts (12 mV) includes 1/2 count (1.5 mV) inherent quantization error and 3.5 counts (10.5 mV) circuit (differential, integral, and offset) error.
5. Maximum source impedance is application-dependent. Error resulting from pin leakage depends on junction leakage into the pin and on leakage due to charge-sharing with internal capacitance.
   Error from junction leakage is a function of external source impedance and input leakage current. Expected error in result value due to junction leakage is expressed in voltage (V_{ERRJ}):

$$V_{ERRJ} = R_S \times I_{OFF}$$

   where I_{OFF} is a function of operating temperature, as shown in **Table A-35**.
   Charge-sharing leakage is a function of input source impedance, conversion rate, change in voltage between successive conversions, and the size of the decoupling capacitor used. Error levels are best determined empirically. In general, continuous conversion of the same channel may not be compatible with high source impedance.

Table B-1 M68HC16 Z-Series Ordering Information (Continued)

(Shaded cells indicate preliminary part numbers)

Device	Crystal Input	Operating Voltage	Package Type	Temperature	Frequency (MHz)	Package Order Quantity	Order Number
MC68HC16Z2 (No ROM)	4 MHz	5 V	144-Pin TQFP	–40 to +85°C	16 MHz	2	SPMCM16Z2BCPV16
						60	MCM16Z2BCPV16
						300	MCM16Z2BCPV16B1
					20 MHz	2	SPMCM16Z2BCPV20
						60	MCM16Z2BCPV20
						300	MCM16Z2BCPV20B1
					25 MHz	2	SPMCM16Z2BCPV25
						60	MCM16Z2BCPV25
						300	MCM16Z2BCPV25B1
				–40 to +105°C	16 MHz	2	SPMCM16Z2BVPV16
						60	MCM16Z2BVPV16
						300	MCM16Z2BVPV16B1
					20 MHz	2	SPMCM16Z2BVPV20
						60	MCM16Z2BVPV20
						300	MCM16Z2BVPV20B1
					25 MHz	2	SPMCM16Z2BVPV25
						60	MCM16Z2BVPV25
						300	MCM16Z2BVPV25B1
				–40 to +125°C	16 MHz	2	SPMCM16Z2BMPV16
						60	MCM16Z2BMPV16
						300	MCM16Z2BMPV16B1
					20 MHz	2	SPMCM16Z2BMPV20
						60	MCM16Z2BMPV20
						300	MCM16Z2BMPV20B1
MC68HC16Z3 (ROM)	4 MHz or 32 kHz	5 V	132-Pin PQFP	–40 to +85°C	16 MHz	2	NA
						36	MC68HC16Z3CFC16
						180	NA
					20 MHz	2	NA
						36	MC68HC16Z3CFC20
						180	NA
					25 MHz	2	NA
						36	MC68HC16Z3CFC25
						180	NA

D.1.1 Condition Code Register

CCR — Condition Code Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S	MV	H	EV	N	Z	V	C	IP[2:0]			SM	PK[3:0]			

The CCR contains processor status flags, the interrupt priority field, and the program counter address extension field. The CPU16 has a special set of instructions that manipulate the CCR.

S — STOP Enable
0 = Stop CPU16 clocks when LPSTOP instruction is executed.
1 = Perform NOPs when LPSTOP instruction is executed.

MV — Accumulator M overflow flag
Set when overflow into AM35 has occurred.

H — Half Carry Flag
Set when a carry from A3 or B3 occurs during BCD addition.

EV — Accumulator M Extension Overflow Flag
EV is set when an overflow into AM31 has occurred.

N — Negative Flag
N is set under the following conditions:
- When the MSB is set in the operand of a read operation.
- When the MSB is set in the result of a logic or arithmetic operation.

Z — Zero Flag
Z is set under the following conditions:
- When all bits are zero in the operand of a read operation.
- When all bits are zero in the result of a logic or arithmetic operation.

V — Overflow Flag
Set when two's complement overflow occurs as the result of an operation.

C — Carry Flag
Set when carry or borrow occurs during arithmetic operation. Also used during shifts and rotates.

IP[2:0] — Interrupt Priority Field
The priority value in this field (0 to 7) is used to mask low priority interrupts.

SM — Saturate Mode Bit
When SM is set, if either EV or MV is set, data read from AM using TMER or TMET is given maximum positive or negative value, depending on the state of the AM sign bit before overflow.

PK[3:0] — Program Counter Address Extension Field
This field is concatenated with the program counter to form a 20-bit address.

D.5.9 Left Justified, Unsigned Result Register

LJURR — Left Justified, Unsigned Result Register **$YFF730–$YFF73F**

15	14	13	12	11	10	9	8	7	6	5 … 0
8/10	8/10	8/10	8/10	8/10	8/10	8/10	8/10	10	10	NOT USED

Conversion result is unsigned left-justified data. Bits [15:6] are used for 10-bit resolution. For 8-bit conversions, bits [15:8] contain data and bits [7:6] are zero. Bits [5:0] always return zero when read.

HMIE — HALTA and MODF Interrupt Enable
0 = HALTA and MODF interrupts disabled.
1 = HALTA and MODF interrupts enabled.
HMIE enables interrupt requests generated by the HALTA status flag or the MODF status flag in SPSR.

HALT — Halt QSPI
0 = QSPI operates normally.
1 = QSPI is halted for subsequent restart.
When HALT is set, the QSPI stops on a queue boundary. It remains in a defined state from which it can later be restarted.

SPIF — QSPI Finished Flag
0 = QSPI is not finished.
1 = QSPI is finished.
SPIF is set after execution of the command at the address in ENDQP[3:0].

MODF — Mode Fault Flag
0 = Normal operation.
1 = Another SPI node requested to become the network SPI master while the QSPI was enabled in master mode.
The QSPI asserts MODF when the QSPI is in master mode (MSTR = 1) and the $\overline{SS}$ input pin is negated by an external driver.

HALTA — Halt Acknowledge Flag
0 = QSPI is not halted.
1 = QSPI is halted.
HALTA is set when the QSPI halts in response to setting the SPCR3 HALT bit.

Bit 4 — Not Implemented

CPTQP[3:0] — Completed Queue Pointer
CPTQP[3:0] points to the last command executed. It is updated when the current command is complete. When the first command in a queue is executing, CPTQP[3:0] contains either the reset value $0 or a pointer to the last command completed in the previous queue.

D.6.14 Receive Data RAM

RR[0:F] — Receive Data RAM **$YFFD00 – $YFFD1F**

Data received by the QSPI is stored in this segment. The CPU16 reads this segment to retrieve data from the QSPI. Data stored in receive RAM is right-justified. Unused bits in a receive queue entry are set to zero by the QSPI upon completion of the individual queue entry. Receive RAM data can be accessed using byte, word, or long-word addressing.