Freescale Semiconductor
Addendum

Document Number: QFN_Addendum
Rev. 0, 07/2014

Addendum for New QFN Package Migration

This addendum provides the changes to the 98A case outline numbers for products covered in this book. Case outlines were changed because of the migration from gold wire to copper wire in some packages. See the table below for the old (gold wire) package versus the new (copper wire) package.

To view the new drawing, go to Freescale.com and search on the new 98A package number for your device.

For more information about QFN package use, see EB806: *Electrical Connection Recommendations for the Exposed Pad on QFN and DFN Packages*.

© Freescale Semiconductor, Inc., 2014. All rights reserved.

Part Number	Package Description	Original (gold wire) package document number	Current (copper wire) package document number
MC68HC908JW32	48 QFN	98ARH99048A	98ASA00466D
MC9S08AC16			
MC9S908AC60			
MC9S08AC128			
MC9S08AW60			
MC9S08GB60A			
MC9S08GT16A			
MC9S08JM16			
MC9S08JM60			
MC9S08LL16			
MC9S08QE128			
MC9S08QE32			
MC9S08RG60			
MCF51CN128			
MC9RS08LA8	48 QFN	98ARL10606D	98ASA00466D
MC9S08GT16A	32 QFN	98ARH99035A	98ASA00473D
MC9S908QE32	32 QFN	98ARE10566D	98ASA00473D
MC9S908QE8	32 QFN	98ASA00071D	98ASA00736D
MC9S08JS16	24 QFN	98ARL10608D	98ASA00734D
MC9S08QB8			
MC9S08QG8	24 QFN	98ARL10605D	98ASA00474D
MC9S08SH8	24 QFN	98ARE10714D	98ASA00474D
MC9RS08KB12	24 QFN	98ASA00087D	98ASA00602D
MC9S08QG8	16 QFN	98ARE10614D	98ASA00671D
MC9RS08KB12	8 DFN	98ARL10557D	98ASA00672D
MC9S08QG8			
MC9RS08KA2	6 DFN	98ARL10602D	98ASA00735D

MC9RS08KA2
MC9RS08KA1

Data Sheet

RS08
Microcontrollers

MC9RS08KA2
Rev. 4
12/2008

freescale.com

MC9RS08KA2 Features

8-Bit RS08 Central Processor Unit (CPU)

- Simplified S08 instruction set with added high-performance instructions
  - LDA, STA, and CLR instructions support the short addressing mode; address $0000 to $001F can be accessed via a single-byte instruction
  - ADD, SUB, INC, and DEC instructions support the tiny addressing mode; address $0000 to $000F can be accessed via a single-byte instruction with reduced instruction cycle
  - Shadow PC register instructions: SHA and SLA
- Pending interrupt indication
- Index addressing via D[X] and X register
- Direct page access to the entire memory map through paging window

Memory

- On-chip Flash EEPROM
  - MC9RS08KA2: 2048 bytes
  - MC9RS08KA1: 1024 bytes
- 63 bytes on-chip RAM

Power-Saving Modes

- Wait and stop
- Wakeup from power-saving modes using real-time interrupt (RTI), KBI, or ACMP

Clock Source

- **ICS** — Trimmable 20-MHz internal clock source
  - Up to 10-MHz internal bus operation
  - 0.2% trimmable resolution, 2% deviation over temperature and voltage range

System Protection

- Computer operating properly (COP) reset running off bus-independent clock source
- Low-voltage detection with reset or stop wakeup

Peripherals

- **MTIM** — 8-bit modulo timer
- **ACMP** — Analog comparator
  - Full rail-to-rail supply operation
  - Option to compare to fixed internal bandgap reference voltage
  - Can operate in stop mode
- **KBI** — Keyboard interrupt ports
  - Three KBI ports in 6-pin package
  - Five KBI ports in 8-pin package

Development Support

- Background debug system
- Breakpoint capability to allow single breakpoint setting during in-circuit debug

Package Options

- 6-pin dual flat no lead (DFN) package
  - Two general-purpose input/output (I/O) pins
  - One general-purpose input pin
  - One general-purpose output pin
- 8-pin plastic dual in-line pin (PDIP) package
  - Four general-purpose input/output (I/O) pins
  - One general-purpose input pin
  - One general-purpose output pin
- 8-pin narrow body SOIC package
  - Four general-purpose input/output (I/O) pins
  - One general-purpose input pin
  - One general-purpose output pin

MC9RS08KA2 Series Data Sheet

Covers: MC9RS08KA2
MC9RS08KA1

MC9RS08KA2
Rev. 4
12/2008

freescale™
semiconductor

Revision History

To provide the most up-to-date information, the revision of our documents on the World Wide Web will be the most current. Your printed copy may be an earlier revision. To verify you have the latest information available, refer to:

http://freescale.com

The following revision history table summarizes changes contained in this document.

Revision Number	Revision Date	Description of Changes
1.0	04/2006	Initial public release version
2	12/2006	Added MC9RS08KA1
3	09/2007	Corrected Instruction Set Summary LDX ,X row operand to read 0E 0F. Revised the Analog Comparator Electrical Specifications including the ACMP Bandgap reference voltage values. Corrected a transposition in the DFN drawing no. Updated the ICS Characteristic table in the Electricals Appendix to include the ICS factory trim and reference the parameters t_{ir_wu} and t_{fll_wu} that are discussed in the ICS chapter.
4	12/2008	Revised Figure 1-2. Updated “How to Reach Us” information. Changed the mechanical drawing of 6-pin DFN in the Appendix B, “Ordering Information and Mechanical Drawings.”

This product incorporates SuperFlash® technology licensed from SST.

Freescale, and the Freescale logo are trademarks of Freescale Semiconductor, Inc.
© Freescale Semiconductor, Inc., 2006-2008. All rights reserved.

List of Chapters

Chapter	Title	Page
Chapter 1	MC9RS08KA2 Series Device Overview	15
Chapter 2	Pins and Connections	17
Chapter 3	Modes of Operation	21
Chapter 4	Memory	25
Chapter 5	Resets, Interrupts, and General System Control	35
Chapter 6	Parallel Input/Output Control	45
Chapter 7	Keyboard Interrupt (RS08KBIV1)	51
Chapter 8	Central Processor Unit (RS08CPUV1)	57
Chapter 9	Internal Clock Source (RS08ICSV1)	75
Chapter 10	Analog Comparator (RS08ACMPV1)	83
Chapter 11	Modulo Timer (RS08MTIMV1)	89
Chapter 12	Development Support	97
Appendix A	Electrical Characteristics	109
Appendix B	Ordering Information and Mechanical Drawings	123

Table of Contents

Section Number **Title** **Page**

Chapter 1
MC9RS08KA2 Series Device Overview

1.1 Overview 15
1.2 MCU Block Diagram 15
1.3 System Clock Distribution 16

Chapter 2
Pins and Connections

2.1 Introduction 17
2.2 Device Pin Assignment 17
2.3 Recommended System Connections 18
2.4 Pin Detail 18
2.4.1 Power 19
2.4.2 PTA2/KBIP2/TCLK/RESET/V_{PP} 19
2.4.3 PTA3/ACMPO/BKGD/MS 19
2.4.4 General-Purpose I/O and Peripheral Ports 20

Chapter 3
Modes of Operation

3.1 Introduction 21
3.2 Features 21
3.3 Run Mode 21
3.4 Active Background Mode 21
3.5 Wait Mode 22
3.6 Stop Mode 23
3.6.1 Active BDM Enabled in Stop Mode 24
3.6.2 LVD Enabled in Stop Mode 24

Chapter 4
Memory

4.1 Memory Map 25
4.2 Unimplemented Memory 27
4.3 Indexed/Indirect Addressing 27
4.4 RAM and Register Addresses and Bit Assignments 27
4.5 RAM 29
4.6 Flash 29
4.6.1 Features 29
4.6.2 Flash Programming Procedure 30
4.6.3 Flash Mass Erase Operation 30

4.6.4 Security 31
4.7 Flash Registers and Control Bits 32
4.7.1 Flash Options Register (FOPT and NVOPT) 32
4.7.2 Flash Control Register (FLCR) 33
4.8 Page Select Register (PAGESEL) 33

Chapter 5
Resets, Interrupts, and General System Control

5.1 Introduction 35
5.2 Features 35
5.3 MCU Reset 35
5.4 Computer Operating Properly (COP) Watchdog 36
5.5 Interrupts 36
5.6 Low-Voltage Detect (LVD) System 37
5.6.1 Power-On Reset Operation 37
5.6.2 LVD Reset Operation 37
5.6.3 LVD Interrupt Operation 37
5.7 Real-Time Interrupt (RTI) 37
5.8 Reset, Interrupt, and System Control Registers and Control Bits 38
5.8.1 System Reset Status Register (SRS) 38
5.8.2 System Options Register (SOPT) 39
5.8.3 System Device Identification Register (SDIDH, SDIDL) 40
5.8.4 System Real-Time Interrupt Status and Control Register (SRTISC) 41
5.8.5 System Power Management Status and Control 1 Register (SPMSC1) 43
5.8.6 System Interrupt Pending Register (SIP1) 44

Chapter 6
Parallel Input/Output Control

6.1 Pin Behavior in Low-Power Modes 46
6.2 Parallel I/O Registers 46
6.2.1 Port A Registers 46
6.3 Pin Control Registers 47
6.3.1 Port A Pin Control Registers 47
6.3.1.1 Internal Pulling Device Enable 47
6.3.1.2 Pullup/Pulldown Control 48
6.3.1.3 Output Slew Rate Control Enable 48

Chapter 7
Keyboard Interrupt (RS08KBIV1)

7.1 Introduction 51
7.1.1 Features 51
7.1.2 Modes of Operation 52
7.1.2.1 Operation in Wait Mode 52
7.1.2.2 Operation in Stop Mode 52
7.1.2.3 Operation in Active Background Mode 52

7.1.3 Block Diagram 52
7.2 External Signal Description 52
7.3 Register Definition 53
7.3.1 KBI Status and Control Register (KBISC) 53
7.3.2 KBI Pin Enable Register (KBIPE) 54
7.3.3 KBI Edge Select Register (KBIES) 54
7.4 Functional Description 55
7.4.1 Edge Only Sensitivity 55
7.4.2 Edge and Level Sensitivity 55
7.4.3 KBI Pullup/Pulldown Device 55
7.4.4 KBI Initialization 55

Chapter 8
Central Processor Unit (RS08CPUV1)

8.1 Introduction 57
8.2 Programmer's Model and CPU Registers 57
8.2.1 Accumulator (A) 58
8.2.2 Program Counter (PC) 59
8.2.3 Shadow Program Counter (SPC) 59
8.2.4 Condition Code Register (CCR) 59
8.2.5 Indexed Data Register (D[X]) 60
8.2.6 Index Register (X) 60
8.2.7 Page Select Register (PAGESEL) 61
8.3 Addressing Modes 61
8.3.1 Inherent Addressing Mode (INH) 61
8.3.2 Relative Addressing Mode (REL) 61
8.3.3 Immediate Addressing Mode (IMM) 62
8.3.4 Tiny Addressing Mode (TNY) 62
8.3.5 Short Addressing Mode (SRT) 63
8.3.6 Direct Addressing Mode (DIR) 63
8.3.7 Extended Addressing Mode (EXT) 63
8.3.8 Indexed Addressing Mode (IX, Implemented by Pseudo Instructions) 63
8.4 Special Operations 63
8.4.1 Reset Sequence 64
8.4.2 Interrupts 64
8.4.3 Wait and Stop Mode 64
8.4.4 Active Background Mode 64
8.5 Summary Instruction Table 65

Chapter 9
Internal Clock Source (RS08ICSV1)

9.1 Introduction 75
9.1.1 Features 76
9.1.2 Modes of Operation 76
9.1.2.1 FLL Engaged Internal (FEI) 76
9.1.2.2 FLL Bypassed Internal (FBI) 76
9.1.2.3 FLL Bypassed Internal Low Power (FBILP) 76
9.1.2.4 Stop (STOP) 76
9.1.3 Block Diagram 76
9.2 External Signal Description 77
9.3 Register Definition 77
9.3.1 ICS Control Register 1 (ICSC1) 77
9.3.2 ICS Control Register 2 (ICSC2) 78
9.3.3 ICS Trim Register (ICSTRM) 79
9.3.4 ICS Status and Control (ICSSC) 79
9.4 Functional Description 80
9.4.1 Operational Modes 80
9.4.1.1 FLL Engaged Internal (FEI) 80
9.4.1.2 FLL Bypassed Internal (FBI) 80
9.4.1.3 FLL Bypassed Internal Low Power (FBILP) 80
9.4.1.4 Stop 81
9.4.2 Mode Switching 81
9.4.3 Bus Frequency Divider 81
9.4.4 Low Power Bit Usage 81
9.4.5 Internal Reference Clock 81
9.4.6 Fixed Frequency Clock 82

Chapter 10
Analog Comparator (RS08ACMPV1)

10.1 Introduction 83
10.1.1 Features 84
10.1.2 Modes of Operation 84
10.1.2.1 Operation in Wait Mode 84
10.1.2.2 Operation in Stop Mode 84
10.1.2.3 Operation in Active Background Mode 84
10.1.3 Block Diagram 84
10.2 External Signal Description 86
10.3 Register Definition 86
10.3.1 ACMP Status and Control Register (ACMPSC) 86
10.4 Functional Description 87

Chapter 11
Modulo Timer (RS08MTIMV1)

11.1 Introduction 89
11.1.1 Features 90
11.1.2 Modes of Operation 90
11.1.2.1 Operation in Wait Mode 90
11.1.2.2 Operation in Stop Modes 90
11.1.2.3 Operation in Active Background Mode 90
11.1.3 Block Diagram 91
11.2 External Signal Description 91
11.3 Register Definition 91
11.3.1 MTIM Status and Control Register (MTIMSC) 92
11.3.2 MTIM Clock Configuration Register (MTIMCLK) 93
11.3.3 MTIM Counter Register (MTIMCNT) 93
11.3.4 MTIM Modulo Register (MTIMMOD) 94
11.4 Functional Description 95
11.4.1 MTIM Operation Example 96

Chapter 12
Development Support

12.1 Introduction 97
12.2 Features 97
12.3 RS08 Background Debug Controller (BDC) 98
12.3.1 BKGD Pin Description 99
12.3.2 Communication Details 99
12.3.3 SYNC and Serial Communication Timeout 102
12.4 BDC Registers and Control Bits 103
12.4.1 BDC Status and Control Register (BDCSCR) 103
12.4.2 BDC Breakpoint Match Register 104
12.5 RS08 BDC Commands 105

Section Number	Title	Page

Appendix A
Electrical Characteristics

A.1 Introduction 109
A.2 Absolute Maximum Ratings 109
A.3 Thermal Characteristics 110
A.4 Electrostatic Discharge (ESD) Protection Characteristics 111
A.5 DC Characteristics 111
A.6 Supply Current Characteristics 115
A.7 Analog Comparator (ACMP) Electricals 117
A.8 Internal Clock Source Characteristics 117
A.9 AC Characteristics 118
A.9.1 Control Timing 118
A.10 FLASH Specifications 119

Appendix B
Ordering Information and Mechanical Drawings

B.1 Ordering Information 123
B.2 Mechanical Drawings 123

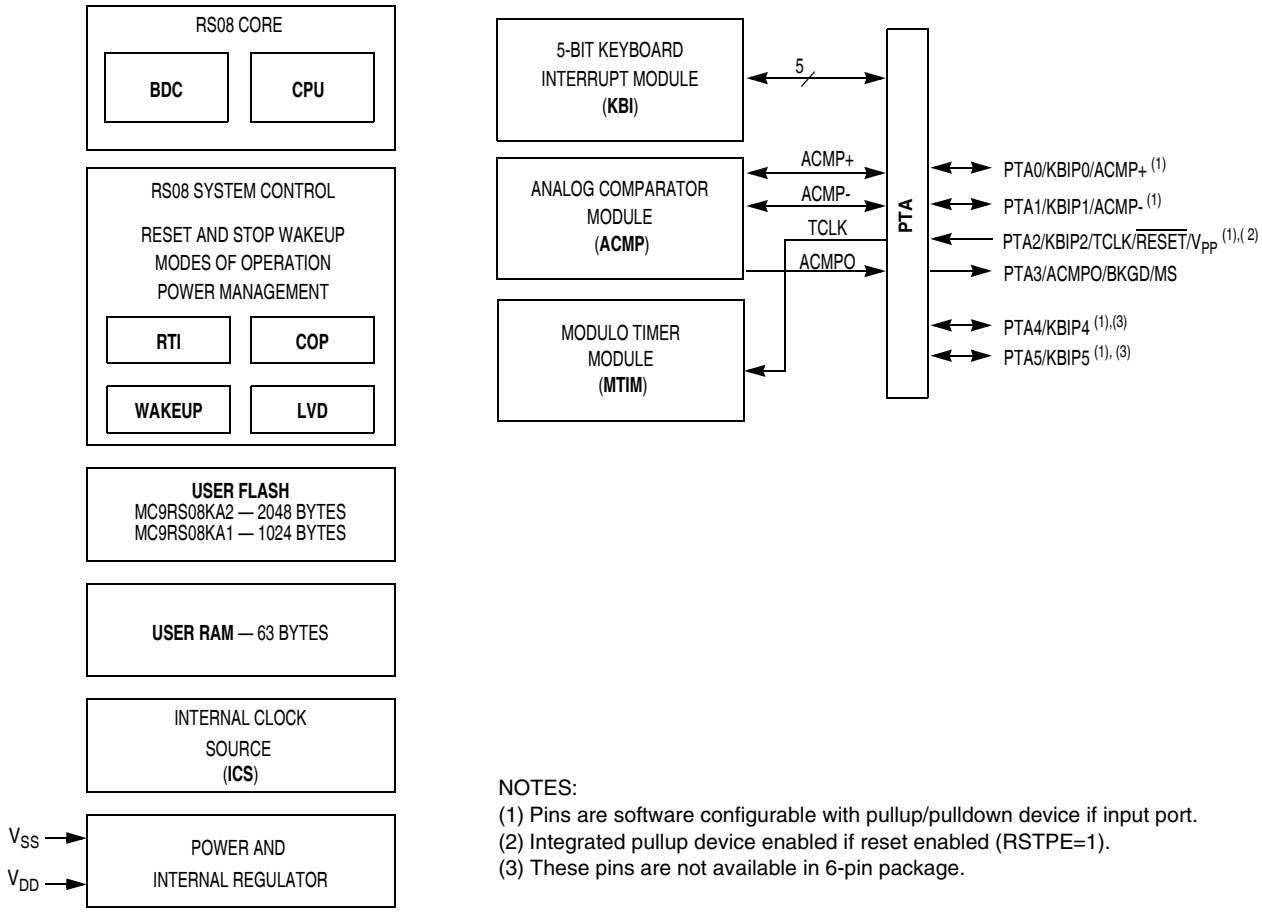
Chapter 1
MC9RS08KA2 Series Device Overview

1.1 Overview

The MC9RS08KA2 Series microcontroller unit (MCU) is an extremely low-cost, small pin count device for home appliances, toys, and small geometry applications. This device is composed of standard on-chip modules including, a very small and highly efficient RS08 CPU core, 63 bytes RAM, 2K bytes Flash, an 8-bit modulo timer, keyboard interrupt, and analog comparator. The device is available in small 6- and 8-pin packages.

1.2 MCU Block Diagram

The block diagram, Figure 1-1, shows the structure of the MC9RS08KA2 Series MCU.



NOTES:
(1) Pins are software configurable with pullup/pulldown device if input port.
(2) Integrated pullup device enabled if reset enabled (RSTPE=1).
(3) These pins are not available in 6-pin package.

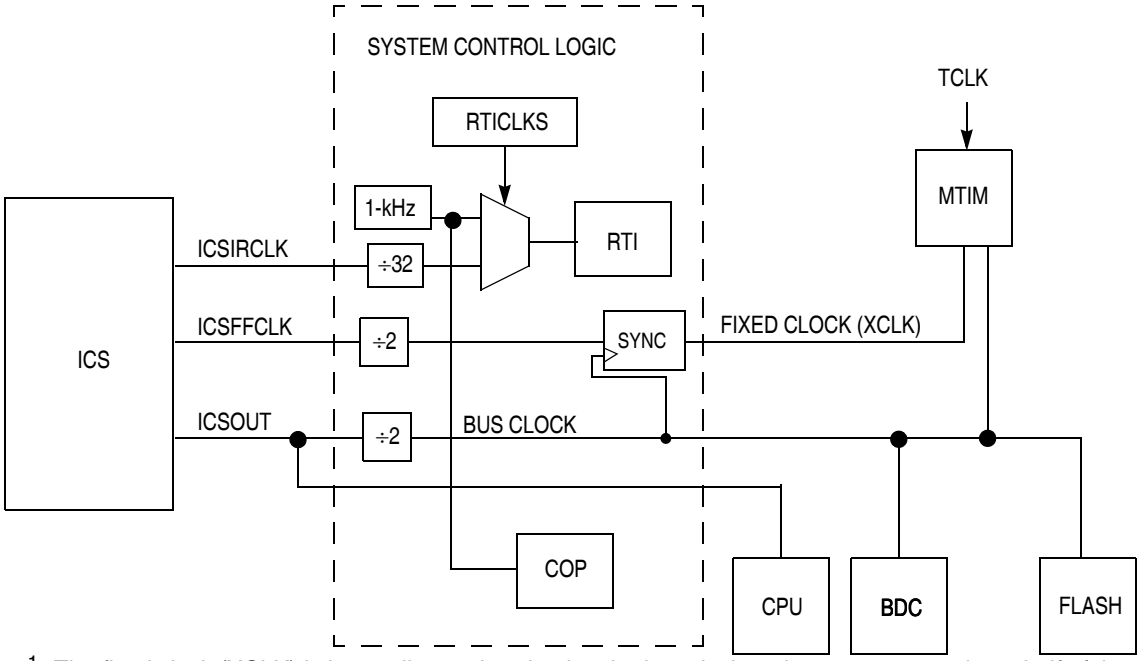
Figure 1-1. MC9RS08KA2 Series Block Diagram

Table 1-1 provides the functional versions of the on-chip modules.

Table 1-1. Block Versions

Module	Version
Analog Comparator (ACMP)	1
Keyboard Interrupt (KBI)	1
Modulo Timer (MTIM)	1
Internal Clock Source (ICS)	1

1.3 System Clock Distribution



1 The fixed clock (XCLK) is internally synchronized to the bus clock and must not exceed one half of the bus clock frequency

Figure 1-2. System Clock Distribution Diagram

Figure 1-2 shows a simplified clock connection diagram for the MCU. The bus clock frequency is half of the ICS output frequency and is used by all of the internal modules.

Chapter 2
Pins and Connections

2.1 Introduction

This chapter describes signals that connect to package pins. It includes a pinout diagram, a table of signal properties, and a detailed discussion of signals.

2.2 Device Pin Assignment

Figure 2-1 and Figure 2-3 show the pin assignments in the packages available for the MC9RS08KA2 Series.

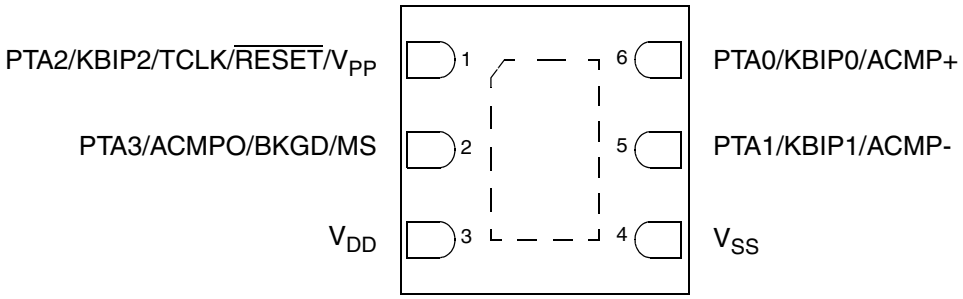


Figure 2-1. MC9RS08KA2 Series in 6-Pin DFN

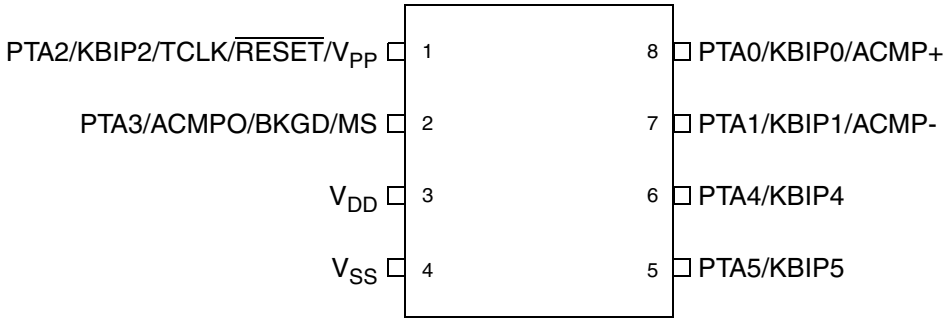


Figure 2-2. MC9RS08KA2 Series in 8-Pin PDIP

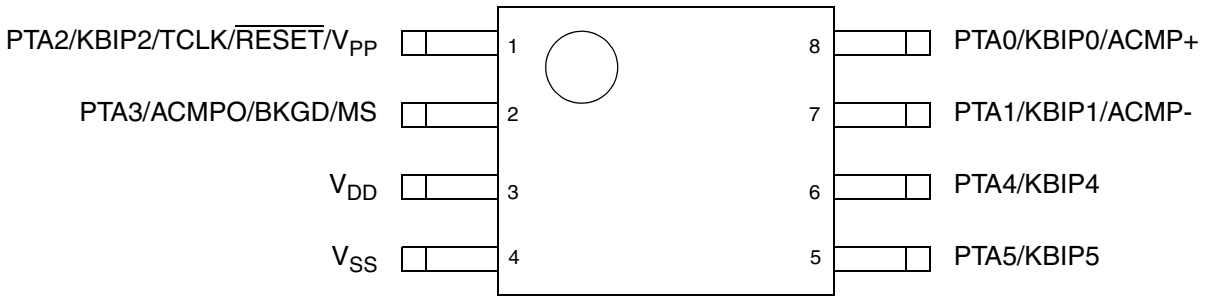


Figure 2-3. MC9RS08KA2 Series in 8-Pin Narrow Body SOIC

2.3 Recommended System Connections

Figure 2-4 shows reference connection for background debug and Flash programming.

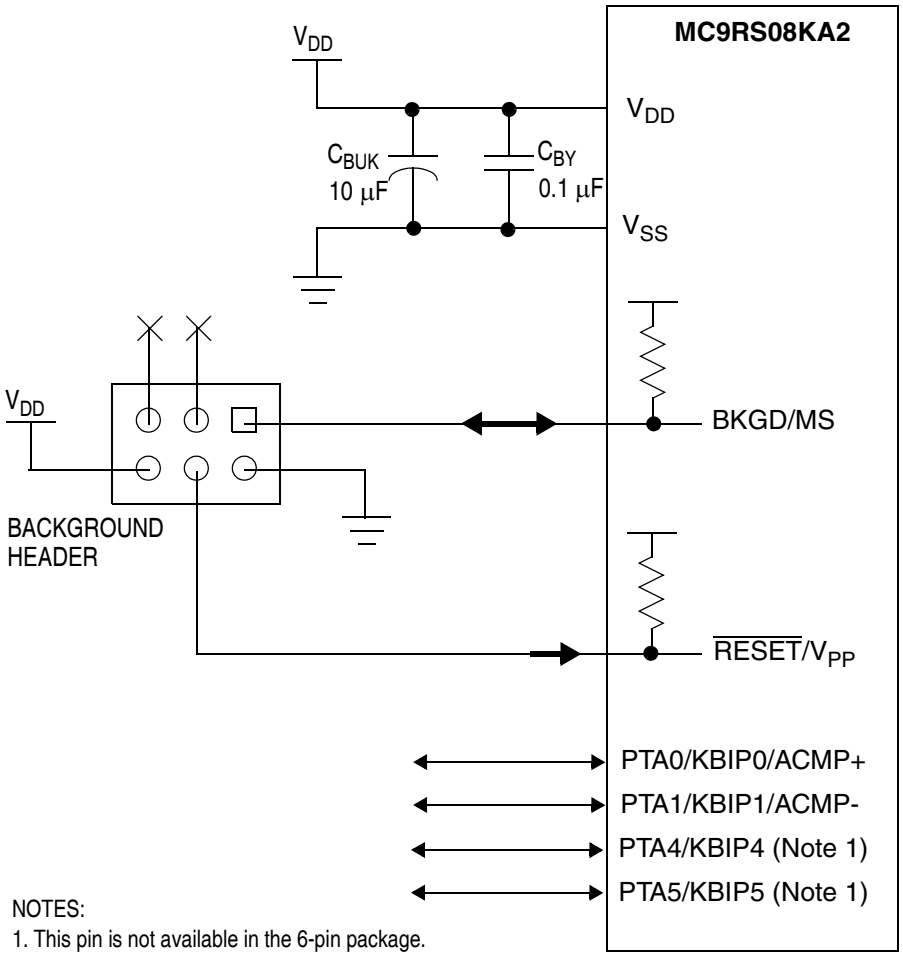


Figure 2-4. Reference System Connection Diagram

2.4 Pin Detail

This section provides a detailed description of system connections.

2.4.1 Power

V_{DD} and V_{SS} are the primary power supply pins for the MCU. This voltage source supplies power to all I/O buffer circuitry and to an internal voltage regulator. The internal voltage regulator provides a regulated lower-voltage source to the CPU and other internal circuitry of the MCU.

Typically, application systems have two separate capacitors across the power pins: a bulk electrolytic capacitor, such as a 10-µF tantalum capacitor, to provide bulk charge storage for the overall system, and a bypass capacitor, such as a 0.1-µF ceramic capacitor, located as near to the MCU power pins as practical to suppress high-frequency noise.

2.4.2 PTA2/KBIP2/TCLK/$\overline{\text{RESET}}$/V_{PP}

After a power-on reset (POR) into user mode, the PTA2/KBIP2/TCLK/$\overline{\text{RESET}}$/V_{PP} pin defaults to a general-purpose input port pin, PTA2. Setting RSTPE in SOPT configures the pin to be the $\overline{\text{RESET}}$ input pin. After configured as $\overline{\text{RESET}}$, the pin will remain as $\overline{\text{RESET}}$ until the next POR. The $\overline{\text{RESET}}$ pin can be used to reset the MCU from an external source when the pin is driven low. When enabled as the $\overline{\text{RESET}}$ pin (RSTPE = 1), the internal pullup device is automatically enabled.

External V_{PP} voltage (typically 12 V, see Section A.10, “FLASH Specifications”) is required on this pin when performing Flash programming or erasing. The V_{PP} connection is always connected to the internal Flash module regardless of the pin function. To avoid over stressing the Flash, external V_{PP} voltage must be removed and voltage higher than V_{DD} must be avoided when Flash programming or erasing is not taking place.

NOTE

This pin does not contain a clamp diode to V_{DD} and should not be driven above V_{DD} when Flash programming or erasing is not taking place.

2.4.3 PTA3/ACMPO/BKGD/MS

The background / mode select function is shared with an output-only pin on PTA3 pin and the optional analog comparator output. While in reset, the pin functions as a mode select pin. Immediately after reset rises, the pin functions as the background pin and can be used for background debug communication. While functioning as a background / mode select pin, this pin has an internal pullup device enabled. To use as an output-only port, BKGDPE in SOPT must be cleared.

If nothing is connected to this pin, the MCU will enter normal operating mode at the rising edge of reset. If a debug system is connected to the 6-pin standard background debug header, it can hold BKGD/MS low during the power-on-reset, which forces the MCU to active background mode.

The BKGD pin is used primarily for background debug controller (BDC) communications using a custom protocol that uses 16 clock cycles of the target MCU’s BDC clock per bit time. The target MCU’s BDC clock equals the bus clock rate; therefore, no significant capacitance should connected to the BKGD/MS pin that could interfere with background serial communications.

Although the BKGD pin is a pseudo open-drain pin, the background debug communication protocol provides brief, actively driven, high speedup pulses to ensure fast rise times. Small capacitances from

cables and the absolute value of the internal pullup device play almost no role in determining rise and fall times on the BKGD pin.

2.4.4 General-Purpose I/O and Peripheral Ports

The remaining pins are shared among general-purpose I/O and on-chip peripheral functions such as timers and analog comparator. Immediately after reset, all of these pins are configured as high-impedance general-purpose inputs with internal pullup/pulldown devices disabled.

NOTE

To avoid extra current drain from floating input pins, the reset initialization routine in the application program should either enable on-chip pullup/pulldown devices or change the direction of unused pins to outputs.

Table 2-1. Pin Sharing Reference

Pin Name	Direction	Pullup/Pulldown[1]	Alternative Functions[2]	
V_{DD}	—	—	Power	
V_{SS}	—	—	Ground	
PTA0	I/O	SWC	PTA0 KBIP0 ACMP+	General-purpose input/output (GPIO) Keyboard interrupt (stop/wait wakeup only) Analog comparator input
PTA1	I/O	SWC	PTA1 KBIP1 ACMP-	General-purpose input/output (GPIO) Keyboard interrupt (stop/wait wakeup only) Analog comparator input
PTA2	I	SWC[4]	PTA2 KBIP2 TCLK $\overline{\text{RESET}}$ V_{PP}	General-purpose input Keyboard interrupt (stop/wait wakeup only) Modulo timer clock source Reset V_{PP}
PTA3	I/O[3]	—[4]	PTA3 ACMPO BKGD MS	General-purpose output Analog comparator output Background debug data Mode select
PTA4[5]	I/O	SWC	PTA4 KBIP4	General-purpose input/output (GPIO) Keyboard interrupt (stop/wait wakeup only)
PTA5[5]	I/O	SWC	PTA5 KBIP5	General-purpose input/output (GPIO) Keyboard interrupt (stop/wait wakeup only)

1 SWC is software-controlled pullup/pulldown resistor; the register is associated with the respective port.

2 Alternative functions are listed lowest priority first. For example, GPIO is the lowest priority alternative function of the PTA0 pin; ACMP+ is the highest priority alternative function of the PTA0 pin.

3 Output-only when configured as PTA3 function.

4 When PTA2 or PTA3 is configured as $\overline{\text{RESET}}$ or BKGD/MS, respectively, pullup is enabled. When V_{PP} is attached, pullup/pulldown is disabled automatically.

5 This pin is not available in 6-pin package. Enabling either the pullup or pulldown device is recommended to prevent extra current leakage from the floating input pin.

Chapter 3
Modes of Operation

3.1 Introduction

This chapter describes the operating modes of the MC9RS08KA2 Series are described in this chapter. It also details entry into each mode, exit from each mode, and functionality while in each of the modes.

3.2 Features

- Active background mode for code development
- Wait mode:
  - CPU shuts down to conserve power
  - System clocks continue to run
  - Full voltage regulation is maintained
- Stop mode:
  - System clocks are stopped; voltage regulator in standby
  - All internal circuits remain powered for fast recovery

3.3 Run Mode

This is the normal operating mode for the MC9RS08KA2 Series. This mode is selected when the BKGD/MS pin is high at the rising edge of reset. In this mode, the CPU executes code from internal memory with execution beginning at the address $3FFD. A JMP instruction (opcode $BC) with operand located at $3FFE–$3FFF must be programmed for correct reset operation into the user application. The operand defines the location at which the user program will start. Instead of using the vector fetching process as in HC08/S08 families, the user program is responsible for performing a JMP instruction to relocate the program counter to the correct user program start location.

3.4 Active Background Mode

The active background mode functions are managed through the background debug controller (BDC) in the RS08 core. The BDC provides the means for analyzing MCU operation during software development.

Active background mode is entered in any of four ways:

- When the BKGD/MS pin is low during power-on-reset (POR) or immediately after issuing a background debug force reset (BDC_RESET) command
- When a BACKGROUND command is received through the BKGD pin
- When a BGND instruction is executed

- When a BDC breakpoint is encountered

After active background mode is entered, the CPU is held in a suspended state waiting for serial background commands rather than executing instructions from the user application program.

Background commands are of two types:

- Non-intrusive commands, defined as commands that can be issued while the user program is running, can be issued through the BKGD pin while the MCU is in run mode. Non-intrusive commands can also be executed when the MCU is in the active background mode. Non-intrusive commands include:
  - Memory access commands
  - Memory-access-with-status commands
  - BACKGROUND command
- Active background commands, which can be executed only while the MCU is in active background mode, include commands to:
  - Read or write CPU registers
  - Trace one user program instruction at a time
  - Leave active background mode to return to the user application program (GO)

Active background mode is used to program user application code into the Flash program memory before the MCU is operated in run mode for the first time. When the MC9RS08KA2 Series is shipped from the Freescale Semiconductor factory, the Flash program memory is usually erased so there is no program that could be executed in run mode until the Flash memory is initially programmed. The active background mode can also be used to erase and reprogram the Flash memory after it has been previously programmed.

For additional information about the active background mode, refer to the Development Support chapter of this data sheet.

3.5 Wait Mode

Wait mode is entered by executing a WAIT instruction. Upon execution of the WAIT instruction, the CPU enters a low-power state in which it is not clocked. The program counter (PC) is halted at the position where the WAIT instruction is executed. When an interrupt request occurs:

1. MCU exits wait mode and resumes processing.
2. PC is incremented by one and fetches the next instruction to be processed.

It is the responsibility of the user program to probe the corresponding interrupt source that woke the MCU, because no vector fetching process is involved.

While the MCU is in wait mode, not all background debug commands can be used. Only the BACKGROUND command and memory-access-with-status commands are available when the MCU is in wait mode. The memory-access-with-status commands do not allow memory access, but they report an error indicating that the MCU is in either stop or wait mode. The BACKGROUND command can be used to wake the MCU from wait mode and enter active background mode.

Table 3-1 summarizes the behavior of the MCU in wait mode.

Table 3-1. Wait Mode Behavior

Mode	CPU	Digital Peripherals	ICS	ACMP	Regulator	I/O Pins	RTI
Wait	Standby	Optionally on	On	Optionally on	On	States held	Optionally on

3.6 Stop Mode

Stop mode is entered upon execution of a STOP instruction when the STOPE bit in the system option register is set. In stop mode, all internal clocks to the CPU and the modules are halted. If the STOPE bit is not set when the CPU executes a STOP instruction, the MCU will not enter stop mode and an illegal opcode reset is forced.

Table 3-2 summarizes the behavior of the MCU in stop mode.

Table 3-2. Stop Mode Behavior

Mode	CPU	Digital Peripherals	ICS[1]	ACMP[2]	Regulator	I/O Pins	RTI[3]
Stop	Standby	Standby	Optionally on	Optionally on	Standby	States held	Optionally on

1 ICS requires IREFSTEN = 1 and LVDE and LVDSE must be set to allow operation in stop.

2 If bandgap reference is required, the LVDE and LVDSE bits in the SPMSC1 must both be set before entering stop.

3 If the 32-kHz trimmed clock in the ICS module is selected as the clock source for the RTI, LVDE and LVDSE bits in the SPMSC1 must both be set before entering stop.

Upon entering stop mode, all of the clocks in the MCU are halted. The ICS is turned off by default when the IREFSTEN bit is cleared and the voltage regulator is put in standby. The states of all of the internal registers and logic, as well as the RAM content, are maintained. The I/O pin states are held.

Exit from stop is done by asserting $\overline{\text{RESET}}$, any asynchronous interrupt that has been enabled, or the real-time interrupt. The asynchronous interrupts are the KBI pins, LVD interrupt, or the ACMP interrupt.

If stop is exited by asserting the $\overline{\text{RESET}}$ pin, the MCU will be reset and program execution starts at location $3FFD. If exited by means of an asynchronous interrupt or real-time interrupt, the next instruction after the location where the STOP instruction was executed will be executed accordingly. It is the responsibility of the user program to probe for the corresponding interrupt source that woke the CPU.

A separate self-clocked source (≈1 kHz) for the real-time interrupt allows a wakeup from stop mode with no external components. When RTIS = 000, the real-time interrupt function and the 1-kHz source are disabled. Power consumption is lower when the 1-kHz source is disabled, but in that case, the real-time interrupt cannot wake the MCU from stop.

The trimmed 32-kHz clock in the ICS module can also be enabled for the real-time interrupt to allow a wakeup from stop mode with no external components. The 32-kHz clock reference is enabled by setting

the IREFSTEN bit. For the ICS to run in stop, the LVDE and LVDSE bits in the SPMSC1 must both be set before entering stop.

3.6.1 Active BDM Enabled in Stop Mode

Entry into active background mode from run mode is enabled if the ENBDM bit in BDCSCR is set. This register is described in the Development Support chapter of this data sheet. If ENBDM is set when the CPU executes a STOP instruction, the system clocks to the background debug logic remain active when the MCU enters stop mode so background debug communication is still possible. In addition, the voltage regulator does not enter its low-power standby state; it maintains full internal regulation.

Most background commands are not available in stop mode. The memory-access-with-status commands do not allow memory access, but they report an error indicating that the MCU is in either stop or wait mode. The BACKGROUND command can be used to wake the MCU from stop and enter active background mode if the ENBDM bit is set. After active background mode is entered, all background commands are available.

Table 3-3 summarizes the behavior of the MCU in stop when entry into the active background mode is enabled.

Table 3-3. BDM Enabled Stop Mode Behavior

Mode	CPU	Digital Peripherals	ICS	ACMP	Regulator	I/O Pins	RTI
Stop	Standby	Standby	On	Optionally on	On	States held	Optionally on

3.6.2 LVD Enabled in Stop Mode

The LVD system is capable of generating either an interrupt or a reset when the supply voltage drops below the LVD voltage. If the LVD is enabled in stop (LVDE and LVDSE bits in SPMSC1 both set) at the time the CPU executes a STOP instruction, the voltage regulator remains active.

Table 3-4 summarizes the behavior of the MCU in stop when LVD reset is enabled.

Table 3-4. LVD Enabled Stop Mode Behavior

Mode	CPU	Digital Peripherals	ICS	ACMP	Regulator	I/O Pins	RTI
Stop	Standby	Standby	Optionally on	Optionally on	On	States held	Optionally on

Chapter 4
Memory

4.1 Memory Map

The memory map of the MCU is divided into the following groups:

- Fast access RAM using tiny and short instructions ($0000–$000E[1])
- Indirect data access D[X] ($000E)
- Index register X for D[X] ($000F)
- Frequently used peripheral registers ($0010–$001E)
- PAGESEL register ($001F)
- RAM ($0020–$004F)
- Paging window ($00C0–$00FF)
- Other peripheral registers ($0200–$023F)
- Nonvolatile memory
  - MC9RS08KA2: $3800–$3FFF
  - MC9RS08KA1: $3C00—$3FFF

1. Physical RAM in $000E can be accessed through the D[X] register when the content of the index register X is $0E.

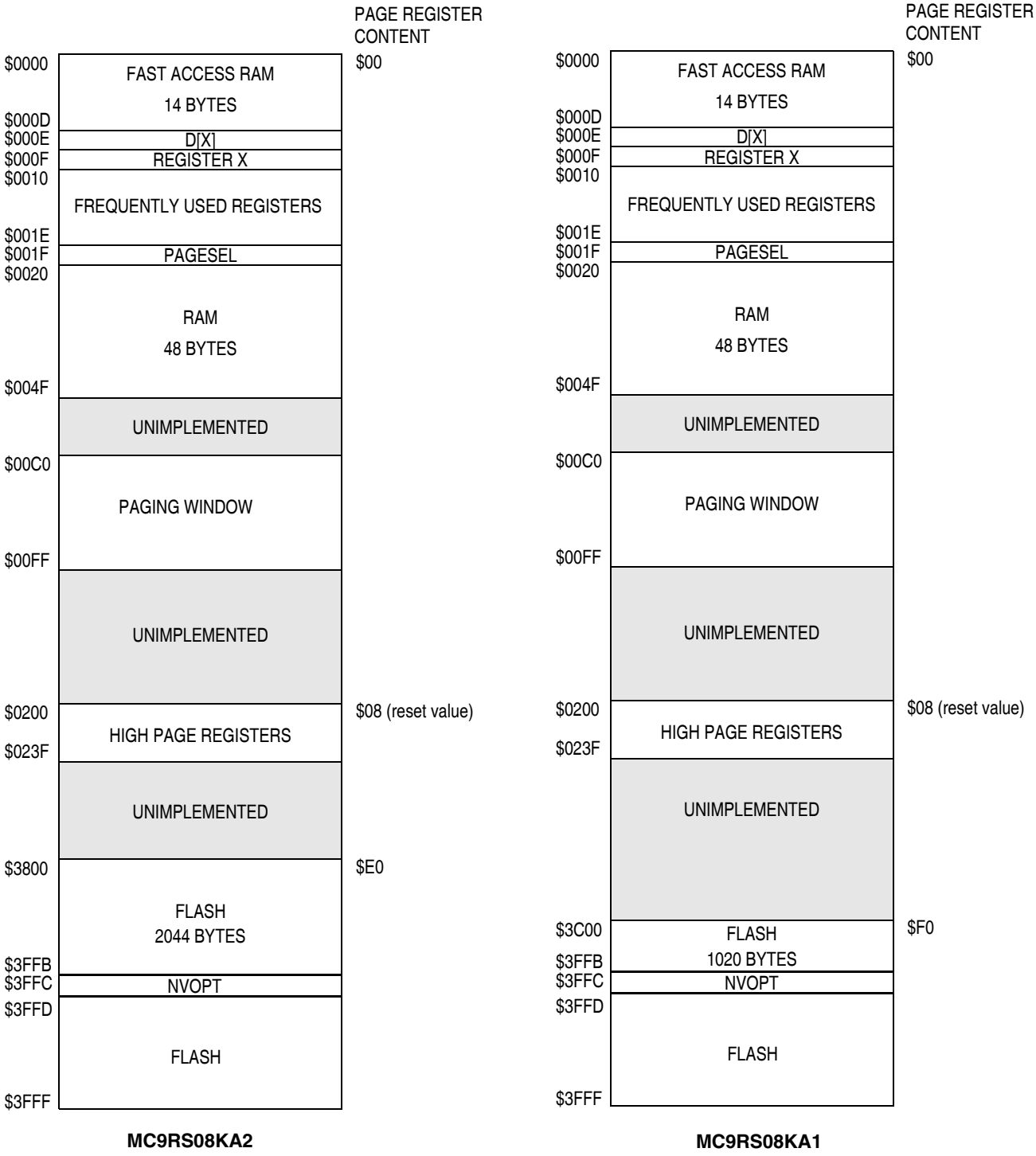


Figure 4-1. MC9RS08KA2 Series Memory Maps

4.2 Unimplemented Memory

Attempting to access either data or an instruction at an unimplemented memory address will cause reset.

4.3 Indexed/Indirect Addressing

Register D[X] and register X together perform the indirect data access. Register D[X] is mapped to address $000E. Register X is located in address $000F. The 8-bit register X contains the address that is used when register D[X] is accessed. Register X is cleared to zero upon reset. By programming register X, any location on the first page ($0000–$00FF) can be read/written via register D[X]. Figure 4-2 shows the relationship between D[X] and register X. For example, in HC08/S08 syntax *lda ,x* is comparable to *lda D[X]* in RS08 coding when register X has been programmed with the index value.

The physical location of $000E is in RAM. Accessing the location through D[X] returns $000E RAM content when register X contains $0E. The physical location of $000F is register X, itself. Reading the location through D[X] returns register X content; writing to the location modifies register X.

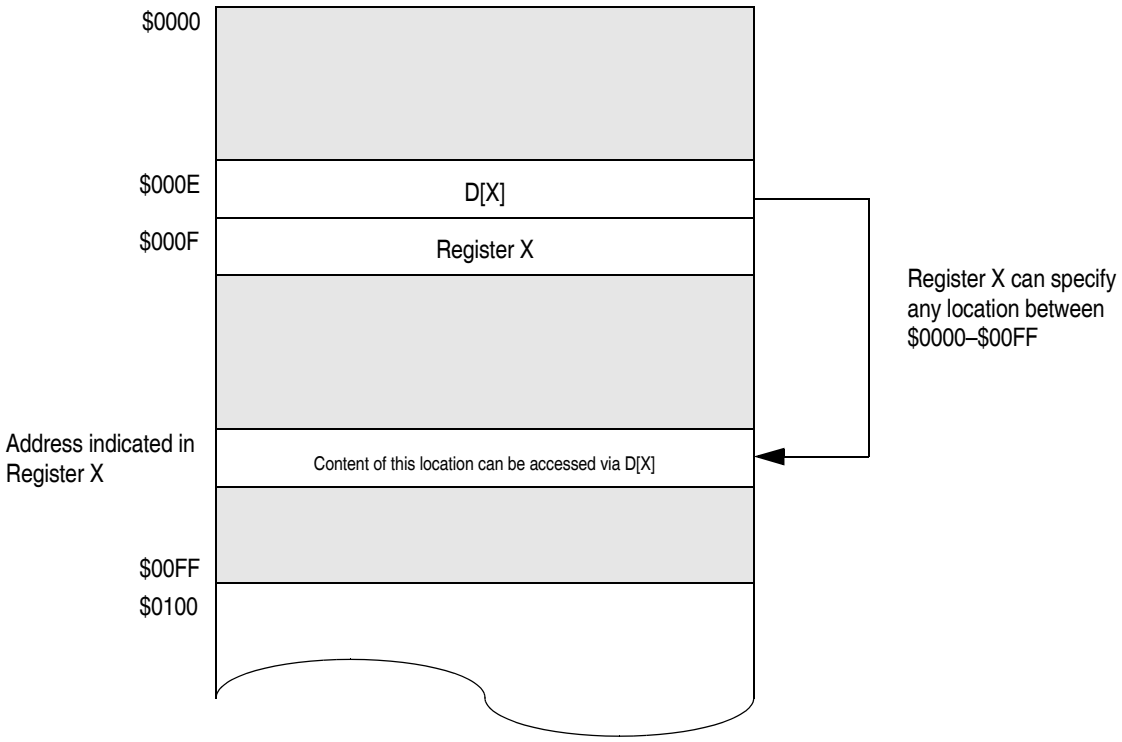


Figure 4-2. Indirect Addressing Registers

4.4 RAM and Register Addresses and Bit Assignments

The fast access RAM area can be accessed by instructions using tiny, short, and direct addressing mode instructions. For tiny addressing mode instructions, the operand is encoded along with the opcode to a single byte.

Frequently used registers can make use of the short addressing mode instructions for faster load, store, and clear operations. For short addressing mode instructions, the operand is encoded along with the opcode to a single byte.

Table 4-1. Register Summary

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
$0000–$000D		Fast Access RAM							
$000E	D[X][1]	Bit 7	6	5	4	3	2	1	Bit 0
$000F	X	Bit 7	6	5	4	3	2	1	Bit 0
$0010	PTAD	0	0	PTAD5	PTAD4	PTAD3	PTAD2	PTAD1	PTAD0
$0011	PTADD	0	0	PTADD5	PTADD4	0	0	PTADD1	PTADD0
$0012	Unimplemented	—	—	—	—	—	—	—	—
$0013	ACMPSC	ACME	ACBGS	ACF	ACIE	ACO	ACOPE	ACMOD	
$0014	ICSC1	0	CLKS	0	0	0	0	0	IREFSTEN
$0015	ICSC2	BDIV		0	0	LP	0	0	0
$0016	ICSTRM	TRIM							
$0017	ICSSC	0	0	0	0	0	CLKST	0	FTRIM
$0018	MTIMSC	TOF	TOIE	TRST	TSTP	0	0	0	0
$0019	MTIMCLK	0	0	CLKS		PS			
$001A	MTIMCNT	COUNT							
$001B	MTIMMOD	MOD							
$001C	KBISC	0	0	0	0	KBF	KBACK	KBIE	KBIMOD
$001D	KBIPE	—	—	KBIPE5	KBIPE4	—	KBIPE2	KBIPE1	KBIPE0
$001E	KBIES	—	—	KBEDG5	KBEDG4	—	KBEDG2	KBEDG1	KBEDG0
$001F	PAGESEL	AD13	AD12	AD11	AD10	AD9	AD8	AD7	AD6
$0020–$004F		RAM							
$0050–$00BF	Unimplemented	—	—	—	—	—	—	—	—
$00C0–$00FF		Paging Window							
$0100–$01FF	Unimplemented	—	—	—	—	—	—	—	—
$0200	SRS	POR	PIN	COP	ILOP	ILAD	0	LVD	0
$0201	SOPT	COPE	COPT	STOPE	0	0	0	BKGDPE	RSTPE
$0202	SIP1	—	—	—	KBI	ACMP	MTIM	RTI	LVD
$0203	Unimplemented	—	—	—	—	—	—	—	—
$0204	Reserved	—	—	—	—	—	—	—	—
$0205	Unimplemented	—	—	—	—	—	—	—	—
$0206	SDIDH	REV3	REV2	REV1	REV0	ID			
$0207	SDIDL	ID							
$0208	SRTISC	RTIF	RTIACK	RTICLKS	RTIE	0	RTIS		
$0209	SPMSC1	LVDF	LVDACK	LVDIE	LVDRE	LVDSE	LVDE	0	BGBE
$020A	Reserved	—	—	—	—	—	—	—	—
$020B	Reserved	—	—	—	—	—	—	—	—

= Unimplemented or Reserved

Table 4-1. Register Summary (continued)

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
$020C–$020F	Unimplemented	—	—	—	—	—	—	—	—
$0210	FOPT	0	0	0	0	0	0	0	SECD
$0211	FLCR	0	0	0	0	HVEN	MASS	0	PGM
$0212–$0213	Reserved	—	—	—	—	—	—	—	—
$0214–$021F	Unimplemented	— —	— —	— —	— —	— —	— —	— —	— —
$0220	PTAPE	0	0	PTAPE5	PTAPE4	0	PTAPE2	PTAPE1	PTAPE0
$0221	PTAPUD	0	0	PTAPUD5	PTAPUD4	0	PTAPUD2	PTAPUD1	PTAPUD0
$0222	PTASE	0	0	PTASE5	PTASE4	PTASE3	0	PTASE1	PTASE0
$0223–$023F	Unimplemented	—	—	—	—	—	—	—	—
$3FF8	Reserved	—	—	—	—	—	—	—	—
$3FF9	Reserved	—	—	—	—	—	—	—	—
$3FFA[2]	Reserved	Reserved for Room Temperature ICS Trim							
$3FFB[2]	Reserved	Reserved							FTRIM
$3FFC	NVOPT	0	0	0	0	0	0	0	SECD

= Unimplemented or Reserved

1 Physical RAM in $000E can be accessed through D[X] register when the content of the index register X is $0E.

2 If using the MCU untrimmed, $3FFA and $3FFB may be used by applications.

4.5 RAM

The device includes two sections of static RAM. The locations from $0000 to $000D can be directly accessed using the more efficient tiny addressing mode instructions and short addressing mode instructions. Location $000E RAM can either be accessed through D[X] register when register X is $0E or through the paging window location $00CE when PAGESEL register is $00. The second section of RAM starts from $0020 to $004F, and it can be accessed using direct addressing mode instructions.

The RAM retains data when the MCU is in low-power wait and stop mode. RAM data is unaffected by any reset provided that the supply voltage does not drop below the minimum value for RAM retention.

4.6 Flash

The Flash memory is intended primarily for program storage. In-circuit programming allows the operating program to be loaded into the Flash memory after final assembly of the application product. It is possible to program the entire array through the single-wire background debug interface. Because the device does not include on-chip charge pump circuitry, external V_{PP} is required for program and erase operations.

4.6.1 Features

Features of the Flash memory include:

- Up to 1000 program/erase cycles at typical voltage and temperature
- Security feature for Flash

4.6.2 Flash Programming Procedure

Programming of Flash memory is done on a row basis. A row consists of 64 consecutive bytes starting from addresses $3X00, $3X40, $3X80, or $3XC0. Use the following procedure to program a row of Flash memory:

1. Apply external V_{PP}.
2. Set the PGM bit. This configures the memory for program operation and enables the latching of address and data for programming.
3. Write any data to any Flash location, via the high page accessing window $00C0–$00FF, within the address range of the row to be programmed. (Prior to the data writing operation, the PAGESEL register must be configured correctly to map the high page accessing window to the corresponding Flash row).
4. Wait for a time, t_{nvs}.
5. Set the HVEN bit.
6. Wait for a time, t_{pgs}.
7. Write data to the Flash location to be programmed.
8. Wait for a time, t_{prog}.
9. Repeat steps 7 and 8 until all bytes within the row are programmed.
10. Clear the PGM bit.
11. Wait for a time, t_{nvh}.
12. Clear the HVEN bit.
13. After time, t_{rcv}, the memory can be accessed in read mode again.
14. Remove external V_{PP}.

This program sequence is repeated throughout the memory until all data is programmed.

NOTE

Flash memory cannot be programmed or erased by software code executed from Flash locations. To program or erase Flash, commands must be executed from RAM or BDC commands. User code should not enter wait or stop during erase or program sequence.

These operations must be performed in the order shown; other unrelated operations may occur between the steps.

4.6.3 Flash Mass Erase Operation

Use the following procedure to mass erase the entire Flash memory:

1. Apply external V_{PP}.
2. Set the MASS bit in the Flash control register.

3. Write any data to any Flash location, via the high page accessing window $00C0–$00FF. (Prior to the data writing operation, the PAGESEL register must be configured correctly to map the high page accessing window to the any Flash locations).
4. Wait for a time, t_{nvs}.
5. Set the HVEN bit.
6. Wait for a time t_{me}.
7. Clear the MASS bit.
8. Wait for a time, t_{nvh1}.
9. Clear the HVEN bit.
10. After time, t_{rcv}, the memory can be accessed in read mode again.
11. Remove external V_{PP}.

NOTE

Flash memory cannot be programmed or erased by software code executed from Flash locations. To program or erase Flash, commands must be executed from RAM or BDC commands. User code should not enter wait or stop during an erase or program sequence.

These operations must be performed in the order shown, but other unrelated operations may occur between the steps.

4.6.4 Security

The MC9RS08KA2 Series includes circuitry to help prevent unauthorized access to the contents of Flash memory. When security is engaged, Flash is considered a secure resource. The RAM, direct-page registers, and background debug controller are considered unsecured resources. Attempts to access a secure memory location through the background debug interface, or whenever BKGDPE is set, are blocked (reads return all 0s).

Security is engaged or disengaged based on the state of a nonvolatile register bit (SECD) in the FOPT register. During reset, the contents of the nonvolatile location NVOPT are copied from Flash into the working FOPT register in high-page register space. A user engages security by programming the NVOPT location, which can be done at the same time the Flash memory is programmed. Notice the erased state (SECD = 1) makes the MCU unsecure. When SECD in NVOPT is programmed (SECD = 0), next time the device is reset via POR, internal reset, or external reset, security is engaged. In order to disengage security, mass erase must be performed via BDM commands and followed by any reset.

The separate background debug controller can still be used for registers and RAM access. Flash mass erase is possible by writing to the Flash control register that follows the Flash mass erase procedure listed in Section 4.6.3, “Flash Mass Erase Operation,” via BDM commands.

Security can always be disengaged through the background debug interface by following these steps:

1. Mass erase Flash via background BDM commands or RAM loaded program.
2. Perform reset and the device will boot up with security disengaged.

NOTE

When the device boots up to normal operating mode, where MS pin is high during reset, with SECD programmed (SECD = 0), Flash security is engaged. BKGDPE is reset to 0, and all BDM communication is blocked, and background debug is not allowed.

4.7 Flash Registers and Control Bits

The Flash module has a nonvolatile register, NVOPT ($3FFC), in Flash memory which is copied into the corresponding control register, FOPT ($0210), at reset.

4.7.1 Flash Options Register (FOPT and NVOPT)

During reset, the contents of the nonvolatile location NVOPT is copied from Flash into FOPT. Bits 7 through 1 are not used and always read 0. This register may be read at any time, but writes have no meaning or effect. To change the value in this register, erase and reprogram the NVOPT location in Flash memory as usual and then issue a new MCU reset.

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	SECD
W								
Reset	This register is loaded from nonvolatile location NVOPT during reset.							

= Unimplemented or Reserved

Figure 4-3. Flash Options Register (FOPT)

Table 4-2. FOPT Field Descriptions

Field	Description
0 SECD	**Security State Code** — This bit field determines the security state of the MCU. When the MCU is secured, the contents of Flash memory cannot be accessed by instructions from any unsecured source including the background debug interface; refer to Section 4.6.4, “Security”. 0 Security engaged. 1 Security disengaged.

4.7.2 Flash Control Register (FLCR)

	7	6	5	4	3	2	1	0
R	0	0	0	0	HVEN	MASS	0	PGM[1]
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-4. Flash Control Register (FLCR)

Table 4-3. FLCR Field Descriptions

Field	Description
3 HVEN	**High Voltage Enable** — This read/write bit enables high voltages to the Flash array for program and erase operations. HVEN can be set only if either PGM = 1 or MASS = 1 and the proper sequence for program or erase is followed. 0 High voltage disabled to array. 1 High voltage enabled to array.
2 MASS	**Mass Erase Control Bit** — This read/write bit configures the memory for mass erase operation. 0 Mass erase operation not selected. 1 Mass erase operation selected.
0 PGM[1]	**Program Control Bit** — This read/write bit configures the memory for program operation. PGM is interlocked with the MASS bit such that both bits cannot be equal to 1 or set to 1 at the same time. 0 Program operation not selected. 1 Program operation selected.

[1] When Flash security is engaged, writing to PGM bit has no effect. As a result, Flash programming is not allowed.

4.8 Page Select Register (PAGESEL)

There is a 64-byte window ($00C0–$00FF) in the direct-page reserved for paging access. Programming the page select register determines the corresponding 64-byte block on the memory map for direct-page access. For example, when the PAGESEL register is programmed with value $08, the high page registers ($0200–$023F) can be accessed through the paging window ($00C0–$00FF) via direct addressing mode instructions.

	7	6	5	4	3	2	1	0
R	AD13	AD12	AD11	AD10	AD9	AD8	AD7	AD6
W								
Reset	0	0	0	0	1	0	0	0

Figure 4-5. Page Select Register (PAGESEL)

Table 4-4. PAGESEL Field Descriptions

Field	Description
7:0 AD[13:6]	**Page Selector**— These bits define the address line bit 6 to bit 13, which determines the 64-byte block boundary of the memory block accessed via the direct page window. See Figure 4-6 and Table 4-5.

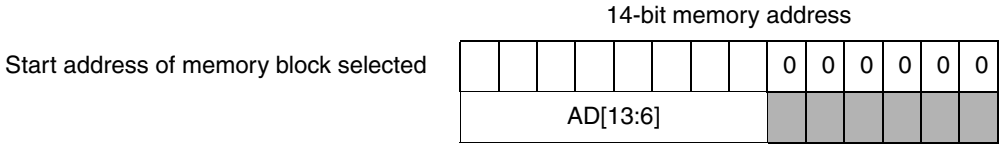


Figure 4-6. Memory Block Boundary Selector

Table 4-5 shows the memory block to be accessed through paging window ($00C0–$00FF).

Table 4-5. Paging Window for $00C0–$00FF

Page	Memory Address
$00	$0000–$003F
$01	$0040–$007F
$02	$0080–$00BF
$03	$00C0–$00FF
$04	$0100–$013F
. . .	. . .
$FE	$3F80–$3FBF
$FF	$3FC0–$3FFF

NOTE

Physical location $0000-$000E is RAM. Physical location $000F is register X. D[X] register is mapped to address $000E only. The physical RAM in $000E can be accessing through D[X] register when X register is either $0E or $CE with PAGESEL is $00.

When PAGESEL register is $00, paging window is mapped to the first page ($00-$3F). Paged location $00C0–$00CE is mapped to physical location $0000-$000E, i.e., RAM. Paged location $00CF is mapped to register X. Therefore, accessing address $CE returns the physical RAM content in $000E, accessing address $000E returns D[X] register content.

Chapter 5
Resets, Interrupts, and General System Control

5.1 Introduction

This chapter discusses basic reset and interrupt mechanisms and the various sources of reset and interrupt in the MC9RS08KA2 Series. Some interrupt sources from peripheral modules are discussed in greater detail within other chapters of this data sheet. This chapter gathers basic information about all reset and interrupt sources in one place for easy reference. A few reset and wakeup sources, including the computer operating properly (COP) watchdog and real-time interrupt (RTI), are not part of on-chip peripheral systems with their own chapters but are part of the system control logic.

5.2 Features

Reset and interrupt features include:

- Multiple sources of reset for flexible system configuration and reliable operation
- System reset status register (SRS) to indicate the source of the most recent reset
- System interrupt pending register (SIP1) to indicate the status of pending system interrupts
  - — Analog comparator interrupt with enable
  - — Keyboard interrupt with enable
  - — Low-voltage detect interrupt with enable
  - — Modulo timer interrupt with enable
  - — Real-time interrupt with enable; available in stop with multiple rates based on a separate 1-kHz self-clocked source

5.3 MCU Reset

Resetting the MCU provides a way to start processing from a known set of initial conditions. During reset, most control and status registers are forced to initial values and the program counter is started from location $3FFD. A JMP instruction (opcode $BC) with operand located at $3FFE–$3FFF must be programmed into the user application for correct reset operation. The operand defines the location at which the user program will start. On-chip peripheral modules are disabled and I/O pins are initially configured as general-purpose high-impedance inputs with pullup/pulldown devices disabled.

The MC9RS08KA2 Series has seven sources for reset:

- External pin reset (PIN) — enabled using RSTPE in SOPT
- Power-on reset (POR)
- Low-voltage detect (LVD)

- Computer operating properly (COP) timer
- Illegal opcode detect (ILOP)
- Illegal address detect (ILAD)
- Background debug forced reset via BDC command BDC_RESET

Each of these sources, with the exception of the background debug forced reset, has an associated bit in the system reset status register (SRS).

5.4 Computer Operating Properly (COP) Watchdog

The COP watchdog is intended to force a system reset if the application software fails to execute as expected. To prevent a system reset from the COP timer (when it is enabled), application software must reset the COP counter periodically. If the application program gets lost and fails to reset the COP counter before it times out, a system reset is generated to force the system back to a known starting point.

After any reset, the COPE becomes set in SOPT, which enables the COP watchdog (see Section 5.8.2, "System Options Register (SOPT)," for additional information). If the COP watchdog is not used in an application, it can be disabled by clearing COPE. The COP counter is reset by writing any value to the address of SRS. This write does not affect the data in the read-only SRS. Instead, the act of writing to this address is decoded and sends a reset signal to the COP counter.

There is an associated short and long time-out controlled by COPT in SOPT. Table 5-1 summaries the control functions of the COPT bit. The COP watchdog operates from the 1-kHz clock source and defaults to the associated long time-out (2^8 cycles).

Table 5-1. COP Configuration Options

COPT	COP Overflow Count[1]
0	2^5 cycles (32 ms)
1	2^8 cycles (256 ms)

[1] Values shown in this column are based on $t_{RTI} \approx 1$ ms. See t_{RTI} in the Section A.9.1, "Control Timing," for the tolerance of this value.

Even if the application will use the reset default settings of COPE and COPT, the user should write to the write-once SOPT registers during reset initialization to lock in the settings. That way, they cannot be changed accidentally if the application program gets lost. The initial write to SOPT will reset the COP counter.

In background debug mode, the COP counter will not increment.

When the MCU enters stop mode, the COP counter is re-initialized to zero upon entry to stop mode. The COP counter begins from zero as soon as the MCU exits stop mode.

5.5 Interrupts

The MC9RS08KA2 Series does not include an interrupt controller with vector table lookup mechanism as used on the HC08 and HCS08 devices. However, the interrupt sources from modules such as LVD, KBI,

and ACMP are still available to wake the CPU from wait or stop mode. It is the responsibility of the user application to poll the corresponding module to determine the source of wakeup.

Each wakeup source of the module is associated with a corresponding interrupt enable bit. If the bit is disabled, the interrupt source is gated, and that particular source cannot wake the CPU from wait or stop mode. However, the corresponding interrupt flag will still be set to indicate that an external wakeup event has occurred.

The system interrupt pending register (SIP1) indicates the status of the system pending interrupt. When the read-only bit of the SIP1 is enabled, it shows there is a pending interrupt to be serviced from the indicated module. Writing to the register bit has no effect. The pending interrupt flag will be cleared automatically when the all corresponding interrupt flags from the indicated module are cleared.

5.6 Low-Voltage Detect (LVD) System

The MC9RS08KA2 Series includes a system to protect against low voltage conditions in order to protect memory contents and control MCU system states during supply voltage variations. The system is comprised of a power-on reset (POR) circuit and an LVD circuit with a predefined trip voltage. The LVD circuit is enabled with LVDE in SPMSC1. The LVD is disabled upon entering stop mode unless LVDSE is set in SPMSC1. If LVDSE and LVDE are both set, the current consumption in stop with the LVD enabled will be greater.

5.6.1 Power-On Reset Operation

When power is initially applied to the MCU, or when the supply voltage drops below the V_{POR} level, the POR circuit will cause a reset condition. As the supply voltage rises, the LVD circuit will hold the MCU in reset until the supply has risen above the V_{LVD} level. Both the POR bit and the LVD bit in SRS are set following a POR.

5.6.2 LVD Reset Operation

The LVD can be configured to generate a reset upon detection of a low voltage condition by setting LVDRE to 1. After an LVD reset has occurred, the LVD system will hold the MCU in reset until the supply voltage has risen above the level V_{LVD}. The LVD bit in the SRS register is set following either an LVD reset or POR.

5.6.3 LVD Interrupt Operation

When a low voltage condition is detected and the LVD circuit is configured using SPMSC1 for interrupt operation (LVDE set, LVDIE set, and LVDRE clear), LVDF in SPMSC1 will be set and an LVD interrupt request will occur.

5.7 Real-Time Interrupt (RTI)

The real-time interrupt function can be used to generate periodic interrupts. The RTI is driven from either the 1-kHz internal clock reference or the trimmed 32-kHz internal clock reference from the ICS module. The 32-kHz internal clock reference is divided by 32 by the RTI logic to produce a trimmed 1-kHz clock

for applications requiring more accurate real-time interrupts. The RTICLKS bit in SRTISC is used to select the RTI clock source. Both the1-kHz and 32-kHz clock sources for the RTI can be used when the MCU is in run, wait or stop mode. For the 32-kHz clock source to run in stop, the LVDE and LVDSE bits in the SPMSC1 must both be set before entering stop.

The SRTISC register includes a read-only status flag, a write-only acknowledge bit, and a 3-bit control value (RTIS) used to select one of seven wakeup periods or disable RTI. The RTI has a local interrupt enable, RTIE, to allow masking of the real-time interrupt. The RTI can be disabled by writing each bit of RTIS to 0s, and no interrupts will be generated. See Section 5.8.4, "System Real-Time Interrupt Status and Control Register (SRTISC)," for detailed information about this register.

5.8 Reset, Interrupt, and System Control Registers and Control Bits

Refer to the direct-page register summary in Chapter 4, "Memory," for the absolute address assignments for all registers. This section refers to registers and control bits only by their names. A Freescale-provided equate or header file is used to translate these names into the appropriate absolute addresses.

Some control bits in the SOPT register are related to modes of operation. Although brief descriptions of these bits are provided here, the related functions are discussed in greater detail in Chapter 3, "Modes of Operation".

5.8.1 System Reset Status Register (SRS)

This high page register includes read-only status flags to indicate the source of the most recent reset. When a debug host forces reset by the BDC_RESET command, all of the status bits in SRS will be cleared. Writing any value to this register address clears the COP watchdog timer without affecting the contents of this register. The reset state of these bits depends on what caused the MCU to reset.

	7	6	5	4	3	2	1	0
R	POR	PIN	COP	ILOP	ILAD	0	LVD	0
W	Writing any value to SRS address clears COP watchdog timer.							
POR:	1	0	0	0	0	0	1	0
LVR:	0	0	0	0	0	0	1	0
Any other reset:	0	Note 1	Note 1	Note 1	Note 1	0	0	0

1. Any of these reset sources that are active at the time of reset entry will cause the corresponding bit(s) to be set; bits corresponding to sources that are not active at the time of reset entry will be cleared.

Figure 5-1. System Reset Status (SRS)

Table 5-2. SRS Field Descriptions

Field	Description
7 POR	**Power-On Reset** — Reset was caused by the power-on detection logic. Because the internal supply voltage was ramping up at the time, the low-voltage reset (LVR) status bit is also set to indicate that the reset occurred while the internal supply was below the LVR threshold. 0 Reset not caused by POR. 1 POR caused reset.
6 PIN	**External Reset Pin** — Reset was caused by an active-low level on the external reset pin. 0 Reset not caused by external reset pin. 1 External reset pin caused reset.
5 COP	**Computer Operating Properly (COP) Watchdog** — Reset was caused by the COP watchdog timer timing out. This reset source can be blocked by COPE = 0. 0 Reset not caused by COP timeout. 1 COP timeout caused reset.
4 ILOP	**Illegal Opcode** — Reset was caused by an attempt to execute an unimplemented or illegal opcode. The STOP instruction is considered illegal if stop is disabled by STOPE = 0 in the SOPT register. The BGND instruction is considered illegal if active background mode is disabled by ENBDM = 0 in the BDCSC register. 0 Reset not caused by an illegal opcode. 1 An illegal opcode caused reset.
3 ILAD	**Illegal Address** — Reset was caused by an attempt to access either data or an instruction at an unimplemented memory address. 0 Reset not caused by an illegal address. 1 An illegal address caused reset.
1 LVD	**Low Voltage Detect** — If the LVDRE bit is set and the supply drops below the LVD trip voltage, an LVD reset will occur. This bit is also set by POR. 0 Reset not caused by LVD trip or POR. 1 Either LVD trip or POR caused reset.

5.8.2 System Options Register (SOPT)

This high page register is a write-once register so only the first write after reset is honored. It can be read at any time. Any subsequent attempt to write to SOPT (intentionally or unintentionally) is ignored to avoid accidental changes to these sensitive settings. SOPT must be written during the user's reset initialization program to set the desired controls even if the desired settings are the same as the reset settings.

	7	6	5	4	3	2	1	0
R	COPE	COPT	STOPE	0	0	0	BKGDPE	RSTPE
W								
Reset:	1	1	0	0	0	0	1 (Note 1)	u
POR:	1	1	0	0	0	0	1 (Note1)	0

= Unimplemented or Reserved u = Unaffected

Figure 5-2. System Options Register 1 (SOPT)

1. When the device is reset into normal operating mode (MS is high during reset), BKGDPE is reset to 1 if Flash security is disengaged (SECD = 1); BKGDPE is reset to 0 if Flash security is engaged (SECD = 0). When the device is reset into active BDM mode (MS is low during reset), BKGDPE is always reset to 1 such that BDM communication is allowed.

Table 5-3. SOPT Register Field Descriptions

Field	Description
7 COPE	**COP Watchdog Enable** — This write-once bit selects whether the COP watchdog is enabled. 0 COP watchdog timer disabled. 1 COP watchdog timer enabled (force reset on timeout).
6 COPT	**COP Watchdog Timeout** — This write-once bit selects the timeout period of the COP. 0 Short timeout period selected. 1 Long timeout period selected.
5 STOPE	**Stop Mode Enable** — This write-once bit is used to enable stop mode. If stop mode is disabled and a user program attempts to execute a STOP instruction, an illegal opcode reset is forced. 0 Stop mode disabled. 1 Stop mode enabled.
1 BKGDPE[1,2]	**Background Debug Mode Pin Enable** — This write-once bit when set enables the PTA3/ACMPO/BKGD/MS pin to function as BKGD/MS. When clear, the pin functions as one of its output only alternative functions. This pin defaults to the BKGD/MS function following any MCU reset. 0 PTA3/ACMPO/BKGD/MS pin functions as PTA3 or ACMPO. 1 PTA3/ACMPO/BKGD/MS pin functions as BKGD/MS.
0 RSTPE	**$\overline{\text{RESET}}$ Pin Enable** — When set, this write-once bit enables the PTA2/KBIP2/TCLK/$\overline{\text{RESET}}$/V_{PP} pin to function as $\overline{\text{RESET}}$. When clear, the pin functions as one of its input-only alternative functions. This pin is input-only port function following an MCU POR. When RSTPE is set, an internal pullup device is enabled on $\overline{\text{RESET}}$. 0 PTA2/KBIP2/TCLK/$\overline{\text{RESET}}$/V_{PP} pin functions as PTA2/KBIP2/TCLK/V_{PP} 1 PTA2/KBIP2/TCLK/$\overline{\text{RESET}}$/V_{PP} pin functions as $\overline{\text{RESET}}$/V_{PP}

1 When the device is reset into normal operating mode (MS is high during reset), BKGDPE is reset to 1 if Flash security is disengaged (SECD = 1); BKGDPE is reset to 0 if Flash security is engaged (SECD = 0). When the device is reset into active BDM mode (MS is low during reset), BKGDPE is always reset to 1 such that BDM communication is allowed.

2 BKGDPE can only write once from value 1 to 0. Writing from value 0 to 1 by user software is not allowed. BKGDPE can be changed back to 1 only by a POR or reset with proper condition as stated in Note 1.

5.8.3 System Device Identification Register (SDIDH, SDIDL)

These high page read-only registers are included so host development systems can identify the RS08 derivative and revision number. This allows the development software to recognize where specific memory blocks, registers, and control bits are located in a target MCU.

	7	6	5	4	3	2	1	0
R	REV3	REV2	REV1	REV0	ID11	ID10	ID9	ID8
W								
Reset:	0 (Note 1)	0 (Note 1)	0 (Note 1)	0 (Note 1)	1	0	0	0

= Unimplemented or Reserved

1. The revision number that is hard coded into these bits reflects the current silicon revision level.

Figure 5-3. System Device Identification Register — High (SDIDH)

Table 5-4. SDIDH Register Field Descriptions

Field	Description
7:4 REV[3:0]	**Revision Number** — The high-order 4 bits of address SDIDH are hard coded to reflect the current mask set revision number (0–F).
3:0 ID[11:8]	**Part Identification Number** — Each derivative in the RS08 Family has a unique identification number. The MC9RS08KA2 Series is hard coded to the value $0800. See also ID bits in Figure 5-4.

	7	6	5	4	3	2	1	0
R	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 5-4. System Device Identification Register — Low (SDIDL)

Table 5-5. SDIDL Register Field Descriptions

Field	Description
7:0 ID[7:0]	**Part Identification Number** — Each derivative in the RS08 Family has a unique identification number. The MC9RS08KA2 Series is hard coded to the value $0800. See also ID bits in Figure 5-3.

5.8.4 System Real-Time Interrupt Status and Control Register (SRTISC)

This high page register contains status and control bits for the RTI.

	7	6	5	4	3	2	1	0
R	RTIF	0	RTICLKS	RTIE	0	RTIS		
W		RTIACK						
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 5-5. System RTI Status and Control Register (SRTISC)

Table 5-6. SRTISC Register Field Descriptions

Field	Description
7 RTIF	**Real-Time Interrupt Flag** — This read-only status bit indicates the periodic wakeup timer has timed out. 0 Periodic wakeup timer not timed out. 1 Periodic wakeup timer timed out.
6 RTIACK	**Real-Time Interrupt Acknowledge** — This write-only bit is used to acknowledge real-time interrupt request (write 1 to clear RTIF). Writing 0 has no meaning or effect. Reads always return 0.
5 RTICLKS	**Real-Time Interrupt Clock Select** — This read/write bit selects the clock source for the real-time interrupt. 0 Real-time interrupt request clock source is internal 1-kHz oscillator. 1 Real-time interrupt request clock source is internal trimmed 32-kHz oscillator (ICS module) and is divided by 32 in RTI logic to produce a trimmed 1-kHz clock source for RTI counter.

Table 5-6. SRTISC Register Field Descriptions (continued)

Field	Description
4 RTIE	**Real-Time Interrupt Enable** — This read-write bit enables real-time interrupts. 0 Real-time interrupts disabled. 1 Real-time interrupts enabled.
2:0 RTIS	**Real-Time Interrupt Delay Selects** — These read/write bits select the period for the RTI. See Table 5-7.

Table 5-7. Real-Time Interrupt Period

RTIS	RTI Timeout[1]
000	Disable RTI
001	8 ms
010	32 ms
011	64 ms
100	128 ms
101	256 ms
110	512 ms
111	1.024 s

1 Timeout values shown based on RTI clock source of 1 ms period. Consult electricals for tolerances of internal 1-kHz source, t_{RTI} (Table A-8) and the internal 32-kHz from ICS (Table A-7).

NOTE

To power down the internal 1-kHz oscillator completely in MCU STOP mode, RTIS bits must be selected to %000 and RTICLKS bit must be set to 1.

5.8.5 System Power Management Status and Control 1 Register (SPMSC1)

This high page register contains status and control bits to support the low voltage detect function, and to enable the bandgap voltage reference for use by the ACMP and the LVD module.

	7	6	5	4	3	2	1	0
R	LVDF	0	LVDIE	LVDRE[1]	LVDSE	LVDE[1]	0	BGBE
W		LVDACK						
Reset:	0	0	0	1	1	1	0	0

= Unimplemented or Reserved

1 This bit can be written only one time after reset. Additional writes are ignored.

Figure 5-6. System Power Management Status and Control 1 Register (SPMSC1)

Table 5-8. SPMSC1 Register Field Descriptions

Field	Description
7 LVDF	**Low-Voltage Detect Flag** — Provided LVDE = 1, this read-only status bit indicates a low-voltage detect event.
6 LVDACK	**Low-Voltage Detect Acknowledge** — This write-only bit is used to acknowledge low voltage detection errors (write 1 to clear LVDF). Reads always return 0.
5 LVDIE	**Low-Voltage Detect Interrupt Enable** — This bit enables hardware interrupt requests for LVDF. 0 Hardware interrupt disabled (use polling). 1 Request a hardware interrupt when LVDF = 1.
4 LVDRE	**Low-Voltage Detect Reset Enable** — This write-once bit enables low-voltage detect events to generate a hardware reset (provided LVDE = 1). 0 LVDF does not generate hardware resets. 1 Force an MCU reset when LVDF = 1.
3 LVDSE	**Low-Voltage Detect Stop Enable** — Provided LVDE = 1, this read/write bit determines whether the low-voltage detect function operates when the MCU is in stop mode. 0 Low-voltage detect disabled during stop mode. 1 Low-voltage detect enabled during stop mode.
2 LVDE	**Low-Voltage Detect Enable** — This write-once bit enables low-voltage detect logic and qualifies the operation of other bits in this register. 0 LVD logic disabled. 1 LVD logic enabled.
0 BGBE	**Bandgap Buffer Enable** — This bit enables an internal buffer for the bandgap voltage reference for use by the ACMP module on one of its internal channels. 0 Bandgap buffer disabled. 1 Bandgap buffer enabled.

5.8.6 System Interrupt Pending Register (SIP1)

This high page register contains status of the pending interrupt from the modules.

	7	6	5	4	3	2	1	0
R	0	0	0	KBI	ACMP	MTIM	RTI	LVD
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 5-7. System Interrupt Pending Register (SIP1)

Table 5-9. SIP1 Register Field Descriptions

Field	Description
4 KBI	**Keyboard Interrupt Pending** — This read-only bit indicates there is a pending interrupt from the KBI module. Clearing the KBF flag of the KBISC register clears this bit. Reset also clears this bit. 0 There is no pending KBI interrupt; i.e., KBF flag and/or KBIE bit is cleared. 1 There is a pending KBI interrupt; i.e., KBF flag and KBIE bit are set.
3 ACMP	**Analog Comparator Interrupt Pending** — This read-only bit indicates there is a pending interrupt from the ACMP module. Clearing the ACF flag of the ACMPSC register clears this bit. Reset also clears this bit. 0 There is no pending ACMP interrupt; i.e., ACF flag and/or ACIE bit is cleared. 1 There is a pending a ACMP interrupt; i.e., ACF flag and ACIE bit are set.
2 MTIM	**Modulo Timer Interrupt Pending** — This read-only bit indicates there is a pending interrupt from the MTIM module. Clearing the TOF flag of the MTIMSC register clears this bit. Reset also clears this bit. 0 There is no pending MTIM interrupt; i.e., TOF flag and/or TOIE bit is cleared. 1 There is a pending MTIM interrupt; i.e., TOF flag and TOIE bit are set.
1 RTI	**Real-Time Interrupt Pending** — This read-only bit indicates there is a pending interrupt from the RTI. Clearing the RTIF flag of the SRTISC register clears this bit. Reset also clears this bit. 0 There is no pending RTI interrupt; i.e., RTIF flag and/or RTIE bit is cleared. 1 There is a pending RTI interrupt; i.e., RTIF flag and RTIE bit are set.
0 LVD	**Low-Voltage Detect Interrupt Pending** — This read-only bit indicates there is a pending interrupt from the low voltage detect module. Clearing the LVDF flag of the SPMSC1 register clears this bit. Reset also clears this bit. 0 There is no pending LVD interrupt; i.e., LVDF flag and/or LVDE bit is cleared. 1 There is a pending LVD interrupt; i.e., LVDF flag, LVDIE, and LVDE bits are set.

Chapter 6
Parallel Input/Output Control

This section explains software controls related to parallel input/output (I/O) and pin control. The MC9RS08KA2 Series has one parallel I/O port, which includes two I/O pins in the 6-pin package or four I/O pins in the 8-pin packages, one output-only pin, and one input-only pin. See Chapter 2, "Pins and Connections," for more information about pin assignments and external hardware considerations for these pins.

All of these I/O pins are shared with on-chip peripheral functions as shown in Table 2-1. The peripheral modules have priority over the I/Os so that when a peripheral is enabled, the I/O functions associated with the shared pins are disabled. After reset, the shared peripheral functions are disabled so that the pins are controlled by the I/O. All of the I/Os are configured as inputs (PTADDn = 0) with pullup/pulldown devices disabled (PTAPEn = 0), except for output-only pin PTA3, which defaults to the BKGD/MS function.

Reading and writing of parallel I/Os is performed through the port data registers. The direction, either input or output, is controlled through the port data direction registers. The parallel I/O port function for an individual pin is illustrated in the block diagram shown in Figure 6-1.

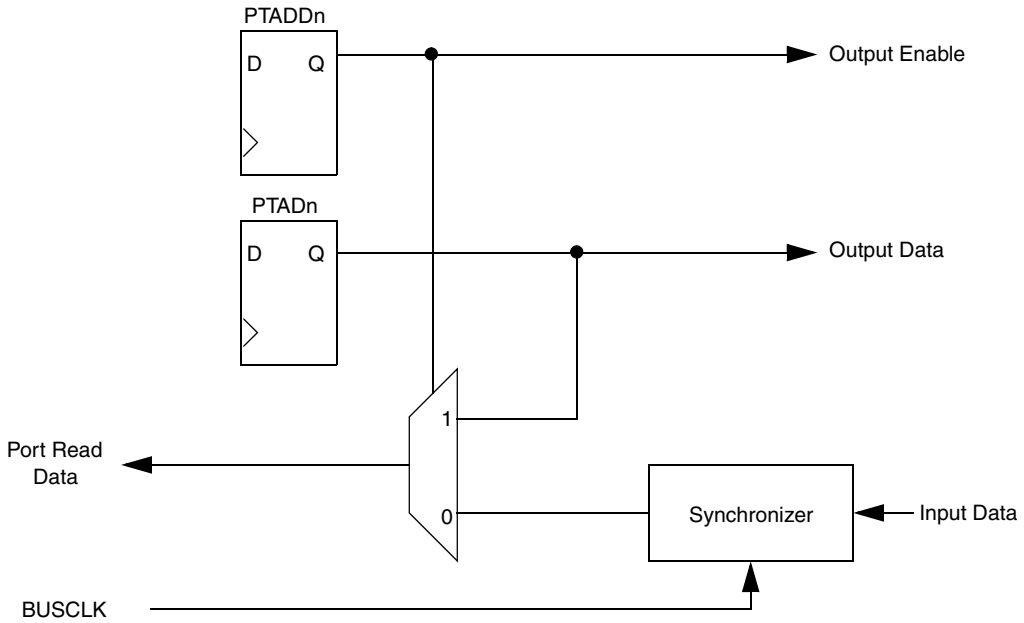


Figure 6-1. Parallel I/O Block Diagram

The data direction control bit (PTADDn) determines whether the output buffer for the associated pin is enabled, and also controls the source for port data register reads. The input buffer for the associated pin is always enabled unless the pin is enabled as an analog function or is an output-only pin.

When a shared digital function is enabled for a pin, the output buffer is controlled by the shared function. However, the data direction register bit will continue to control the source for reads of the port data register.

When a shared analog function is enabled for a pin, both the input and output buffers are disabled. A value of 0 is read for any port data bit where the bit is an input (PTADDn = 0) and the input buffer is disabled. In general, whenever a pin is shared with both an alternative digital function and an analog function, the analog function has priority such that if both the digital and analog functions are enabled, the analog function controls the pin.

It is a good programming practice to write to the port data register before changing the direction of a port pin to become an output. This ensures that the pin will not be driven temporarily with an old data value that happened to be in the port data register.

Associated with the parallel I/O ports is a set of registers located in the high page register space that operate independently of the parallel I/O registers. These registers are used to control pullup/pulldown and slew rate for the pins. See Section 6.3, “Pin Control Registers” for more information.

6.1 Pin Behavior in Low-Power Modes

In wait and stop modes, all pin states are maintained because internal logic stays powered up. Upon recovery, all pin functions are the same as before entering stop.

6.2 Parallel I/O Registers

This section provides information about the registers associated with the parallel I/O ports. The parallel I/O registers are located within the $001F memory boundary of the memory map, so that short and direct addressing mode instructions can be used.

Refer to tables in Chapter 4, “Memory,” for the absolute address assignments for all parallel I/O. This section refers to registers and control bits only by their names. A Freescale Semiconductor-provided equate or header file normally is used to translate these names into the appropriate absolute addresses.

6.2.1 Port A Registers

Port A parallel I/O function is controlled by the data and data direction registers described in this section.

	7	6	5	4	3	2	1	0
R	0	0	PTAD5	PTAD4	PTAD3	PTAD2	PTAD1	PTAD0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 6-2. Port A Data Register (PTAD)

Table 6-1. PTAD Register Field Descriptions

Field	Description
5:0 PTAD[5:0]	**Port A Data Register Bits** — For port A pins that are inputs, reads return the logic level on the pin. For port A pins that are configured as outputs, reads return the last value written to this register. Writes are latched into all bits of this register. For port A pins that are configured as outputs, the logic level is driven out the corresponding MCU pin. Reset forces PTAD to all 0s, but these 0s are not driven out the corresponding pins because reset also configures all port pins as high-impedance inputs with pullup/pulldowns disabled.

	7	6	5	4	3	2	1	0
R	0	0	PTADD5	PTADD4	0	0	PTADD1	PTADD0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 6-3. Port A Data Direction Register (PTADD)

Table 6-2. PTADD Register Field Descriptions

Field	Description
5:4,1:0 PTADD[5:4,1:0]	**Data Direction for Port A Bits** — These read/write bits control the direction of port A pins and what is read for PTAD reads. 0 Input (output driver disabled) and reads return the pin value. 1 Output driver enabled for port A bit n and PTAD reads return the contents of PTADn.

6.3 Pin Control Registers

This section provides information about the registers associated with the parallel I/O ports that are used for pin control functions.

Refer to tables in Chapter 4, "Memory," for the absolute address assignments of the pin control registers. This section refers to registers and control bits only by their names. A Freescale Semiconductor-provided equate or header file normally is used to translate these names into the appropriate absolute addresses.

6.3.1 Port A Pin Control Registers

The pins associated with port A are controlled by the registers provided in this section. These registers control the pin pullup/pulldown and slew rate of the port A pins independent of the parallel I/O registers.

6.3.1.1 Internal Pulling Device Enable

An internal pulling device can be enabled for each port pin by setting the corresponding bit in the pulling device enable register (PTAPEn). The pulling device is disabled if the pin is configured as an output by the parallel I/O control logic or any shared peripheral output function regardless of the state of the

corresponding pulling device enable register bit. The pulling device is also disabled if the pin is controlled by an analog function.

	7	6	5	4	3	2	1	0
R	0	0	PTAPE5	PTAPE4	0	PTAPE2	PTAPE1	PTAPE0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 6-4. Internal Pulling Device Enable for Port A Register (PTAPE)

Table 6-3. PTAPE Register Field Descriptions

Field	Description
5:4,2:0 PTAPE[5:4,2:0]	**Internal Pulling Device Enable for Port A Bits** — Each of these control bits determines whether the internal pulling device is enabled for the associated PTA pin. For port A pins that are configured as outputs, these bits have no effect and the internal pullup devices are disabled. 0 Internal pulling device disabled for port A bit n. 1 Internal pulling device enabled for port A bit n.

6.3.1.2 Pullup/Pulldown Control

Pullup/pulldown control is used to select the pullup or pulldown device enabled by the corresponding PTAPE bit.

	7	6	5	4	3	2	1	0
R	0	0	PTAPUD5	PTAPUD4	0	PTAPUD2	PTAPUD1	PTAPUD0
W								
Reset:	0	0	0	0	0	0	0	0

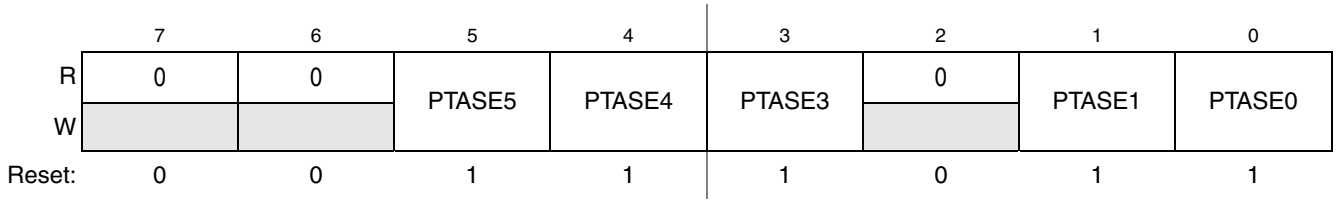
Figure 6-5. Pullup/Pulldown Device Control for Port A (PTAPUD)

Table 6-4. PTAPUD Register Field Descriptions

Field	Description
5:4,2:0 PTAPUD[5:4,2:0]	**Pullup/Pulldown Device Control for Port A Bits** — Each of these control bits determines whether the internal pullup or pulldown device is selected for the associated PTA pin. The actual pullup/pulldown device is only enabled by enabling the associated PTAPE bit. 0 Internal pullup device is selected for port A bit n. 1 Internal pulldown device is selected for port A bit n.

6.3.1.3 Output Slew Rate Control Enable

Slew rate control can be enabled for each port pin by setting the corresponding bit in the slew rate control register (PTASEn). When enabled, slew control limits the rate at which an output can transition in order to reduce EMC emissions. Slew rate control has no effect on pins that are configured as inputs.



	7	6	5	4	3	2	1	0
R	0	0	PTASE5	PTASE4	PTASE3	0	PTASE1	PTASE0
W								
Reset:	0	0	1	1	1	0	1	1

Figure 6-6. Slew Rate Enable for Port A Register (PTASE)

Table 6-5. PTASE Register Field Descriptions

Field	Description
5:3;1:0 PTASE[5:3;1:0]	**Output Slew Rate Enable for Port A Bits** — Each of these control bits determines whether the output slew rate control is enabled for the associated PTA pin. For port A pins that are configured as inputs, these bits have no effect. 0 Output slew rate control disabled for port A bit n. 1 Output slew rate control enabled for port A bit n.

Chapter 7
Keyboard Interrupt (RS08KBIV1)

7.1 Introduction

The keyboard interrupt (KBI) module provides independently enabled external interrupt sources.

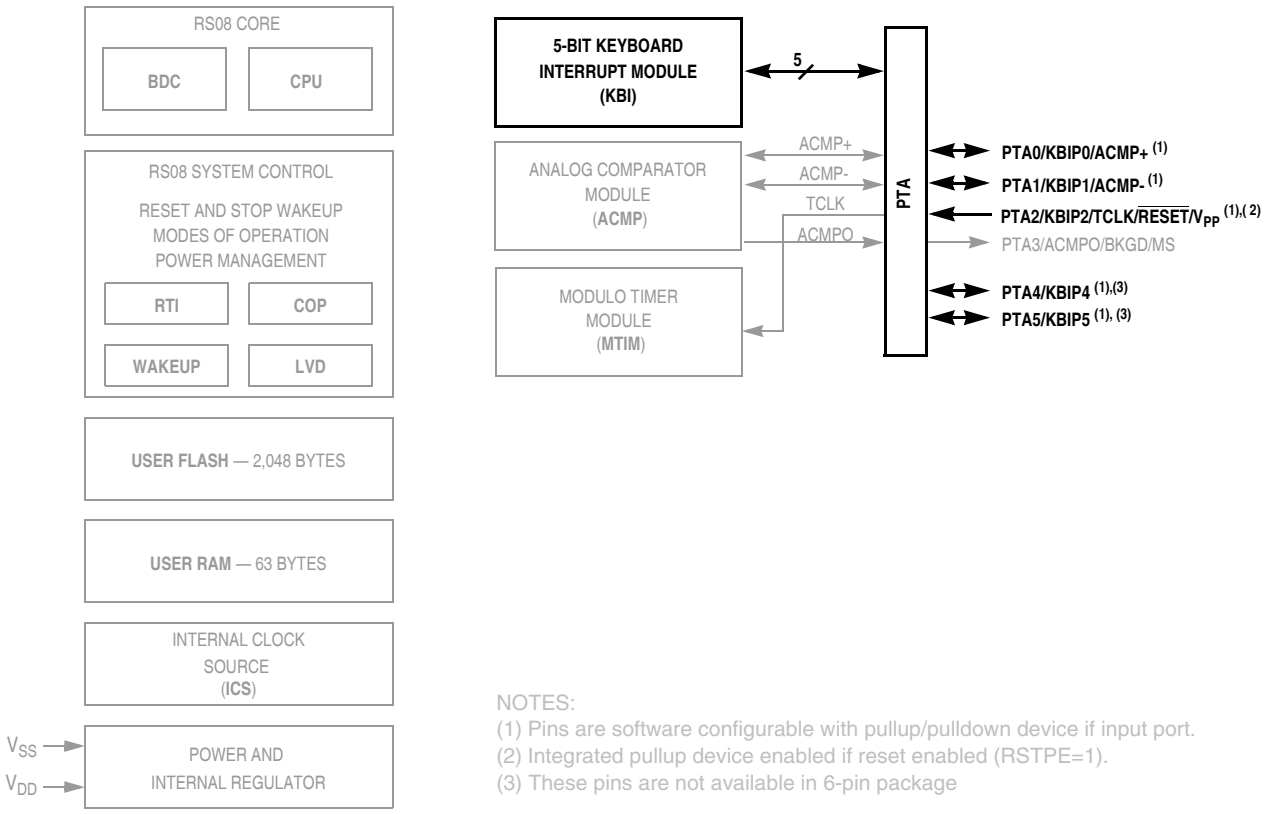


Figure 7-1. MC9RS08KA2 Series Block Diagram with KBI Block and Pins Highlighted

7.1.1 Features

The KBI features include:

- Each keyboard interrupt pin has an individual pin enable bit
- Each keyboard interrupt pin is programmable as falling edge (or rising edge) only, or both falling edge and low level (or both rising edge and high level) interrupt sensitivity
- One software-enabled keyboard interrupt
- Exit from low-power modes

7.1.2 Modes of Operation

This section defines the KBI operation in wait, stop, and background debug modes.

7.1.2.1 Operation in Wait Mode

The KBI continues to operate in wait mode if enabled before executing the WAIT instruction. Therefore, an enabled KBI pin (KBPEn = 1) can be used to bring the MCU out of wait mode if the KBI interrupt is enabled (KBIE = 1).

7.1.2.2 Operation in Stop Mode

The KBI operates asynchronously in stop mode if enabled before executing the STOP instruction. Therefore, an enabled KBI pin (KBPEn = 1) can be used to bring the MCU out of stop mode if the KBI interrupt is enabled (KBIE = 1).

7.1.2.3 Operation in Active Background Mode

When the microcontroller is in active background mode, the KBI will continue to operate normally.

7.1.3 Block Diagram

The block diagram for the keyboard interrupt module is shown Figure 7-2.

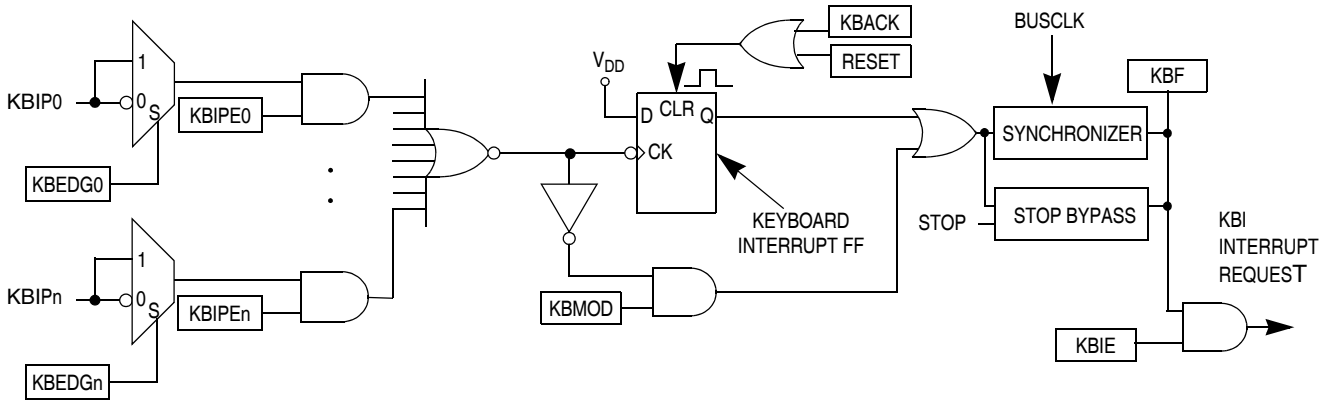


Figure 7-2. Keyboard Interrupt (KBI) Block Diagram

7.2 External Signal Description

The KBI input pins can be used to detect either falling edges, or both falling edge and low level interrupt requests. The KBI input pins can also be used to detect either rising edges, or both rising edge and high level interrupt requests.

The signal properties of KBI are shown in Table 7-1.

Table 7-1. Signal Properties

Signal	Function	I/O
KBIPn	Keyboard interrupt pins	I

7.3 Register Definition

The KBI includes three registers:

- An 8-bit pin status and control register
- An 8-bit pin enable register
- An 8-bit edge select register

Refer to the direct-page register summary in Chapter 4, "Memory," for the absolute address assignments for all KBI registers. This section refers to registers and control bits only by their names.

The KBI registers are summarized in Table 7-2.

Table 7-2. KBI Register Summary

Name		7	6	5	4	3	2	1	0
KBISC	R	0	0	0	0	KBF	0	KBIE	KBMOD
	W						KBACK		
KBIPE	R	0	0	KBIPE5	KBIPE4	0	KBIPE2	KBIPE1	KBIPE0
	W								
KBIES	R	0	0	KBEDG5	KBEDG4	0	KBEDG2	KBEDG1	KBEDG0
	W								

7.3.1 KBI Status and Control Register (KBISC)

KBISC contains the status flag and control bits, which are used to configure the KBI.

	7	6	5	4	3	2	1	0
R	0	0	0	0	KBF	0	KBIE	KBMOD
W						KBACK		
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 7-3. KBI Status and Control Register (KBISC)

Table 7-3. KBISC Register Field Descriptions

Field	Description
3 KBF	**Keyboard Interrupt Flag** — KBF indicates that a keyboard interrupt is detected. Writes have no effect on KBF. 0 No keyboard interrupt detected. 1 Keyboard interrupt detected.
2 KBACK	**Keyboard Acknowledge** — Writing a 1 to KBACK is part of the flag-clearing mechanism. KBACK always reads as 0.

Table 7-3. KBISC Register Field Descriptions (continued)

Field	Description
1 KBIE	**Keyboard Interrupt Enable** — KBIE enables keyboard interrupt requests. 0 Keyboard interrupt request not enabled. 1 Keyboard interrupt request enabled.
0 KBMOD	**Keyboard Detection Mode** — KBMOD (along with the KBEDG bits) controls the detection mode of the keyboard interrupt pins. 0 Keyboard detects edges only. 1 Keyboard detects both edges and levels.

7.3.2 KBI Pin Enable Register (KBIPE)

KBIPE contains the pin enable control bits.

	7	6	5	4	3	2	1	0
R	0	0	KBIPE5	KBIPE4	0	KBIPE2	KBIPE1	KBIPE0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 7-4. KBI Pin Enable Register (KBIPE)

Table 7-4. KBIPE Register Field Descriptions

Field	Description
5,4, 2:0 KBIPEn	**Keyboard Pin Enables** — Each of the KBIPEn bits enables the corresponding keyboard interrupt pin. 0 Corresponding pin not enabled as keyboard interrupt. 1 Corresponding pin enabled as keyboard interrupt.

7.3.3 KBI Edge Select Register (KBIES)

KBIES contains the edge select control bits.

	7	6	5	4	3	2	1	0
R	0	0	KBEDG5	KBEDG4	0	KBEDG2	KBEDG1	KBEDG0
W								
Reset:	0	0	0	0	0	0	0	0

Figure 7-5. KBI Edge Select Register (KBIES)

Table 7-5. KBIES Register Field Descriptions

Field	Description
5,4, 2:0 KBEDGn	**Keyboard Edge Selects** — Each of the KBEDGn bits selects the falling edge/low level or rising edge/high level function of the corresponding pin. 0 Falling edge/low level. 1 Rising edge/high level.

7.4 Functional Description

This on-chip peripheral module is called a keyboard interrupt (KBI) module because it was originally designed to simplify the connection and use of row-column matrices of keyboard switches. However, these inputs are also useful as extra external interrupt inputs and as an external means of waking the MCU from stop or wait low-power modes.

The KBI module allows its pins to act as additional interrupt sources. Writing to the KBIPEn bits in the keyboard interrupt pin enable register (KBIPE) independently enables or disables each KBI pin. Each KBI pin can be configured as edge sensitive or edge and level sensitive based on the KBMOD bit in the keyboard interrupt status and control register (KBISC). Edge sensitive can be software programmed to be either falling or rising; the level can be either low or high. The polarity of the edge or edge and level sensitivity is selected using the KBEDGn bits in the keyboard interrupt edge select register (KBIES).

Synchronous logic is used to detect edges. Prior to detecting an edge, enabled keyboard inputs must be at the deasserted logic level. A falling edge is detected when an enabled keyboard input signal is seen as a logic 1 (the deasserted level) during one bus cycle and then a logic 0 (the asserted level) during the next cycle. A rising edge is detected when the input signal is seen as a logic 0 during one bus cycle and then a logic 1 during the next cycle.

7.4.1 Edge Only Sensitivity

A valid edge on an enabled KBI pin will set KBF in KBISC. If KBIE in KBISC is set, an interrupt request will be presented to the CPU. Clearing of KBF is accomplished by writing a 1 to KBACK in KBISC.

7.4.2 Edge and Level Sensitivity

A valid edge or level on an enabled KBI pin will set KBF in KBISC. If KBIE in KBISC is set, an interrupt request will be presented to the CPU. Clearing of KBF is accomplished by writing a 1 to KBACK in KBISC, provided all enabled keyboard inputs are at their deasserted levels. KBF will remain set if any enabled KBI pin is asserted while attempting to clear by writing a 1 to KBACK.

7.4.3 KBI Pullup/Pulldown Device

The KBI pins does not automatically configure an internal pullup/pulldown device when a KBI pin is enabled. An internal pull device can be used by configuring the associated I/O port pull device enable register (PTAPE) and pullup/pulldown control register (PTAPUD).

7.4.4 KBI Initialization

When a keyboard interrupt pin is first enabled, it is possible to get a false keyboard interrupt flag. To prevent a false interrupt request during keyboard initialization, the user should do the following:

1. Mask keyboard interrupts by clearing KBIE in KBISC.
2. If using internal pullup/pulldown device, configure the associated I/O port pullup/pulldown device.
3. Enable the KBI polarity by setting the appropriate KBEDGn bits in KBIES.
4. Enable the KBI pins by setting the appropriate KBIPEn bits in KBIPE.

5. Write to KBACK in KBISC to clear any false interrupts.
6. Set KBIE in KBISC to enable interrupts.

Chapter 8
Central Processor Unit (RS08CPUV1)

8.1 Introduction

This chapter is a summary of information about the registers, addressing modes, and instruction set of the RS08 Family CPU. For a more detailed discussion, refer to the RS08 Core Reference Manual, volume 1, Freescale Semiconductor document order number RS08RMv1.

The RS08 CPU has been developed to target extremely low-cost embedded applications using a process-independent design methodology, allowing it to keep pace with rapid developments in silicon processing technology.

The main features of the RS08 core are:

- Streamlined programmer's model
- Subset of HCS08 instruction set with minor instruction extensions
- Minimal instruction set for cost-sensitive embedded applications
- New instructions for shadow program counter manipulation, SHA and SLA
- New short and tiny addressing modes for code size optimization
- 16K bytes accessible memory space
- Reset will fetch the first instruction from $3FFD
- Low-power modes supported through the execution of the STOP and WAIT instructions
- Debug and FLASH programming support using the background debug controller module
- Illegal address and opcode detection with reset

8.2 Programmer's Model and CPU Registers

Figure 8-1 shows the programmer's model for the RS08 CPU. These registers are not located in the memory map of the microcontroller. They are built directly inside the CPU logic.

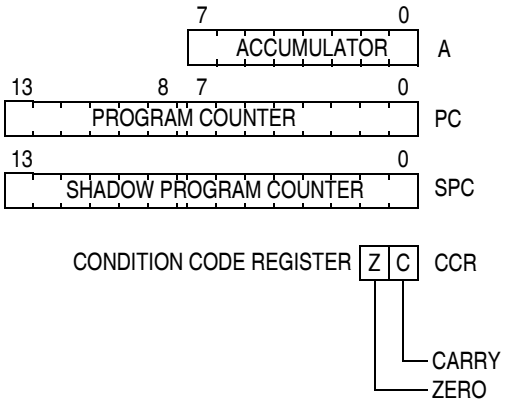


Figure 8-1. CPU Registers

In addition to the CPU registers, there are three memory mapped registers that are tightly coupled with the core address generation during data read and write operations. They are the indexed data register (D[X]), the index register (X), and the page select register (PAGESEL). These registers are located at $000E, $000F, and $001F, respectively.

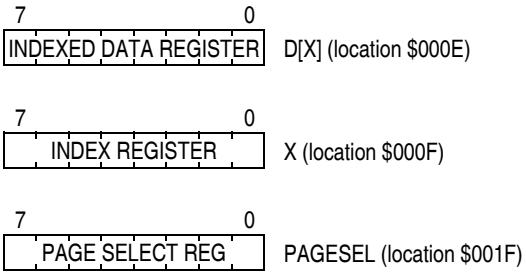


Figure 8-2. Memory Mapped Registers

8.2.1 Accumulator (A)

This general-purpose 8-bit register is the primary data register for RS08 MCUs. Data can be read from memory into A with a load accumulator (LDA) instruction. The data in A can be written into memory with a store accumulator (STA) instruction. Various addressing mode variations allow a great deal of flexibility in specifying the memory location involved in a load or store instruction. Exchange instructions allow values to be exchanged between A and SPC high (SHA) and also between A and SPC low (SLA).

Arithmetic, shift, and logical operations can be performed on the value in A as in ADD, SUB, RORA, INCA, DECA, AND, ORA, EOR, etc. In some of these instructions, such as INCA and LSLA, the value in A is the only input operand and the result replaces the value in A. In other cases, such as ADD and AND, there are two operands: the value in A and a second value from memory. The result of the arithmetic or logical operation replaces the value in A.

Some instructions, such as memory-to-memory move instructions (MOV), do not use the accumulator. DBNZ also relieves A because it allows a loop counter to be implemented in a memory variable rather than the accumulator.

During reset, the accumulator is loaded with $00.

8.2.2 Program Counter (PC)

The program counter is a 14-bit register that contains the address of the next instruction or operand to be fetched.

During normal execution, the program counter automatically increments to the next sequential memory location each time an instruction or operand is fetched. Jump, branch, and return operations load the program counter with an address other than that of the next sequential location. This is called a change-of-flow.

During reset, the program counter is loaded with $3FFD and the program will start execution from this specific location.

8.2.3 Shadow Program Counter (SPC)

The shadow program counter is a 14-bit register. During a subroutine call using either a JSR or a BSR instruction, the return address will be saved into the SPC. Upon completion of the subroutine, the RTS instruction will restore the content of the program counter from the shadow program counter.

During reset, the shadow program counter is loaded with $3FFD.

8.2.4 Condition Code Register (CCR)

The 2-bit condition code register contains two status flags. The content of the CCR in the RS08 is not directly readable. The CCR bits can be tested using conditional branch instructions such as BCC and BEQ. These two register bits are directly accessible through the BDC interface. The following paragraphs provide detailed information about the CCR bits and how they are used. Figure 8-3 identifies the CCR bits and their bit positions.

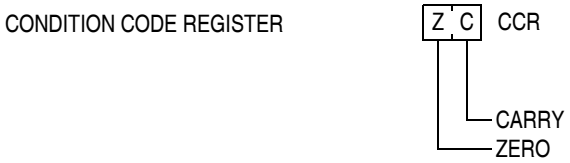


Figure 8-3. Condition Code Register (CCR)

The status bits (Z and C) are cleared to 0 after reset.

The two status bits indicate the results of arithmetic and other instructions. Conditional branch instructions will either branch to a new program location or allow the program to continue to the next instruction after the branch, depending on the values in the CCR status bit. Conditional branch instructions, such as BCC, BCS, and BNE, cause a branch depending on the state of a single CCR bit.

Often, the conditional branch immediately follows the instruction that caused the CCR bit(s) to be updated, as in this sequence:

```
        cmp     #5        ;compare accumulator A to 5
        blo     lower     ;branch if A smaller 5
more:   deca              ;do this if A not higher than or same as 5
lower:
```

Other instructions may be executed between the test and the conditional branch as long as the only instructions used are those which do not disturb the CCR bits that affect the conditional branch. For instance, a test is performed in a subroutine or function and the conditional branch is not executed until the subroutine has returned to the main program. This is a form of parameter passing (that is, information is returned to the calling program in the condition code bits).

Z — Zero Flag

The Z bit is set to indicate the result of an operation was $00.

Branch if equal (BEQ) and branch if not equal (BNE) are simple branches that branch based solely on the value in the Z bit. All load, store, move, arithmetic, logical, shift, and rotate instructions cause the Z bit to be updated.

C — Carry

After an addition operation, the C bit is set if the source operands were both greater than or equal to $80 or if one of the operands was greater than or equal to $80 and the result was less than $80. This is equivalent to an unsigned overflow. A subtract or compare performs a subtraction of a memory operand from the contents of a CPU register so after a subtract operation, the C bit is set if the unsigned value of the memory operand was greater than the unsigned value of the CPU register. This is equivalent to an unsigned borrow or underflow.

Branch if carry clear (BCC) and branch if carry set (BCS) are branches that branch based solely on the value in the C bit. The C bit is also used by the unsigned branches BLO and BHS. Add, subtract, shift, and rotate instructions cause the C bit to be updated. The branch if bit set (BRSET) and branch if bit clear (BRCLR) instructions copy the tested bit into the C bit to facilitate efficient serial-to-parallel conversion algorithms. Set carry (SEC) and clear carry (CLC) allow the carry bit to be set or cleared directly. This is useful in combination with the shift and rotate instructions and for routines that pass status information back to a main program, from a subroutine, in the C bit.

The C bit is included in shift and rotate operations so those operations can easily be extended to multi-byte operands. The shift and rotate operations can be considered 9-bit shifts that include an 8-bit operand or CPU register and the carry bit of the CCR. After a logical shift, C holds the bit that was shifted out of the 8-bit operand. If a rotate instruction is used next, this C bit is shifted into the operand for the rotate, and the bit that gets shifted out the other end of the operand replaces the value in C so it can be used in subsequent rotate instructions.

8.2.5 Indexed Data Register (D[X])

This 8-bit indexed data register allows the user to access the data in the direct page address space indexed by X. This register resides at the memory mapped location $000E. For details on the D[X] register, please refer to Section 8.3.8, “Indexed Addressing Mode (IX, Implemented by Pseudo Instructions).”

8.2.6 Index Register (X)

This 8-bit index register allows the user to index or address any location in the direct page address space. This register resides at the memory mapped location $000F. For details on the X register, please refer to Section 8.3.8, “Indexed Addressing Mode (IX, Implemented by Pseudo Instructions).”

8.2.7 Page Select Register (PAGESEL)

This 8-bit page select register allows the user to access all memory locations in the entire 16K-byte address space through a page window located from $00C0 to $00FF. This register resides at the memory mapped location $001F. For details on the PAGESEL register, please refer to the RS08 Core Reference Manual.

8.3 Addressing Modes

Whenever the MCU reads information from memory or writes information into memory, an addressing mode is used to determine the exact address where the information is read from or written to. This section explains several addressing modes and how each is useful in different programming situations.

Every opcode tells the CPU to perform a certain operation in a certain way. Many instructions, such as load accumulator (LDA), allow several different ways to specify the memory location to be operated on, and each addressing mode variation requires a separate opcode. All of these variations use the same instruction mnemonic, and the assembler knows which opcode to use based on the syntax and location of the operand field. In some cases, special characters are used to indicate a specific addressing mode (such as the # [pound] symbol, which indicates immediate addressing mode). In other cases, the value of the operand tells the assembler which addressing mode to use. For example, the assembler chooses short addressing mode instead of direct addressing mode if the operand address is from $0000 to $001F. Besides allowing the assembler to choose the addressing mode based on the operand address, assembler directives can also be used to force direct or tiny/short addressing mode by using the “>” or “<” prefix before the operand, respectively.

Some instructions use more than one addressing mode. For example, the move instructions use one addressing mode to access the source value from memory and a second addressing mode to access the destination memory location. For these move instructions, both addressing modes are listed in the documentation. All branch instructions use relative (REL) addressing mode to determine the destination for the branch, but BRCLR, BRSET, CBEQ, and DBNZ also must access a memory operand. These instructions are classified by the addressing mode used for the memory operand, and the relative addressing mode for the branch offset is assumed.

The discussion in the following paragraphs includes how each addressing mode works and the syntax clues that instruct the assembler to use a specific addressing mode.

8.3.1 Inherent Addressing Mode (INH)

This addressing mode is used when the CPU inherently knows everything it needs to complete the instruction and no addressing information is supplied in the source code. Usually, the operands that the CPU needs are located in the CPU’s internal registers, as in LSLA, CLRA, INCA, SLA, RTS, and others. A few inherent instructions, including no operation (NOP) and background (BGND), have no operands.

8.3.2 Relative Addressing Mode (REL)

Relative addressing mode is used to specify the offset address for branch instructions relative to the program counter. Typically, the programmer specifies the destination with a program label or an

expression in the operand field of the branch instruction; the assembler calculates the difference between the location counter (which points at the next address after the branch instruction at the time) and the address represented by the label or expression in the operand field. This difference is called the offset and is an 8-bit two's complement number. The assembler stores this offset in the object code for the branch instruction.

During execution, the CPU evaluates the condition that controls the branch. If the branch condition is true, the CPU sign-extends the offset to a 14-bit value, adds the offset to the current PC, and uses this as the address where it will fetch the next instruction and continue execution rather than continuing execution with the next instruction after the branch. Because the offset is an 8-bit two's complement value, the destination must be within the range –128 to +127 locations from the address that follows the last byte of object code for the branch instruction.

A common method to create a simple infinite loop is to use a branch instruction that branches to itself. This is sometimes used to end short code segments during debug. Typically, to get out of this infinite loop, use the debug host (through background commands) to stop the program, examine registers and memory, or to start execution from a new location. This construct is not used in normal application programs except in the case where the program has detected an error and wants to force the COP watchdog timer to timeout. (The branch in the infinite loop executes repeatedly until the watchdog timer eventually causes a reset.)

8.3.3 Immediate Addressing Mode (IMM)

In this addressing mode, the operand is located immediately after the opcode in the instruction stream. This addressing mode is used when the programmer wants to use an explicit value that is known at the time the program is written. A # (pound) symbol is used to tell the assembler to use the operand as a data value rather than an address where the desired value will be accessed.

The size of the immediate operand is always 8 bits. The assembler automatically will truncate or extend the operand as needed to match the size needed for the instruction. Most assemblers generate a warning if a 16-bit operand is provided.

It is the programmer's responsibility to use the # symbol to tell the assembler when immediate addressing will be used. The assembler does not consider it an error to leave off the # symbol because the resulting statement is still a valid instruction (although it may mean something different than the programmer intended).

8.3.4 Tiny Addressing Mode (TNY)

TNY addressing mode is capable of addressing only the first 16 bytes in the address map, from $0000 to $000F. This addressing mode is available for INC, DEC, ADD, and SUB instructions. A system can be optimized by placing the most computation-intensive data in this area of memory.

Because the 4-bit address is embedded in the opcode, only the least significant four bits of the address must be included in the instruction; this saves program space and execution time. During execution, the CPU adds 10 high-order 0s to the 4-bit operand address and uses the combined 14-bit address ($000x) to access the intended operand.

8.3.5 Short Addressing Mode (SRT)

SRT addressing mode is capable of addressing only the first 32 bytes in the address map, from $0000 to $001F. This addressing mode is available for CLR, LDA, and STA instructions. A system can be optimized by placing the most computation-intensive data in this area of memory.

Because the 5-bit address is embedded in the opcode, only the least significant five bits of the address must be included in the instruction; this saves program space and execution time. During execution, the CPU adds nine high-order 0s to the 5-bit operand address and uses the combined 14-bit address ($000x or $001x) to access the intended operand.

8.3.6 Direct Addressing Mode (DIR)

DIR addressing mode is used to access operands located in direct address space ($0000 through $00FF).

During execution, the CPU adds six high-order 0s to the low byte of the direct address operand that follows the opcode. The CPU uses the combined 14-bit address ($00xx) to access the intended operand.

8.3.7 Extended Addressing Mode (EXT)

In the extended addressing mode, the 14-bit address of the operand is included in the object code in the low-order 14 bits of the next two bytes after the opcode. This addressing mode is only used in JSR and JMP instructions for jump destination address in RS08 MCUs.

8.3.8 Indexed Addressing Mode (IX, Implemented by Pseudo Instructions)

Indexed addressing mode is sometimes called indirect addressing mode because an index register is used as a reference to access the intended operand.

An important feature of indexed addressing mode is that the operand address is computed during execution based on the current contents of the X index register located in $000F of the memory map rather than being a constant address location that was determined during program assembly. This allows writing of a program that accesses different operand locations depending on the results of earlier program instructions (rather than accessing a location that was determined when the program was written).

The index addressing mode supported by the RS08 Family uses the register X located at $000F as an index and D[X] register located at $000E as the indexed data register. By programming the index register X, any location in the direct page can be read/written via the indexed data register D[X].

These pseudo instructions can be used with all instructions supporting direct, short, and tiny addressing modes by using the D[X] as the operand.

8.4 Special Operations

Most of what the CPU does is described by the instruction set, but a few special operations must be considered, such as how the CPU starts at the beginning of an application program after power is first applied. After the program begins running, the current instruction normally determines what the CPU will do next. Two exceptional events can cause the CPU to temporarily suspend normal program execution:

- Reset events force the CPU to start over at the beginning of the application program, which forces execution to start at $3FFD.
- A host development system can cause the CPU to go to active background mode rather than continuing to the next instruction in the application program.

8.4.1 Reset Sequence

Processing begins at the trailing edge of a reset event. The number of things that can cause reset events can vary slightly from one RS08 derivative to another; however, the most common sources are: power-on reset, the external $\overline{\text{RESET}}$ pin, low-voltage reset, COP watchdog timeout, illegal opcode detect, and illegal address access. For more information about how the MCU recognizes reset events and determines the difference between internal and external causes, refer to the Resets and Interrupts chapter.

Reset events force the MCU to immediately stop what it is doing and begin responding to reset. Any instruction that was in process will be aborted immediately without completing any remaining clock cycles. A short sequence of activities is completed to decide whether the source of reset was internal or external and to record the cause of reset. For the remainder of the time, the reset source remains active and the internal clocks are stopped to save power. At the trailing edge of the reset event, the clocks resume and the CPU exits from the reset condition. The program counter is reset to $3FFD and an instruction fetch will be started after the release of reset.

For the device to execute code from the on-chip memory starting from $3FFD after reset, care must be taken to not force the BKDG pin low on the end of reset because this will force the device into active background mode where the CPU will wait for a command from the background communication interface.

8.4.2 Interrupts

The interrupt mechanism in RS08 is not used to interrupt the normal flow of instructions; it is used to wake up the RS08 from wait and stop modes. In run mode, interrupt events must be polled by the CPU. The interrupt feature is not compatible with Freescale's HC05, HC08, or HCS08 Families.

8.4.3 Wait and Stop Mode

Wait and stop modes are entered by executing a WAIT or STOP instruction, respectively. In these modes, the clocks to the CPU are shut down to save power and CPU activity is suspended. The CPU remains in this low-power state until an interrupt or reset event wakes it up. Please refer to the Resets and Interrupts chapter for the effects of wait and stop on other device peripherals.

8.4.4 Active Background Mode

Active background mode refers to the condition in which the CPU has stopped executing user program instructions and is waiting for serial commands from the background debug system. Refer to the Development Support chapter for detailed information on active background mode.

The arithmetic left shift pseudo instruction is also available because its operation is identical to logical shift left.

8.5 Summary Instruction Table

Instruction Set Summary Nomenclature

The nomenclature listed here is used in the instruction descriptions in Table 8-1 through Table 8-2.

Operators

( )	=	Contents of register or memory location shown inside parentheses
←	=	Is loaded with (read: “gets”)
⇔	=	Exchange with
&	=	Boolean AND
|	=	Boolean OR
⊕	=	Boolean exclusive-OR
:	=	Concatenate
+	=	Add

CPU registers

A	=	Accumulator
CCR	=	Condition code register
PC	=	Program counter
PCH	=	Program counter, higher order (most significant) six bits
PCL	=	Program counter, lower order (least significant) eight bits
SPC	=	Shadow program counter
SPCH	=	Shadow program counter, higher order (most significant) six bits
SPCL	=	Shadow program counter, lower order (least significant) eight bits

Memory and addressing

M	=	A memory location or absolute data, depending on addressing mode
rel	=	The relative offset, which is the two’s complement number stored in the last byte of machine code corresponding to a branch instruction
X	=	Pseudo index register, memory location $000F
,X or D[X]	=	Memory location $000E pointing to the memory location defined by the pseudo index register (location $000F)

Condition code register (CCR) bits

Z	=	Zero indicator
C	=	Carry/borrow

CCR activity notation

–	=	Bit not affected
0	=	Bit forced to 0
1	=	Bit forced to 1
↕	=	Bit set or cleared according to results of operation
U	=	Undefined after the operation

Machine coding notation

dd	=	Low-order eight bits of a direct address $0000–$00FF (high byte assumed to be $00)
ii	=	One byte of immediate data
hh	=	High-order 6-bit of 14-bit extended address prefixed with 2-bit of 0
ll	=	Low-order byte of 14-bit extended address
rr	=	Relative offset

Source form

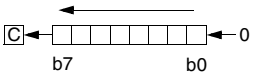
Everything in the source forms columns, *except expressions in italic characters,* is literal information which must appear in the assembly source file exactly as shown. The initial 3- to 5-letter mnemonic is always a literal expression. All commas, pound signs (#), parentheses, and plus signs (+) are literal characters.

n	—	Any label or expression that evaluates to a single integer in the range 0–7.
x	—	Any label or expression that evaluates to a single hexadecimal integer in the range $0–$F.
opr8i	—	Any label or expression that evaluates to an 8-bit immediate value.
opr4a	—	Any label or expression that evaluates to a Tiny address (4-bit value). The instruction treats this 4-bit value as the low order four bits of an address in the 16-Kbyte address space ($0000–$000F). This 4-bit value is embedded in the low order four bits in the opcode.
opr5a	—	Any label or expression that evaluates to a Short address (5-bit value). The instruction treats this 5-bit value as the low order five bits of an address in the 16-Kbyte address space ($0000–$001F). This 5-bit value is embedded in the low order 5 bits in the opcode.
opr8a	—	Any label or expression that evaluates to an 8-bit value. The instruction treats this 8-bit value as the low order eight bits of an address in the 16-Kbyte address space ($0000–$00FF).
opr16a	—	Any label or expression that evaluates to a 14-bit value. On the RS08 core, the upper two bits are always 0s. The instruction treats this value as an address in the 16-Kbyte address space.
rel	—	Any label or expression that refers to an address that is within –128 to +127 locations from the next address after the last byte of object code for the current instruction. The assembler will calculate the 8-bit signed offset and include it in the object code for this instruction.

Address modes

INH	=	Inherent (no operands)
IMD	=	Immediate to Direct (in MOV instruction)
IMM	=	Immediate
DD	=	Direct to Direct (in MOV instruction)
DIR	=	Direct
SRT	=	Short
TNY	=	Tiny
EXT	=	Extended
REL	=	8-bit relative offset

Table 8-1. Instruction Set Summary (Sheet 1 of 6)

Source Form	Description	Operation	Effect on CCR		Address Mode	Opcode	Operand	Cycles
			Z	C				
ADC #*opr8i* ADC *opr8a* ADC ,X (1) ADC X	Add with Carry	A ← (A) + (M) + (C) A ← (A) + (X) + (C)	↕	↕	IMM DIR IX DIR	A9 B9 B9 B9	ii dd 0E 0F	2 3 3 3
ADD #*opr8i* ADD *opr8a* ADD *opr4a* ADD ,X (1) ADD X	Add without Carry	A ← (A) + (M)	↕	↕	IMM DIR TNY IX DIR	AB BB 6*x* 6E 6F	ii dd	2 3 3 3 3
AND #*opr8i* AND *opr8a* AND ,X (1) AND X	Logical AND	A ← (A) & (M) A ← (A) & (X)	↕	—	IMM DIR IX DIR	A4 B4 B4 B4	ii dd 0E 0F	2 3 3 3
ASLA(1)	Arithmetic Shift Left		↕	↕	INH	48		1
BCC *rel*	Branch if Carry Bit Clear	PC ← (PC) + $0002 + *rel*, if (C) = 0	—	—	REL	34	rr	3
BCLR *n*,*opr8a* BCLR *n*,D[X] BCLR *n*,X	Clear Bit n in Memory	Mn ← 0	—	—	DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7) IX (b0) IX (b1) IX (b2) IX (b3) IX (b4) IX (b5) IX (b6) IX (b7) DIR (b0) DIR (b1) DIR (b2) DIR (b3) DIR (b4) DIR (b5) DIR (b6) DIR (b7)	11 13 15 17 19 1B 1D 1F 11 13 15 17 19 1B 1D 1F 11 13 15 17 19 1B 1D 1F	dd dd dd dd dd dd dd dd 0E 0E 0E 0E 0E 0E 0E 0E 0F 0F 0F 0F 0F 0F 0F 0F	5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
BCS *rel*	Branch if Carry Bit Set (Same as BLO)	PC ← (PC) + $0002 + *rel*, if (C) = 1	—	—	REL	35	rr	3
BEQ *rel*	Branch if Equal	PC ← (PC) + $0002 + *rel*, if (Z) = 1	—	—	REL	37	rr	3
BGND	Background	Enter Background Debug Mode	—	—	INH	BF		5+

1. This is a pseudo instruction supported by the normal RS08 instruction set.

2. This instruction is different from that of the HC08 and HCS08 in that the RS08 does not auto-increment the index register.

Table 8-1. Instruction Set Summary (Sheet 2 of 6)

Source Form	Description	Operation	Effect on CCR		Address Mode	Opcode	Operand	Cycles
			Z	C				
BHS *rel* (1)	Branch if Higher or Same (Same as BCC)	PC ← (PC) + $0002 + *rel*, if (C) = 0	—	—	REL	34	rr	3
BLO *rel* (1)	Branch if Lower (Same as BCS)	PC ← (PC) + $0002 + *rel*, if (C) = 1	—	—	REL	35	rr	3
BNE *rel*	Branch if Not Equal	PC ← (PC) + $0002 + *rel*, if (Z) = 0	—	—	REL	36	rr	3
BRA *rel*	Branch Always	PC ← (PC) + $0002 + *rel*	—	—	REL	30	rr	3
BRN *rel* (1)	Branch Never	PC ← (PC) + $0002	—	—	REL	30	00	3
BRCLR *n*,*opr8a*,*rel*	Branch if Bit *n* in Memory Clear	PC ← (PC) + $0003 + *rel*, if (Mn) = 0	—	↕	DIR (b0)	01	dd rr	5
					DIR (b1)	03	dd rr	5
					DIR (b2)	05	dd rr	5
					DIR (b3)	07	dd rr	5
					DIR (b4)	09	dd rr	5
					DIR (b5)	0B	dd rr	5
					DIR (b6)	0D	dd rr	5
					DIR (b7)	0F	dd rr	5
BRCLR *n*,D[X],*rel*					IX (b0)	01	0E rr	5
					IX (b1)	03	0E rr	5
					IX (b2)	05	0E rr	5
					IX (b3)	07	0E rr	5
					IX (b4)	09	0E rr	5
					IX (b5)	0B	0E rr	5
					IX (b6)	0D	0E rr	5
					IX (b7)	0F	0E rr	5
BRCLR *n*,X,*rel*					DIR (b0)	01	0F rr	5
					DIR (b1)	03	0F rr	5
					DIR (b2)	05	0F rr	5
					DIR (b3)	07	0F rr	5
					DIR (b4)	09	0F rr	5
					DIR (b5)	0B	0F rr	5
					DIR (b6)	0D	0F rr	5
					DIR (b7)	0F	0F rr	5

1. This is a pseudo instruction supported by the normal RS08 instruction set.
2. This instruction is different from that of the HC08 and HCS08 in that the RS08 does not auto-increment the index register.

Table 8-1. Instruction Set Summary (Sheet 3 of 6)

Source Form	Description	Operation	Effect on CCR		Address Mode	Opcode	Operand	Cycles
			Z	C				
BRSET *n*,*opr8a*,*rel*	Branch if Bit *n* in Memory Set	PC ← (PC) + $0003 + *rel*, if (Mn) = 1	—	↕	DIR (b0)	00	dd rr	5
					DIR (b1)	02	dd rr	5
					DIR (b2)	04	dd rr	5
					DIR (b3)	06	dd rr	5
					DIR (b4)	08	dd rr	5
					DIR (b5)	0A	dd rr	5
					DIR (b6)	0C	dd rr	5
					DIR (b7)	0E	dd rr	5
BRSET *n*,D[X],*rel*					IX (b0)	00	0E rr	5
					IX (b1)	02	0E rr	5
					IX (b2)	04	0E rr	5
					IX (b3)	06	0E rr	5
					IX (b4)	08	0E rr	5
					IX (b5)	0A	0E rr	5
					IX (b6)	0C	0E rr	5
					IX (b7)	0E	0E rr	5
BRSET *n*,X,*rel*					DIR (b0)	00	0F rr	5
					DIR (b1)	02	0F rr	5
					DIR (b2)	04	0F rr	5
					DIR (b3)	06	0F rr	5
					DIR (b4)	08	0F rr	5
					DIR (b5)	0A	0F rr	5
					DIR (b6)	0C	0F rr	5
					DIR (b7)	0E	0F rr	5
BSET *n*,*opr8a*	Set Bit *n* in Memory	Mn ← 1	—	—	DIR (b0)	10	dd	5
					DIR (b1)	12	dd	5
					DIR (b2)	14	dd	5
					DIR (b3)	16	dd	5
					DIR (b4)	18	dd	5
					DIR (b5)	1A	dd	5
					DIR (b6)	1C	dd	5
					DIR (b7)	1E	dd	5
BSET *n*,D[X]					IX (b0)	10	0E	5
					IX (b1)	12	0E	5
					IX (b2)	14	0E	5
					IX (b3)	16	0E	5
					IX (b4)	18	0E	5
					IX (b5)	1A	0E	5
					IX (b6)	1C	0E	5
					IX (b7)	1E	0E	5
BSET *n*,X					DIR (b0)	10	0F	5
					DIR (b1)	12	0F	5
					DIR (b2)	14	0F	5
					DIR (b3)	16	0F	5
					DIR (b4)	18	0F	5
					DIR (b5)	1A	0F	5
					DIR (b6)	1C	0F	5
					DIR (b7)	1E	0F	5

1. This is a pseudo instruction supported by the normal RS08 instruction set.

2. This instruction is different from that of the HC08 and HCS08 in that the RS08 does not auto-increment the index register.

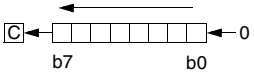
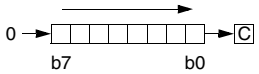
Table 8-1. Instruction Set Summary (Sheet 4 of 6)

Source Form	Description	Operation	Effect on CCR		Address Mode	Opcode	Operand	Cycles
			Z	C				
BSR *rel*	Branch Subroutine	PC ← (PC) + 2 Push PC to shadow PC PC ← (PC) + *rel*	—	—	REL	AD	rr	3
CBEQA #*opr8i*,*rel* CBEQ *opr8a*,*rel* CBEQ ,X,*rel* (1),(2) CBEQ X,*rel* (1)	Compare and Branch if Equal	PC ← (PC) + $0003 + *rel*, if (A) – (M) = $00 PC ← (PC) + $0003 + *rel*, if (A) – (M) = $00 PC ← (PC) + $0003 + *rel*, if (A) – (X) = $00	—	—	IMM DIR IX DIR	41 31 31 31	ii rr dd rr 0E rr 0F rr	4 5 5 5
CLC	Clear Carry Bit	C ← 0	—	0	INH	38		1
CLR *opr8a* CLR *opr5a* CLR ,X (1) CLRA CLRX (1)	Clear	M ← $00 A ← $00 X ← $00	1	—	DIR SRT IX INH INH	3F 8*x* / 9*x* 8E 4F 8F	dd	3 2 2 1 2
CMP #*opr8i* CMP *opr8a* CMP ,X (1) CMP X (1)	Compare Accumulator with Memory	(A) – (M) (A) – (X)	↕	↕	IMM DIR IX INH	A1 B1 B1 B1	ii dd 0E 0F	2 3 3 3
COMA	Complement (One's Complement)	A ← ($\overline{A}$)	↕	1	INH	43		1
DBNZ *opr8a*,*rel* DBNZ ,X,rel (1) DBNZA *rel* DBNZX rel (1)	Decrement and Branch if Not Zero	A ← (A) – $01 or M ← (M) - $01 PC ← (PC) + $0003 + *rel* if (result) ≠ 0 for DBNZ direct PC ← (PC) + $0002 + *rel* if (result) ≠ 0 for DBNZA X ← (X) – $01 PC ← (PC) + $0003 + *rel* if (result) ≠ 0	—	—	DIR IX INH INH	3B 3B 4B 3B	dd rr 0E rr rr 0F rr	7 7 4 7
DEC *opr8a* DEC *opr4a* DEC ,X (1) DECA DEC X	Decrement	M ← (M) – $01 A ← (A) – $01 X ← (X) – $01	↕	—	DIR TNY IX INH DIR	3A 5*x* 5E 4A 5F	dd	5 4 4 1 4
EOR #*opr8i* EOR *opr8a* EOR ,X (1) EOR X	Exclusive OR Memory with Accumulator	A ← (A ⊕ M) A ← (A ⊕ X)	↕	—	IMM DIR IX DIR	A8 B8 B8 B8	ii dd 0E 0F	2 3 3 3
INC *opr8a* INC *opr4a* INC ,X (1) INCA INCX (1)	Increment	M ← (M) + $01 A ← (A) + $01 X ← (X) + $01	↕	—	DIR TNY IX INH INH	3C 2*x* 2E 4C 2F	dd	5 4 4 1 4
JMP *opr16a*	Jump	PC ← Effective Address	—	—	EXT	BC	hh ll	4
JSR *opr16a*	Jump to Subroutine	PC ← (PC) + 3 Push PC to shadow PC PC ← Effective Address	—	—	EXT	BD	hh ll	4
LDA #*opr8i* LDA *opr8a* LDA *opr5a* LDA ,X (1)	Load Accumulator from Memory	A ← (M)	↕	—	IMM DIR SRT IX	A6 B6 C*x*/D*x* CE	ii dd	2 3 3 3

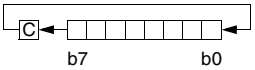
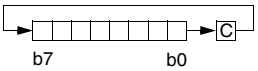
1. This is a pseudo instruction supported by the normal RS08 instruction set.

2. This instruction is different from that of the HC08 and HCS08 in that the RS08 does not auto-increment the index register.

Table 8-1. Instruction Set Summary (Sheet 5 of 6)

Source Form	Description	Operation	Effect on CCR		Address Mode	Opcode	Operand	Cycles
			Z	C				
LDX #*opr8i* (1) LDX *opr8a* (1) LDX ,X (1)	Load Index Register from Memory	$0F ← (M)	↕	—	IMD DIR IX	3E 4E 4E	ii 0F dd 0F 0E 0E	4 5 5
LSLA	Logical Shift Left		↕	↕	INH	48		1
LSRA	Logical Shift Right		↕	↕	INH	44		1

MOV *oprr8a,oprr8a*
MOV #*opr8i,oprr8a*
MOV D[X],*oprr8a*
MOV *oprr8a*,D[X]
MOV #*opr8i*,D[X]
	Move	$(M)_{destination} ← (M)_{source}$	↕	—	DD
					IMD
					IX/DIR
					DIR/IX
					IMM/IX
						4E
						3E
						4E
						4E
						3E
							dd dd
							ii dd
							0E dd
							dd 0E
							ii 0E
								5
								4
								5
								5
								4

NOP	No Operation	None	—	—	INH	AC		1
ORA #*opr8i* ORA *opr8a* ORA ,X (1) ORA X	Inclusive OR Accumulator and Memory	A ← (A) | (M) A ← (A) | (X)	↕	—	IMM DIR IX DIR	AA BA BA BA	ii dd 0E 0F	2 3 3 3
ROLA	Rotate Left through Carry		↕	↕	INH	49		1
RORA	Rotate Right through Carry		↕	↕	INH	46		1
RTS	Return from Subroutine	Pull PC from shadow PC	—	—	INH	BE		3
SBC #*opr8i* SBC *opr8a* SBC ,X (1) SBC X	Subtract with Carry	A ← (A) – (M) – (C) A ← (A) – (X) – (C)	↕	↕	IMM DIR IX DIR	A2 B2 B2 B2	ii dd 0E 0F	2 3 3 3
SEC	Set Carry Bit	C ← 1	—	1	INH	39		1
SHA	Swap Shadow PC High with A	A ⇔ SPCH	—	—	INH	45		1
SLA	Swap Shadow PC Low with A	A ⇔ SPCL	—	—	INH	42		1
STA *opr8a* STA *opr5a* STA ,X (1) STA X	Store Accumulator in Memory	M ← (A)	↕	—	DIR SRT IX SRT	B7 E*x* / F*x* EE EF	dd	3 2 2 2
STX *opr8a* (1)	Store Index Register in Memory	M ← (X)	↕	—	DIR	4E	0F dd	5
STOP	Put MCU into stop mode		—	—	INH	AE		2+

1. This is a pseudo instruction supported by the normal RS08 instruction set.

2. This instruction is different from that of the HC08 and HCS08 in that the RS08 does not auto-increment the index register.

Table 8-1. Instruction Set Summary (Sheet 6 of 6)

Source Form	Description	Operation	Effect on CCR		Address Mode	Opcode	Operand	Cycles
			Z	C				
SUB #*opr8i* SUB *opr8a* SUB *opr4a* SUB ,X [1] SUB X	Subtract	A ← (A) – (M) A ← (A) – (X)	↕	↕	IMM DIR TNY IX DIR	A0 B0 7*x* 7E 7F	ii dd	2 3 3 3 3
TAX[1]	Transfer A to X	X ← (A)	↕	—	INH	EF		2
TST opr8a [1] TSTA [1] TST ,X [1] TSTX [1]	Test for Zero	(M) – $00 (A) – $00 (X) – $00	↕	—	DD INH IX INH	4E AA 4E 4E	dd dd 00 0E 0E 0F 0F	5 2 5 5
TXA[1]	Transfer X to A	A ← (X)	↕	—	INH	CF		3
WAIT	Put MCU into WAIT mode		—	—	INH	AF		2+

1. This is a pseudo instruction supported by the normal RS08 instruction set.

2. This instruction is different from that of the HC08 and HCS08 in that the RS08 does not auto-increment the index register.

Table 8-2. Opcode Map

HIGH / LOW	DIR 0	DIR 1	TNY 2	DIR/REL 3	INH 4	TNY 5	TNY 6	TNY 7	SRT 8	SRT 9	IMM/INH A	DIR/EXT B	SRT C	SRT D	SRT E	SRT F
0	5 BRSET0 3 DIR	5 BSET0 2 DIR	4 INC 1 TNY	3 BRA 2 REL		4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2 SUB 2 IMM	3 SUB 2 DIR	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
1	5 BRCLR0 3 DIR	5 BCLR0 2 DIR	4 INC 1 TNY	5 CBEQ 3 DIR	4 CBEQA 3 IMM	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2 CMP 2 IMM	3 CMP 2 DIR	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
2	5 BRSET1 3 DIR	5 BSET1 2 DIR	4 INC 1 TNY		1 SLA 1 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2 SBC 2 IMM	3 SBC 2 DIR	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
3	5 BRCLR1 3 DIR	5 BCLR1 2 DIR	4 INC 1 TNY		1 COMA 1 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT			3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
4	5 BRSET2 3 DIR	5 BSET2 2 DIR	4 INC 1 TNY	3 BCC 2 REL	1 LSRA 1 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2 AND 2 IMM	3 AND 2 DIR	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
5	5 BRCLR2 3 DIR	5 BCLR2 2 DIR	4 INC 1 TNY	3 BCS 2 REL	1 SHA 1 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT			3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
6	5 BRSET3 3 DIR	5 BSET3 2 DIR	4 INC 1 TNY	3 BNE 2 REL	1 RORA 1 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2 LDA 2 IMM	3 LDA 2 DIR	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
7	5 BRCLR3 3 DIR	5 BCLR3 2 DIR	4 INC 1 TNY	3 BEQ 2 REL		4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT		3 STA 2 DIR	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
8	5 BRSET4 3 DIR	5 BSET4 2 DIR	4 INC 1 TNY	1 CLC 1 INH	1 LSLA 1 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2 EOR 2 IMM	3 EOR 2 DIR	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
9	5 BRCLR4 3 DIR	5 BCLR4 2 DIR	4 INC 1 TNY	1 SEC 1 INH	1 ROLA 1 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2 ADC 2 IMM	3 ADC 2 DIR	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
A	5 BRSET5 3 DIR	5 BSET5 2 DIR	4 INC 1 TNY	5 DEC 2 DIR	1 DECA 1 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2 ORA 2 IMM	3 ORA 2 DIR	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
B	5 BRCLR5 3 DIR	5 BCLR5 2 DIR	4 INC 1 TNY	6 DBNZ 3 DIR	4 DBNZA 2 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2 ADD 2 IMM	3 ADD 2 DIR	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
C	5 BRSET6 3 DIR	5 BSET6 2 DIR	4 INC 1 TNY	5 INC 2 DIR	1 INCA 1 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	1 NOP 1 INH	4 JMP 3 EXT	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
D	5 BRCLR6 3 DIR	5 BCLR6 2 DIR	4 INC 1 TNY			4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	3 BSR 2 REL	4 JSR 3 EXT	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
E	5 BRSET7 3 DIR	5 BSET7 2 DIR	4 INC 1 TNY	4 MOV 3 IMD	5 MOV 3 DD	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2+ STOP 1 INH	3 RTS 1 INH	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT
F	5 BRCLR7 3 DIR	5 BCLR7 2 DIR	4 INC 1 TNY	3 CLR 2 DIR	1 CLRA 1 INH	4 DEC 1 TNY	3 ADD 1 TNY	3 SUB 1 TNY	2 CLR 1 SRT	2 CLR 1 SRT	2+ WAIT 1 INH	5+ BGND 1 INH	3 LDA 1 SRT	3 LDA 1 SRT	2 STA 1 SRT	2 STA 1 SRT

INH	**Inherent**	**REL**	**Relative**
IMM	**Immediate**	**SRT**	**Short**
DIR	**Direct**	**TNY**	**Tiny**
EXT	**Extended**		
DD	**Direct-Direct**	**IMD**	**Immediate-Direct**

High Byte of Opcode in Hexadecimal **B**

Gray box is decoded as illegal instruction

Low Byte of Opcode in Hexadecimal **0**

3 SUB 2 DIR

RS08 Cycles
Opcode Mnemonic
Number of Bytes / Addressing Mode

Chapter 9
Internal Clock Source (RS08ICSV1)

9.1 Introduction

The internal clock source (ICS) module provides clock source choices for the MCU. The module contains a frequency-locked loop (FLL) as a clock source that is controllable by an internal reference clock. The module can provide this FLL clock or the internal reference clock as a source for the MCU system clock, ICSOUT.

Whichever clock source is chosen, ICSOUT is passed through a bus clock divider (BDIV), which allows a lower final output clock frequency to be derived. ICSOUT is two times the bus frequency.

Figure 9-1 shows the MC9RS08KA2 Series block diagram with the ICS highlighted.

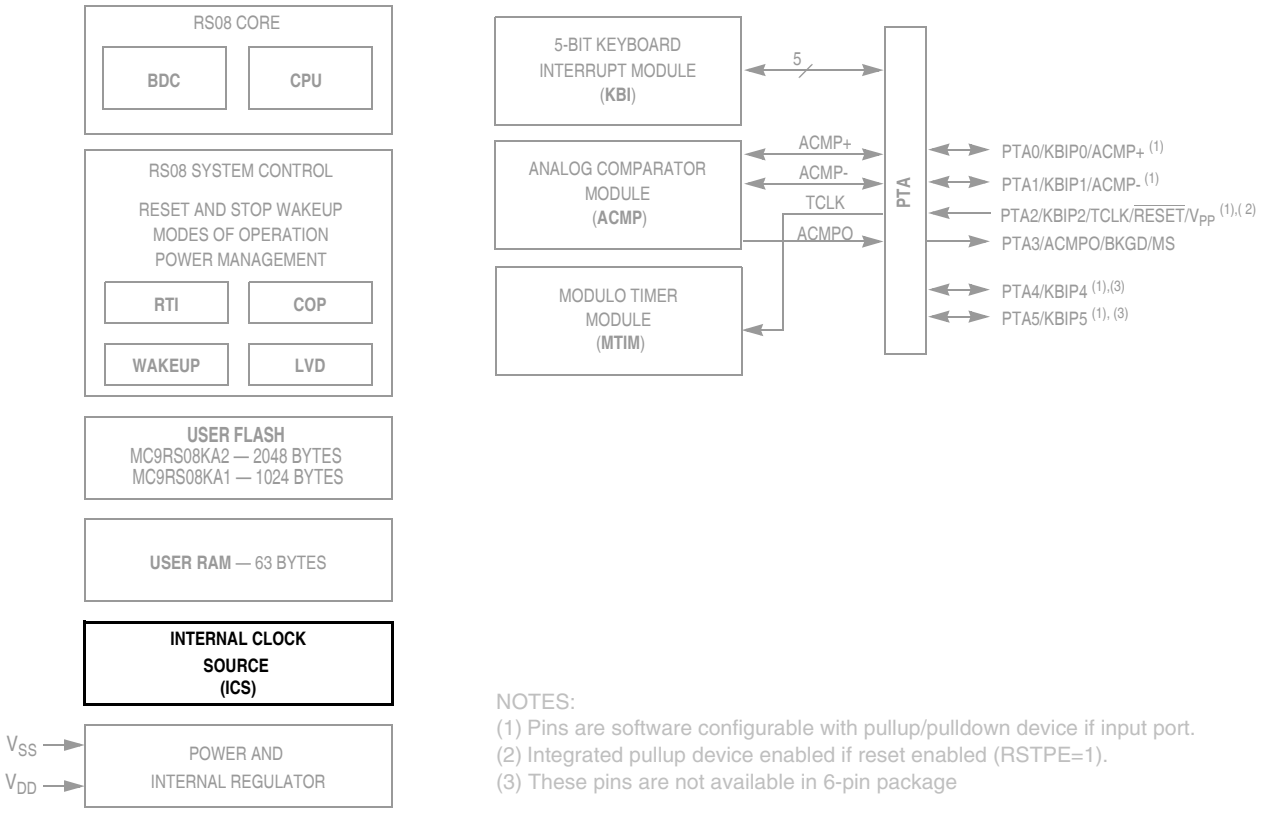


Figure 9-1. MC9RS08KA2 Series Block Diagram Highlighting ICS Block

9.1.1 Features

Key features of the ICS module are:

- Frequency-locked loop (FLL) is trimmable for accuracy
  - 0.2% resolution using internal 32 kHz reference
  - 2% deviation over voltage and temperature using internal 32 kHz reference
  - DCO output is 512 times internal reference frequency
- Internal reference clock has 9 trim bits available
- Internal reference clock can be selected as the clock source for the MCU
- Whichever clock is selected as the source can be divided down
  - 2 bit select for clock divider is provided (allowable dividers are: 1, 2, 4, and 8)
- FLL engaged internal mode is automatically selected out of reset

9.1.2 Modes of Operation

There are four modes of operation for the ICS: FEI, FBI, FBILP, and stop.

9.1.2.1 FLL Engaged Internal (FEI)

In FLL engaged internal mode, which is the default mode, the ICS supplies a clock derived from the FLL which is controlled by the internal reference clock.

9.1.2.2 FLL Bypassed Internal (FBI)

In FLL bypassed internal mode, the FLL is enabled and controlled by the internal reference clock, but is bypassed. The ICS supplies a clock derived from the internal reference clock.

9.1.2.3 FLL Bypassed Internal Low Power (FBILP)

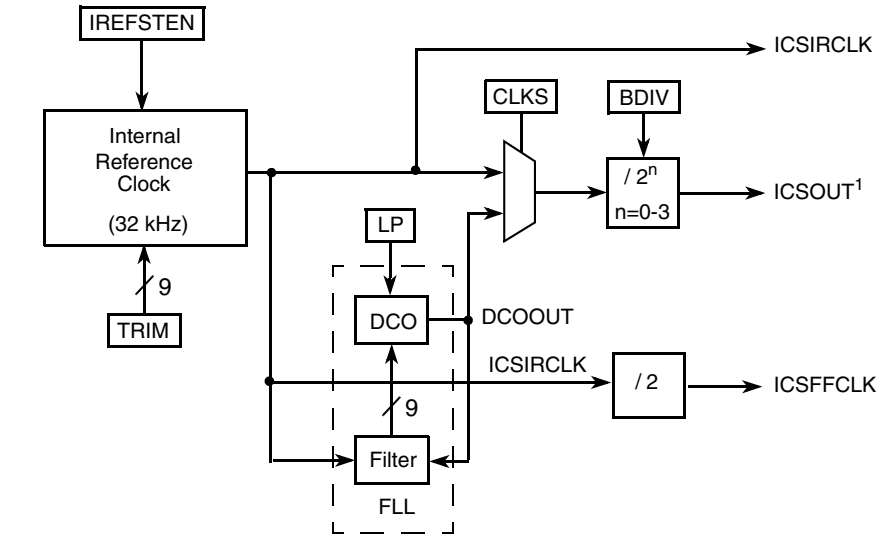
In FLL bypassed internal low power mode, the FLL is disabled and bypassed, and the ICS supplies a clock derived from the internal reference clock.

9.1.2.4 Stop (STOP)

In stop mode, the FLL is disabled and the internal reference clocks can be selected to be enabled or disabled. The ICS does not provide an MCU clock source.

9.1.3 Block Diagram

Figure 9-2 shows the ICS block diagram.



[1] ICSOUT is two times the bus frequency

Figure 9-2. Internal Clock Source (ICS) Block Diagram

9.2 External Signal Description

No ICS signal connects off chip.

9.3 Register Definition

Table 9-1 is a summary of ICS registers.

Table 9-1. ICS Register Summary

Name		7	6	5	4	3	2	1	0
ICSC1	R	0	CLKS	0	0	0	0	0	IREFSTEN
	W								
ICSC2	R	BDIV		0	0	LP	0	0	0
	W								
ICSTRM	R	TRIM							
	W								
ICSSC	R	0	0	0	0	0	CLKST	0	FTRIM
	W								

9.3.1 ICS Control Register 1 (ICSC1)

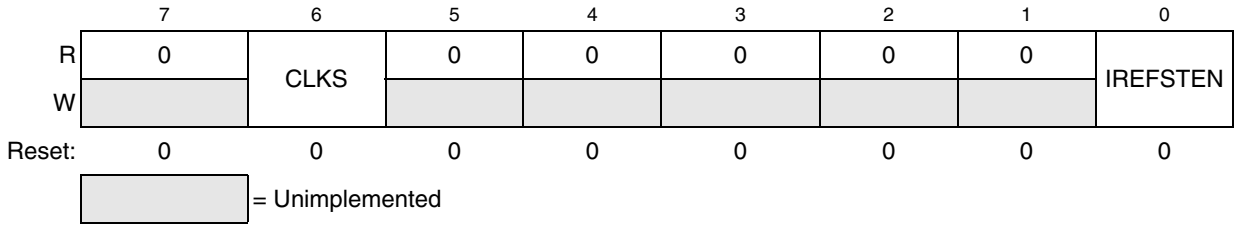


Figure 9-3. ICS Control Register 1 (ICSC1)

Table 9-2. ICSC1 Field Descriptions

Field	Description
6 CLKS	**Clock Source Select** — Selects the clock source that controls the bus frequency. The actual bus frequency depends on the value of the BDIV bits. 0 Output of FLL is selected 1 Internal reference clock is selected
0 IREFSTEN	**Internal Reference Stop Enable** — Controls whether the internal reference clock remains enabled when the ICS enters stop mode. 1 Internal reference clock remains enabled in stop 0 Internal reference clock is disabled in stop

9.3.2 ICS Control Register 2 (ICSC2)

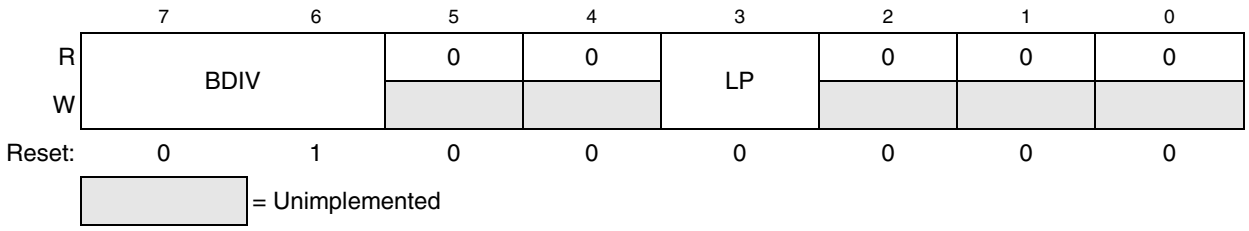


Figure 9-4. ICS Control Register 2 (ICSC2)

Table 9-3. ICSC2 Field Descriptions

Field	Description
7:6 BDIV	**Bus Frequency Divider** — Selects the amount to divide down the clock source selected by the CLKS bit. This controls the bus frequency. 00 Encoding 0 — Divides selected clock by 1 01 Encoding 1 — Divides selected clock by 2 (reset default) 10 Encoding 2 — Divides selected clock by 4 11 Encoding 3 — Divides selected clock by 8
3 LP	**Low Power Select** — Controls whether the FLL is disabled in FLL bypassed modes. 1 FLL is disabled in bypass modes 0 FLL is not disabled in bypass mode

9.3.3 ICS Trim Register (ICSTRM)

	7	6	5	4	3	2	1	0
R W	TRIM							
POR:	1	0	0	0	0	0	0	0
Reset:	U	U	U	U	U	U	U	U

Figure 9-5. ICS Trim Register (ICSTRM)

Table 9-4. ICSTRM Field Descriptions

Field	Description
7:0 TRIM	**ICS Trim Setting** — The TRIM bits control the internal reference clock frequency by controlling the internal reference clock period. The bits' effect are binary weighted (i.e., bit 1 will adjust twice as much as bit 0). Increasing the binary value in TRIM will increase the period, and decreasing the value will decrease the period. An additional fine trim bit is available in ICSSC as the FTRIM bit.

9.3.4 ICS Status and Control (ICSSC)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	CLKST	0	FTRIM
W								
POR:	0	0	0	0	0	0	0	0
Reset:	0	0	0	0	0	0	0	U

= Unimplemented

Figure 9-6. ICS Status and Control Register (ICSSC)

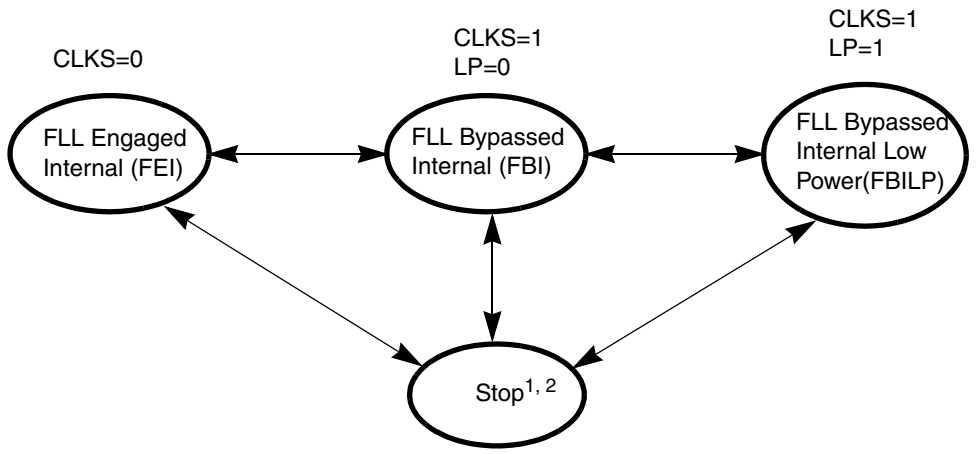
Table 9-5. ICSSC Field Descriptions

Field	Description
2 CLKST	**Clock Mode Status** — The CLKST read-only bit indicate the current clock mode. The CLKST bit does not update immediately after a write to the CLKS bit due to internal synchronization between clock domains. 0 Output of FLL is selected 1 Internal reference clock is selected
0 FTRIM	**ICS Fine Trim** — The FTRIM bit controls the smallest adjustment of the internal reference clock frequency. Setting FTRIM will increase the period and clearing FTRIM will decrease the period by the smallest amount possible.

9.4 Functional Description

9.4.1 Operational Modes

The states of the ICS are shown as a state diagram and are described in this section. The arrows indicate the allowed movements between the states.



[1] ICS enters its Stop state when MCU enters stop, FLL is always disabled. ICS returns to the state that was active before MCU entered stop, unless a reset occurs while in stop.
[2] If IREFSTEN is set when MCU enters stop, the ICSIRCLK remains running.

Figure 9-7. Clock Switching Modes

9.4.1.1 FLL Engaged Internal (FEI)

FLL engaged internal (FEI) is the default mode of operation out of any reset and is entered when CLKS is written to 0.

In FLL engaged internal mode, the ICSOUT clock is derived from the FLL clock, which is controlled by the internal reference clock. The FLL loop will lock the frequency to 512 times the filter frequency.

9.4.1.2 FLL Bypassed Internal (FBI)

The FLL bypassed internal (FBI) mode is entered when CLKS is written to 1 and LP bit is a 0.

In FLL bypassed internal mode, the ICSOUT clock is derived from the internal reference clock. The FLL clock is controlled by the internal reference clock, and the FLL loop will lock the FLL frequency to 512 times the filter frequency.

9.4.1.3 FLL Bypassed Internal Low Power (FBILP)

The FLL bypassed internal low power (FBILP) mode is entered when CLKS is written to 1 and LP = 1.

In FLL bypassed internal low power mode, the ICSOUT clock is derived from the internal reference clock and the FLL is disabled.

9.4.1.4 Stop

ICS stop mode is entered whenever the MCU enters stop. In this mode, all ICS clocks are stopped except ICSIRCLK which will remaining running if IREFSTEN is written to a 1.

When the MCU is interrupted from stop, the ICS will go back to the operating mode that was running when the MCU entered stop. If the internal reference was not running in stop (IREFSTEN = 0), the ICS will take some time, t_{ir_wu}, for the internal reference to wakeup. If the internal reference was already running in stop (IREFSTEN = 1), entering into FEI will take some time, t_{fll_wu}, for the FLL to return its previous acquired frequency.

9.4.2 Mode Switching

When changing from FBILP to either FEI or FBI, or anytime the trim value is written, the user should wait the FLL acquisition time, $t_{acquire}$, before FLL will be guaranteed to be at desired frequency.

9.4.3 Bus Frequency Divider

The BDIV bits can be changed at anytime and the actual switch to the new frequency will occur immediately.

9.4.4 Low Power Bit Usage

The low power bit (LP) is provided to allow the FLL to be disabled and thus conserve power when it is not being used. However, in some applications it may be desirable to enable the FLL and allow it to lock for maximum accuracy before switching to an FLL engaged mode. The FLL is disabled in bypass mode when LP = 1.

9.4.5 Internal Reference Clock

The ICSIRCLK frequency can be re-targeted by trimming the period of the internal reference clock. This can be done by writing a new value to the TRIM bits in the ICSTRM register. Writing a larger value will slow down the ICSIRCLK frequency, and writing a smaller value to the ICSTRM register will speed up the ICSIRCLK frequency. The TRIM bits will affect the ICSOUT frequency if the ICS is in FLL engaged internal (FEI), FLL bypassed internal (FBI), or FLL bypassed internal low power (FBILP) mode. The TRIM and FTRIM values will not be affected by a reset. For the ICS to run in stop, the LVDE and LVDSE bits in the SPMSC1 must both be set before entering stop.

Until ICSIRCLK is trimmed, ICSOUT frequencies may exceed the maximum chip-level frequency and violate the chip-level clock timing specifications (see the Device Overview chapter). The BDIV is reset to a divide by 2 to prevent the bus frequency from exceeding the maximum. The user should trim the device to an allowable frequency before changing BDIV to a divide by 1 operation.

9.4.6 Fixed Frequency Clock

The ICS provides the ICSFFCLK output which can be used as an additional clock source to a peripheral such as a timer, when the ICS is in FEI. ICSFFCLK is not a valid clock source for a peripheral when in either FBI or FBILP modes. ICSFFCLK is ICSRCLK divided by two.

Chapter 10
Analog Comparator (RS08ACMPV1)

10.1 Introduction

The analog comparator module (ACMP) provides a circuit for comparing two analog input voltages or for comparing one analog input voltage to an internal reference voltage. The comparator circuit is designed to operate across the full range of the supply voltage (rail to rail operation).

Figure 10-1 shows the MC9RS08KA2 Series block diagram with the ACMP highlighted.

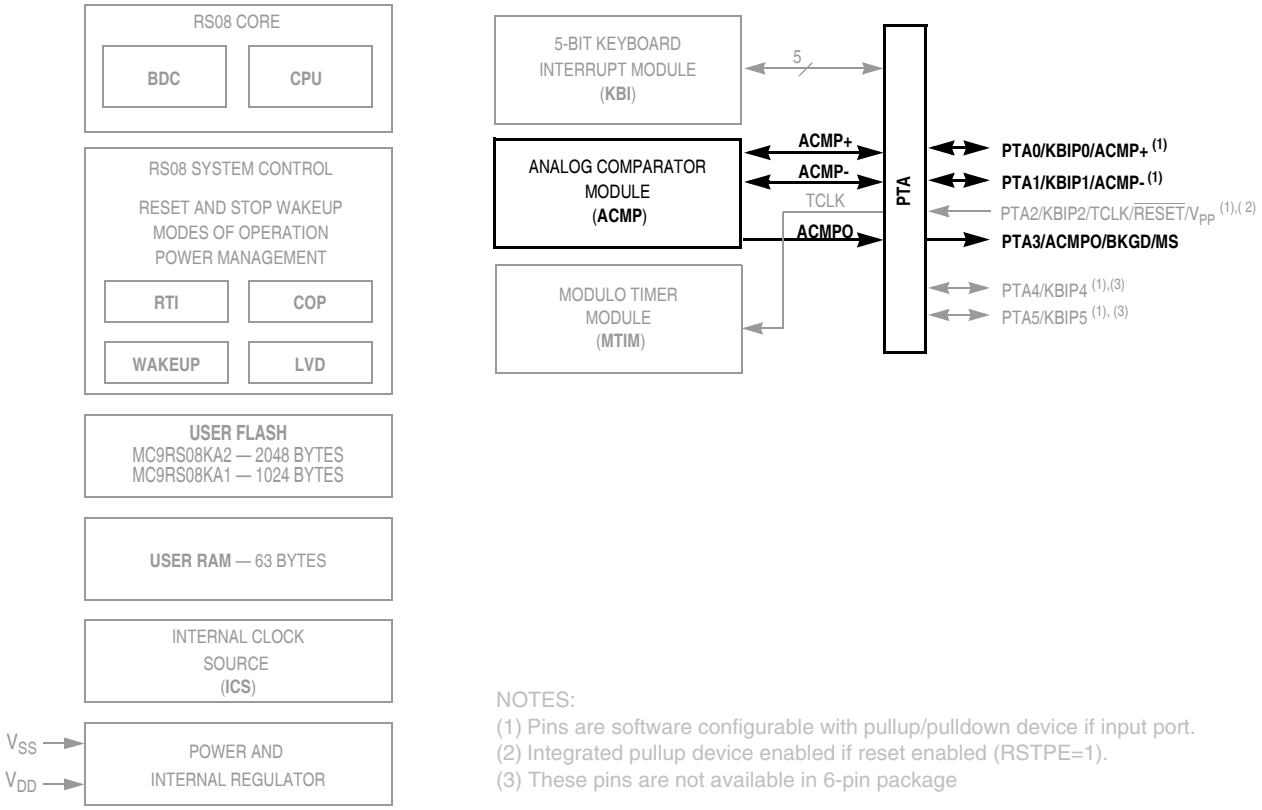


Figure 10-1. MC9RS08KA2 Series Block Diagram Highlighting ACMP Block and Pins

10.1.1 Features

The ACMP has the following features:

- Full rail-to-rail supply operation
- Less than 40 mV of input offset
- Less than 15 mV of hysteresis
- Selectable interrupt on rising edge, falling edge, or either rising or falling edges of comparator output
- Option to compare to fixed internal bandgap reference voltage
- Option to allow comparator output to be visible on a pin, ACMPO
- Remains operational in stop mode

10.1.2 Modes of Operation

This section defines the ACMP operation in wait, stop, and background debug modes.

10.1.2.1 Operation in Wait Mode

The ACMP continues to operate in wait mode if enabled before executing the WAIT instruction. Therefore, the ACMP can be used to bring the MCU out of wait mode if the ACMP interrupt is enabled (ACIE = 1). For lowest possible current consumption, the ACMP should be disabled by software if not required as an interrupt source during wait mode.

10.1.2.2 Operation in Stop Mode

The ACMP continues to operate in stop mode if enabled and compare operation remains active. If ACOPE is enabled, comparator output operates as in the normal operating mode and comparator output is placed onto the external pin. The MCU is brought out of stop when a compare event occurs and ACIE is enabled; ACF flag sets accordingly.

If stop is exited with a reset, the ACMP will be put into its reset state.

10.1.2.3 Operation in Active Background Mode

When the MCU is in active background mode, the ACMP will continue to operate normally.

10.1.3 Block Diagram

The block diagram for the analog comparator module is shown in Figure 10-2.

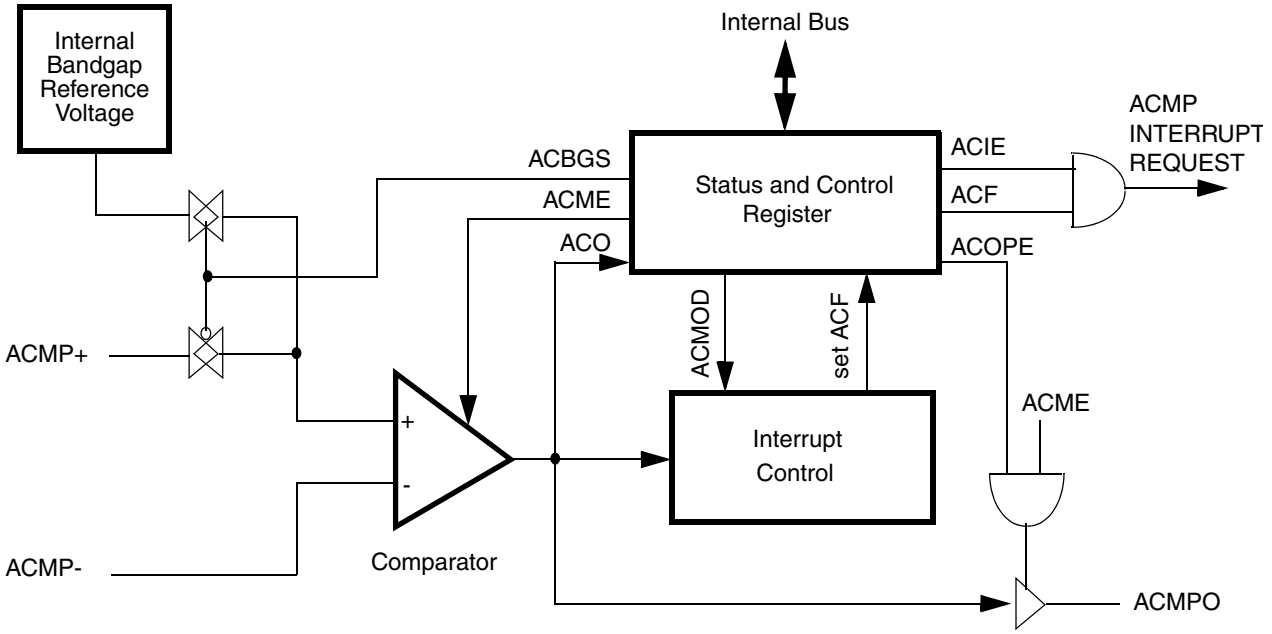


Figure 10-2. Analog Comparator (ACMP) Block Diagram

10.2 External Signal Description

The ACMP has two analog input pins, ACMP+ and ACMP–, and one digital output pin, ACMPO. Each of the input pins can accept an input voltage that varies across the full operating voltage range of the MCU. As shown in Figure 10-2, the ACMP– pin is connected to the inverting input of the comparator, and the ACMP+ pin is connected to the non-inverting input of the comparator if ACBGS=0. As shown in Figure 10-2, the ACMPO pin can be enabled to drive an external pin.

The signal properties of ACMP are shown in Table 10-1.

Table 10-1. Signal Properties

Signal	Function	I/O
ACMP-	Inverting analog input to the ACMP (Minus input)	I
ACMP+	Non-inverting analog input to the ACMP (Positive input)	I
ACMPO	Digital output of the ACMP	O

10.3 Register Definition

The ACMP includes one register:

- An 8-bit status and control register

Refer to the direct-page register summary in the memory chapter of this data sheet for the absolute address assignments for all ACMP registers.

10.3.1 ACMP Status and Control Register (ACMPSC)

ACMPSC contains the status flag and control bits which are used to enable and configure the ACMP.

	7	6	5	4	3	2	1	0
R	ACME	ACBGS	ACF	ACIE	ACO	ACOPE	ACMOD	
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 10-3. ACMP Status and Control Register (ACMPSC)

Table 10-2. ACMPSC Field Descriptions

Field	Description
7 ACME	**Analog Comparator Module Enable** — ACME enables the ACMP module. 0 ACMP not enabled. 1 ACMP is enabled.
6 ACBGS	**Analog Comparator Bandgap Select** — ACBGS is used to select between the internal bandgap reference voltage or the ACMP+ pin as the non-inverting input of the analog comparator. 0 External pin ACMP+ selected as non-inverting input to comparator. 1 Internal bandgap reference voltage selected as non-inverting input to comparator.
5 ACF	**Analog Comparator Flag** — ACF is set when a compare event occurs. Compare events are defined by ACMOD. ACF is cleared by writing a one to ACF. 0 Compare event has not occurred. 1 Compare event has occurred.
4 ACIE	**Analog Comparator Interrupt Enable** — ACIE enables the interrupt for the ACMP. When ACIE is set, an interrupt will be asserted when ACF is set. 0 Interrupt disabled. 1 Interrupt enabled.
3 ACO	**Analog Comparator Output** — Reading ACO will return the current value of the analog comparator output. ACO is reset to a 0 and will read as a 0 when the ACMP is disabled (ACME = 0).
2 ACOPE	**Analog Comparator Output Pin Enable** — ACOPE is used to enable the comparator output to be placed onto the external pin, ACMPO. ACOPE will only control the pin if the ACMP is active (ACME=1). 0 Analog comparator output not available on ACMPO. 1 Analog comparator output is driven out on ACMPO.
1:0 ACMOD	**Analog Comparator Mode** — ACMOD selects the type of compare event which sets ACF. 00 Encoding 0 — Comparator output falling edge. 01 Encoding 1 — Comparator output rising edge. 10 Encoding 2 — Comparator output falling edge. 11 Encoding 3 — Comparator output rising or falling edge.

10.4 Functional Description

The analog comparator can be used to compare two analog input voltages applied to ACMP+ and ACMP–; or it can be used to compare an analog input voltage applied to ACMP– with an internal bandgap reference voltage. ACBGS is used to select between the bandgap reference voltage or the ACMP+ pin as the input to the non-inverting input of the analog comparator.

The comparator output is high when the non-inverting input is greater than the inverting input, and it is low when the non-inverting input is less than the inverting input. ACMOD is used to select the condition which will cause ACF to be set. ACF can be set on a rising edge of the comparator output, a falling edge of the comparator output, or either a rising or a falling edge (toggle). The comparator output can be read directly through ACO. The comparator output can also be driven onto the ACMPO pin using ACOPE.

NOTE

Comparator inputs are high impedence analog pins which are sensitive to noise. Noisy VDD and/or pin toggling adjacent to the analog inputs may cause the comparator offset/hysteresis performance to exceed the specified values. Maximum source impedence is restricted to the value specified in Table A-6. To achieve maximum performance device is recommended to enter WAIT/STOP mode for ACMP measurement and adjacent pin toggling must be avoided.

Chapter 11
Modulo Timer (RS08MTIMV1)

11.1 Introduction

The MTIM is a simple 8-bit timer with several software selectable clock sources and a programmable interrupt.

The central component of the MTIM is the 8-bit counter that can operate as a free-running counter or a modulo counter. A timer overflow interrupt can be enabled to generate periodic interrupts for time-based software loops.

The TCLK input is connected to the PTA2 pin of the MC9RS08KA2 Series. The XCLK input is connected to the ICSFFCLK clock divided by two, where the ICSFFCLK is the fixed-frequency internal reference clock from the ICS module.

Figure 11-1 shows the MC9RS08KA2 Series block diagram with the MTIM highlighted.

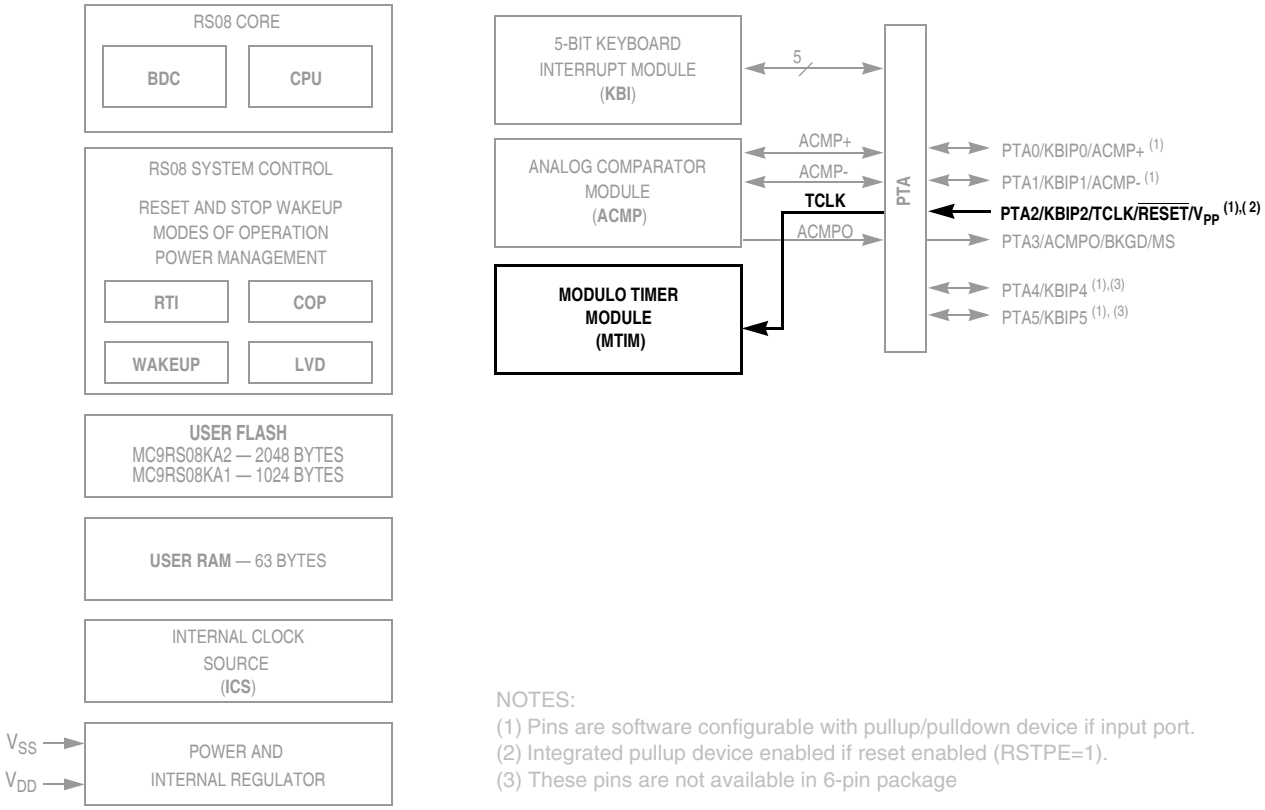


Figure 11-1. MC9RS08KA2 Series Block Diagram Highlighting MTIM Block and Pin

11.1.1 Features

Timer system features include:

- 8-bit up-counter
  - Free-running or 8-bit modulo limit
  - Software controllable interrupt on overflow
  - Counter reset bit (TRST)
  - Counter stop bit (TSTP)
- Four software selectable clock sources for input to prescaler:
  - System bus clock — rising edge
  - Fixed frequency clock (XCLK) — rising edge
  - External clock source on the TCLK pin — rising edge
  - External clock source on the TCLK pin — falling edge
- Nine selectable clock prescale values:
  - Clock source divide by 1, 2, 4, 8, 16, 32, 64, 128, or 256

11.1.2 Modes of Operation

This section defines the MTIM's operation in stop, wait and background debug modes.

11.1.2.1 Operation in Wait Mode

The MTIM continues to run in wait mode if enabled before executing the WAIT instruction. Therefore, the MTIM can be used to bring the MCU out of wait mode if the timer overflow interrupt is enabled. For lowest possible current consumption, the MTIM must be disabled by software if not needed as an interrupt source during wait mode.

11.1.2.2 Operation in Stop Modes

The MTIM is disabled in all stop modes, regardless of the settings before executing the STOP instruction. Therefore, the MTIM cannot be used as a wake up source from stop mode.

If stop is exited with a reset, the MTIM will be put into its reset state. If stop is exited with an interrupt, the MTIM continues from the state it was in when stop was entered. If the counter was active upon entering stop, the count will resume from the current value.

11.1.2.3 Operation in Active Background Mode

The MTIM suspends all counting until the MCU returns to normal user operating mode. Counting resumes from the suspended value as long as an MTIM reset did not occur (TRST written to a 1 or any value is written to the MTIMMOD register).

11.1.3 Block Diagram

The block diagram for the modulo timer module is shown Figure 11-2.

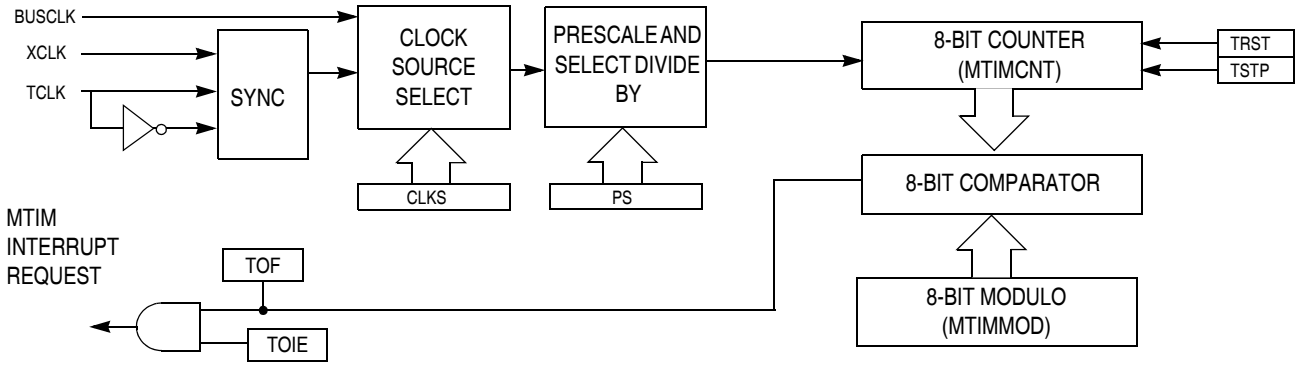


Figure 11-2. Modulo Timer (MTIM) Block Diagram

11.2 External Signal Description

The MTIM includes one external signal, TCLK, used to input an external clock when selected as the MTIM clock source. The signal properties of TCLK are shown in Table 11-1.

Table 11-1. Signal Properties

Signal	Function	I/O
TCLK	External clock source input into MTIM	I

The TCLK input must be synchronized by the bus clock. Also, variations in duty cycle and clock jitter must be accommodated. Therefore, the TCLK signal must be limited to one-fourth of the bus frequency.

The TCLK pin can be muxed with a general-purpose port pin. See the Pins and Connections chapter for the pin location and priority of this function.

11.3 Register Definition

Each MTIM includes four registers, which are summarized in Table 11-2:

- An 8-bit status and control register
- An 8-bit clock configuration register
- An 8-bit counter register
- An 8-bit modulo register

Refer to the direct-page register summary in the memory section of this data sheet for the absolute address assignments for all MTIM registers. This section refers to registers and control bits only by their names.

Table 11-2. MTIM Register Summary

Name		7	6	5	4	3	2	1	0
MTIMSC	R	TOF	TOIE	0	TSTP	0	0	0	0
	W			TRST					
MTIMCLK	R	0	0	CLKS		PS			
	W								
MTIMCNT	R	COUNT							
	W								
MTIMMOD	R	MOD							
	W								

11.3.1 MTIM Status and Control Register (MTIMSC)

MTIMSC contains the overflow status flag and control bits which are used to configure the interrupt enable, reset the counter, and stop the counter.

	7	6	5	4	3	2	1	0
R	TOF	TOIE	0	TSTP	0	0	0	0
W			TRST					
Reset:	0	0	0	1	0	0	0	0

Figure 11-3. MTIM Status and Control Register (MTIMSC)

Table 11-3. MTIMSC Field Descriptions

Field	Description
7 TOF	**MTIM Overflow Flag** — This read-only bit is set when the MTIM counter register overflows to $00 after reaching the value in the MTIM modulo register. Clear TOF by reading the MTIMSC register while TOF is set, then writing a 0 to TOF. TOF is also cleared when TRST is written to a 1 or when any value is written to the MTIMMOD register. 0 MTIM counter has not reached the overflow value in the MTIM modulo register. 1 MTIM counter has reached the overflow value in the MTIM modulo register.
6 TOIE	**MTIM Overflow Interrupt Enable** — This read/write bit enables MTIM overflow interrupts. If TOIE is set, then an interrupt is generated when TOF = 1. Reset clears TOIE. Do not set TOIE if TOF = 1. Clear TOF first, then set TOIE. 0 TOF interrupts are disabled. Use software polling. 1 TOF interrupts are enabled.
5 TRST	**MTIM Counter Reset** — When a 1 is written to this write-only bit, the MTIM counter register resets to $00 and TOF is cleared. Reading this bit always returns 0. 0 No effect. MTIM counter remains at current state. 1 MTIM counter is reset to $00.
4 TSTP	**MTIM Counter Stop** — When set, this read/write bit stops the MTIM counter at its current value. Counting resumes from the current value when TSTP is cleared. Reset sets TSTP to prevent the MTIM from counting. 0 MTIM counter is active. 1 MTIM counter is stopped.

11.3.2 MTIM Clock Configuration Register (MTIMCLK)

MTIMCLK contains the clock select bits (CLKS) and the prescaler select bits (PS).

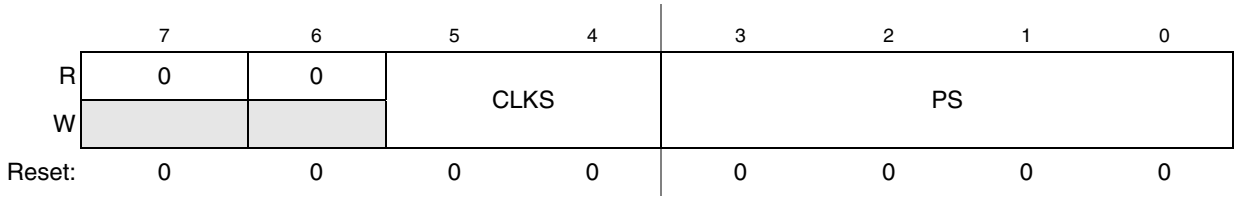


Figure 11-4. MTIM Clock Configuration Register (MTIMCLK)

Table 11-4. MTIMCLK Field Description

Field	Description
5:4 CLKS	**Clock Source Select** — These two read/write bits select one of four different clock sources as the input to the MTIM prescaler. Changing the clock source while the counter is active does not clear the counter. The count continues with the new clock source. Reset clears CLKS to 00. 00 Encoding 0 — Bus clock (BUSCLK). 01 Encoding 1 — Fixed-frequency clock (XCLK). 10 Encoding 3 — External source (TCLK pin), falling edge. 11 Encoding 4 — External source (TCLK pin), rising edge.
3:0 PS	**Clock Source Prescaler** — These four read/write bits select one of nine outputs from the 8-bit prescaler. Changing the prescaler value while the counter is active does not clear the counter. The count continues with the new prescaler value. Reset clears PS to 0000. 0000 Encoding 0 — MTIM clock source ÷ 1. 0001 Encoding 1 — MTIM clock source ÷ 2. 0010 Encoding 2 — MTIM clock source ÷ 4. 0011 Encoding 3 — MTIM clock source ÷ 8. 0100 Encoding 4 — MTIM clock source ÷ 16. 0101 Encoding 5 — MTIM clock source ÷ 32. 0110 Encoding 6 — MTIM clock source ÷ 64. 0111 Encoding 7 — MTIM clock source ÷ 128. 1000 Encoding 8 — MTIM clock source ÷ 256. All other encodings default to MTIM clock source ÷ 256.

11.3.3 MTIM Counter Register (MTIMCNT)

MTIMCNT is the read-only value of the current MTIM count of the 8-bit counter.

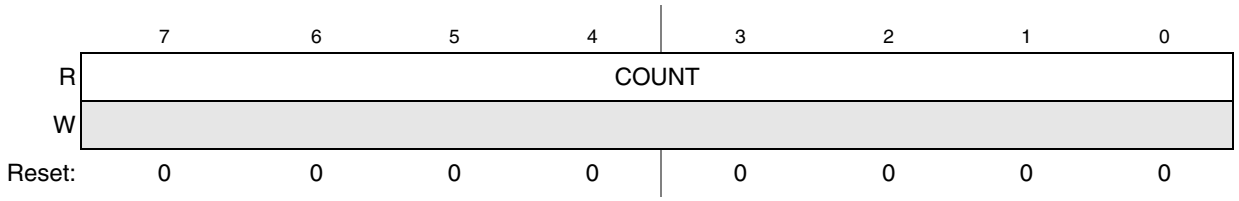


Figure 11-5. MTIM Counter Register (MTIMCNT)

Table 11-5. MTIMCNT Field Description

Field	Description
7:0 COUNT	**MTIM Count** — These eight read-only bits contain the current value of the 8-bit counter. Writes have no effect to this register. Reset clears the count to $00.

11.3.4 MTIM Modulo Register (MTIMMOD)

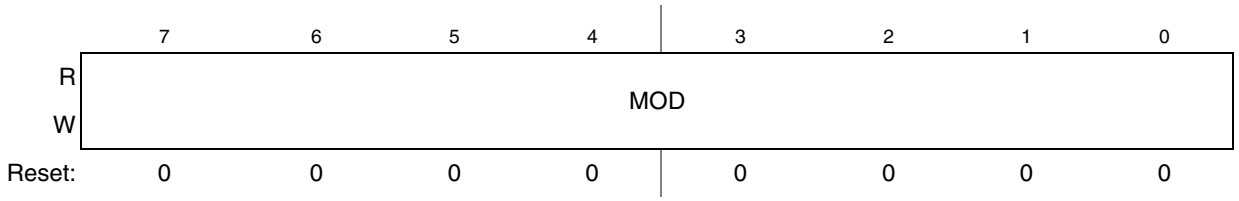


Figure 11-6. MTIM Modulo Register (MTIMMOD)

Table 11-6. MTIMMOD Descriptions

Field	Description
7:0 MOD	**MTIM Modulo** — These eight read/write bits contain the modulo value used to reset the count and set TOF. A value of $00 puts the MTIM in free-running mode. Writing to MTIMMOD resets the COUNT to $00 and clears TOF. Reset sets the modulo to $00.

11.4 Functional Description

The MTIM is composed of a main 8-bit up-counter with an 8-bit modulo register, a clock source selector, and a prescaler block with nine selectable values. The module also contains software selectable interrupt logic.

The MTIM counter (MTIMCNT) has three modes of operation: stopped, free-running, and modulo. Out of reset, the counter is stopped. If the counter is started without writing a new value to the modulo register, then the counter will be in free-running mode. The counter is in modulo mode when a value other than $00 is in the modulo register while the counter is running.

After any MCU reset, the counter is stopped and reset to $00, and the modulus is set to $00. The bus clock is selected as the default clock source and the prescale value is divide by 1. To start the MTIM in free-running mode, simply write to the MTIM status and control register (MTIMSC) and clear the MTIM stop bit (TSTP).

Four clock sources are software selectable: the internal bus clock, the fixed frequency clock (XCLK), and an external clock on the TCLK pin, selectable as incrementing on either rising or falling edges. The MTIM clock select bits (CLKS) in MTIMCLK are used to select the desired clock source. If the counter is active (TSTP = 0) when a new clock source is selected, the counter will continue counting from the previous value using the new clock source.

Nine prescale values are software selectable: clock source divided by 1, 2, 4, 8, 16, 32, 64, 128, or 256. The prescaler select bits (PS) in MTIMCLK select the desired prescale value. If the counter is active (TSTP = 0) when a new prescaler value is selected, the counter will continue counting from the previous value using the new prescaler value.

The MTIM modulo register (MTIMMOD) allows the overflow compare value to be set to any value from $01 to $FF. Reset clears the modulo value to $00, which results in a free running counter.

When the counter is active (TSTP = 0), the counter increments at the selected rate until the count matches the modulo value. When these values match, the counter overflows to $00 and continues counting. The MTIM overflow flag (TOF) is set whenever the counter overflows. The flag sets on the transition from the modulo value to $00. Writing to MTIMMOD while the counter is active resets the counter to $00 and clears TOF.

Clearing TOF is a two-step process. The first step is to read the MTIMSC register while TOF is set. The second step is to write a 0 to TOF. If another overflow occurs between the first and second steps, the clearing process is reset and TOF will remain set after the second step is performed. This will prevent the second occurrence from being missed. TOF is also cleared when a 1 is written to TRST or when any value is written to the MTIMMOD register.

The MTIM allows for an optional interrupt to be generated whenever TOF is set. To enable the MTIM overflow interrupt, set the MTIM overflow interrupt enable bit (TOIE) in MTIMSC. TOIE must never be written to a 1 while TOF = 1. Instead, TOF must be cleared first, then the TOIE can be set to 1.

11.4.1 MTIM Operation Example

This section shows an example of the MTIM operation as the counter reaches a matching value from the modulo register.

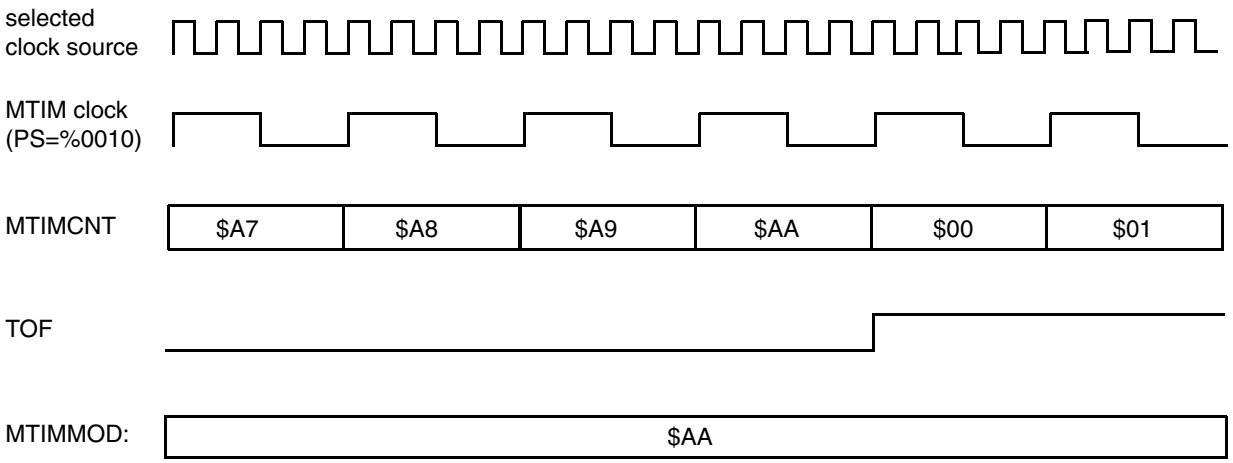


Figure 11-7. MTIM Counter Overflow Example

In the example of Figure 11-7, the selected clock source could be any of the four possible choices. The prescaler is set to PS = %0010 or divide-by-4. The modulo value in the MTIMMOD register is set to $AA. When the counter, MTIMCNT, reaches the modulo value of $AA, the counter overflows to $00 and continues counting. The timer overflow flag, TOF, sets when the counter value changes from $AA to $00. An MTIM overflow interrupt is generated when TOF is set, if TOIE = 1.

Chapter 12
Development Support

12.1 Introduction

Development support systems in the RS08 family include the RS08 background debug controller (BDC).

The BDC provides a single-wire debug interface to the target MCU. This interface provides a convenient means for programming the on-chip FLASH and other nonvolatile memories. Also, the BDC is the primary debug interface for development and allows non-intrusive access to memory data and traditional debug features such as CPU register modify, breakpoint, and single-instruction trace commands.

In the RS08 Family, address and data bus signals are not available on external pins. Debug is done through commands fed into the target MCU via the single-wire background debug interface, including resetting the device without using a reset pin.

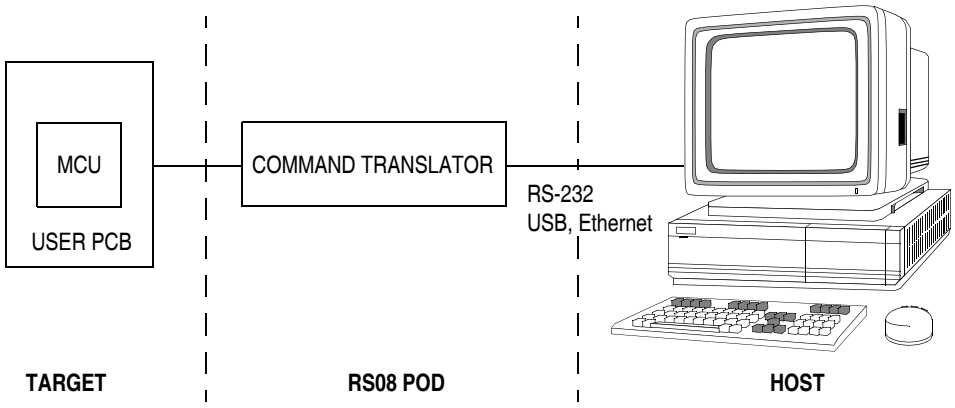


Figure 12-1. Connecting MCU to Host for Debugging

12.2 Features

Features of the RS08 background debug controller (BDC) include:

- Uses a single pin for background debug serial communications
- Non-intrusive of user memory resources; BDC registers are not located in the memory map
- SYNC command to determine target communications rate
- Non-intrusive commands allow access to memory resources while CPU is running user code without stopping applications
- Active background mode commands for CPU register access
- GO and TRACE1 commands
- BACKGROUND command can wake CPU from wait or stop modes

- BDC_RESET command allows host to reset MCU without using a reset pin
- One hardware address breakpoint built into BDC
- RS08 clock source runs in stop mode if BDM enabled to allow debugging when CPU is in stop mode
- COP watchdog suspended while in active background mode

12.3 RS08 Background Debug Controller (BDC)

All MCUs in the RS08 Family contain a single-wire background debug interface which supports in-circuit programming of on-chip non-volatile memory and sophisticated debug capabilities. Unlike debug interfaces on earlier 8-bit MCUs, this debug system provides for minimal interference with normal application resources. It does not use any user memory or locations in the memory map. It requires use of only the output-only BKGD pin. This pin will be shared with simple user output-only functions (typically port, comparator outputs, etc.), which can be easily debugged in normal user mode.

RS08 BDM commands are divided into two groups:

- Active background mode commands require that the target MCU is in active background mode (the user program is not running). The BACKGROUND command causes the target MCU to enter active background mode. Active background mode commands allow the CPU registers to be read or written and allow the user to trace one (TRACE1) user instruction at a time or GO to the user program from active background mode.
- Non-intrusive commands can be executed at any time even while the user program is running. Non-intrusive commands allow a user to read or write MCU memory locations or access status and control registers within the background debug controller (BDC).

Typically, a relatively simple interface pod is used to translate commands from a host computer into commands for the custom serial interface to the single-wire background debug system. Depending on the development tool vendor, this interface pod may use a standard RS-232 serial port, a parallel printer port, or some other type of communication such as Ethernet or a universal serial bus (USB) to communicate between the host PC and the pod.

Figure 12-2 shows the standard header for connection of a RS08 BDM pod. A pod is a small interface device that connects a host computer such as a personal computer to a target RS08 system. BKGD and GND are the minimum connections required to communicate with a target MCU. The pseudo-open-drain $\overline{\text{RESET}}$ signal is included in the connector to allow a direct hardware method for the host to force or monitor (if $\overline{\text{RESET}}$ is available as output) a target system reset.

The RS08 BDM pods supply the V_{PP} voltage to the RS08 MCU when in-circuit programming is required. The V_{PP} connection from the pod is shared with $\overline{\text{RESET}}$ as shown in Figure 12-2. For V_{PP} requirements see the FLASH specifications in the electricals appendix.

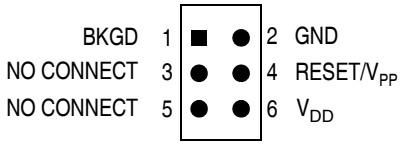


Figure 12-2. Standard RS08 BDM Tool Connector

Background debug controller (BDC) serial communications use a custom serial protocol that was first introduced on the M68HC12 Family of microcontrollers. This protocol requires that the host knows the communication clock rate, which is determined by the target BDC clock rate. If a host is attempting to communicate with a target MCU that has an unknown BDC clock rate, a SYNC command may be sent to the target MCU to request a timed sync response signal from which the host can determine the correct communication speed.

For RS08 MCUs, the BDC clock is the same frequency as the MCU bus clock. For a detailed description of the communications protocol, refer to Section 12.3.2, “Communication Details."

12.3.1 BKGD Pin Description

BKGD is the single-wire background debug interface pin. BKGD is a pseudo-open-drain pin that contains an on-chip pullup, therefore it requires no external pullup resistor. Unlike typical open-drain pins, the external resistor capacitor (RC) time constant on this pin, which is influenced by external capacitance, plays almost no role in signal rise time. The custom protocol provides for brief, actively driven speedup pulses to force rapid rise times on this pin without risking harmful drive level conflicts. Refer to Section 12.3.2, “Communication Details," for more detail.

The primary function of this pin is bidirectional serial communication of background debug commands and data. During reset, this pin selects between starting in active background mode and normal user mode running an application program. This pin is also used to request a timed sync response pulse to allow a host development tool to determine the target BDC clock frequency.

By controlling the BKGD pin and forcing an MCU reset (issuing a BDC_RESET command, or through a power-on reset (POR)), the host can force the target system to reset into active background mode rather than start the user application program. This is useful to gain control of a target MCU whose FLASH program memory has not yet been programmed with a user application program.

When no debugger pod is connected to the 6-pin BDM interface connector, the internal pullup on BKGD determines the normal operating mode.

On some RS08 devices, the BKGD pin may be shared with an alternative output-only function. To support BDM debugging, the user must disable this alternative function. Debugging of the alternative function must be done in normal user mode without using BDM.

12.3.2 Communication Details

The BDC serial interface requires the host to generate a falling edge on the BKGD pin to indicate the start of each bit time. The host provides this falling edge whether data is transmitted or received.

The BDC serial communication protocol requires the host to know the target BDC clock speed. Commands and data are sent most significant bit first (MSB-first) at 16 BDC clock cycles per bit. The interface times out if 512 BDC clock cycles occur between falling edges from the host. Any BDC command that was in progress when this timeout occurs is aborted without affecting the memory or operating mode of the target MCU system.

Figure 12-3 shows an external host transmitting a logic 1 or 0 to the BKGD pin of a target MCU. The host is asynchronous to the target so there is a 0-to-1 cycle delay from the host-generated falling edge to where the target perceives the beginning of the bit time. Ten target BDC clock cycles later, the target senses the bit level on the BKGD pin. Typically, the host actively drives the pseudo-open-drain BKGD pin during host-to-target transmissions to speed up rising edges. Because the target does not drive the BKGD pin during the host-to-target period, there is no need to treat the line as an open-drain signal during this period.

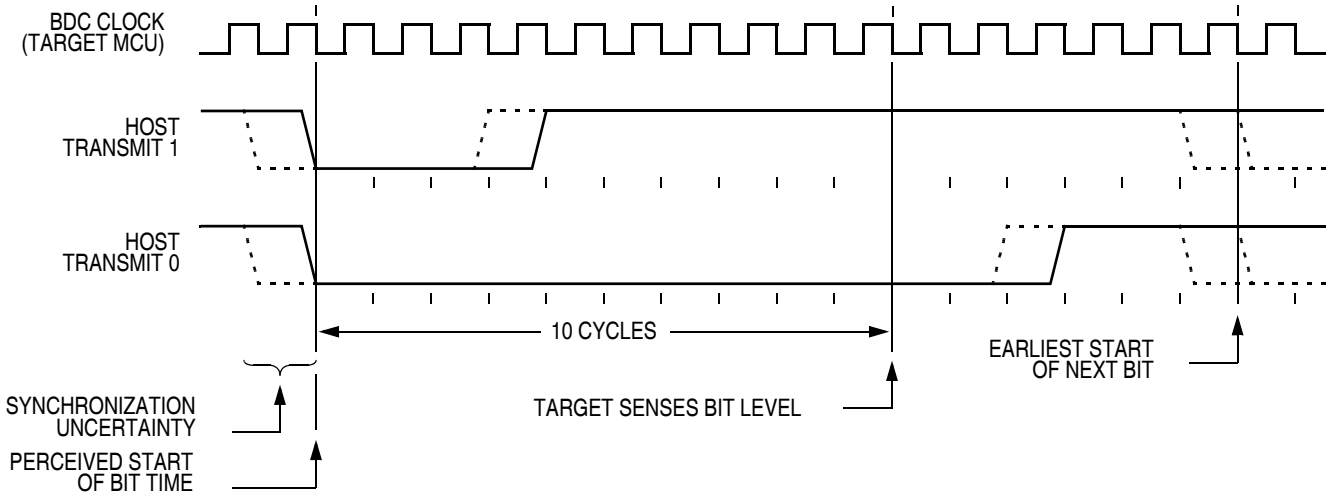


Figure 12-3. BDC Host-to-Target Serial Bit Timing

Figure 12-4 shows the host receiving a logic 1 from the target MCU. Because the host is asynchronous to the target, there is a 0-to-1 cycle delay from the host-generated falling edge on BKGD to the perceived start of the bit time in the target. The host holds the BKGD pin low long enough for the target to recognize it (at least two target BDC cycles). The host must release the low drive before the target drives a brief active-high speedup pulse seven cycles after the perceived start of the bit time. The host must sample the bit level approximately 10 cycles after it started the bit time.

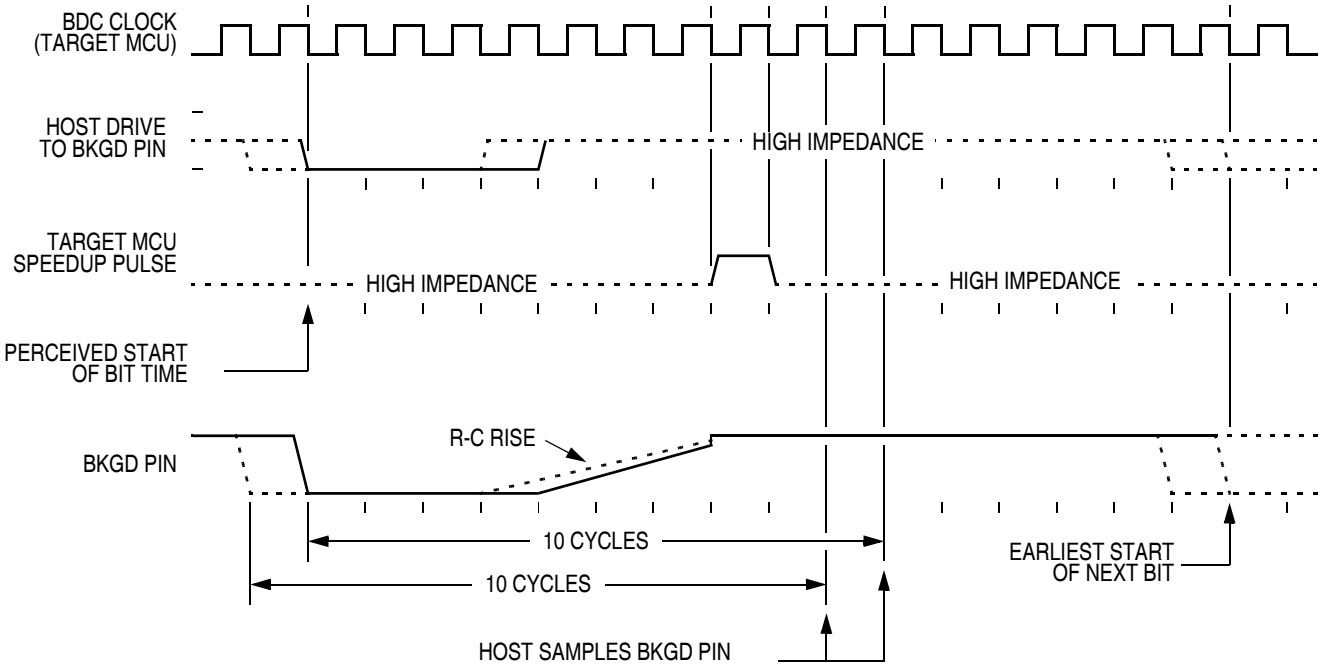


Figure 12-4. BDC Target-to-Host Serial Bit Timing (Logic 1)

Figure 12-5 shows the host receiving a logic 0 from the target MCU. Because the host is asynchronous to the target, there is a 0-to-1 cycle delay from the host-generated falling edge on BKGD to the start of the bit time as perceived by the target. The host initiates the bit time but the target finishes it. Because the target wants the host to receive a logic 0, it drives the BKGD pin low for 13 BDC clock cycles, then briefly drives it high to speed up the rising edge. The host samples the bit level approximately 10 cycles after starting the bit time.

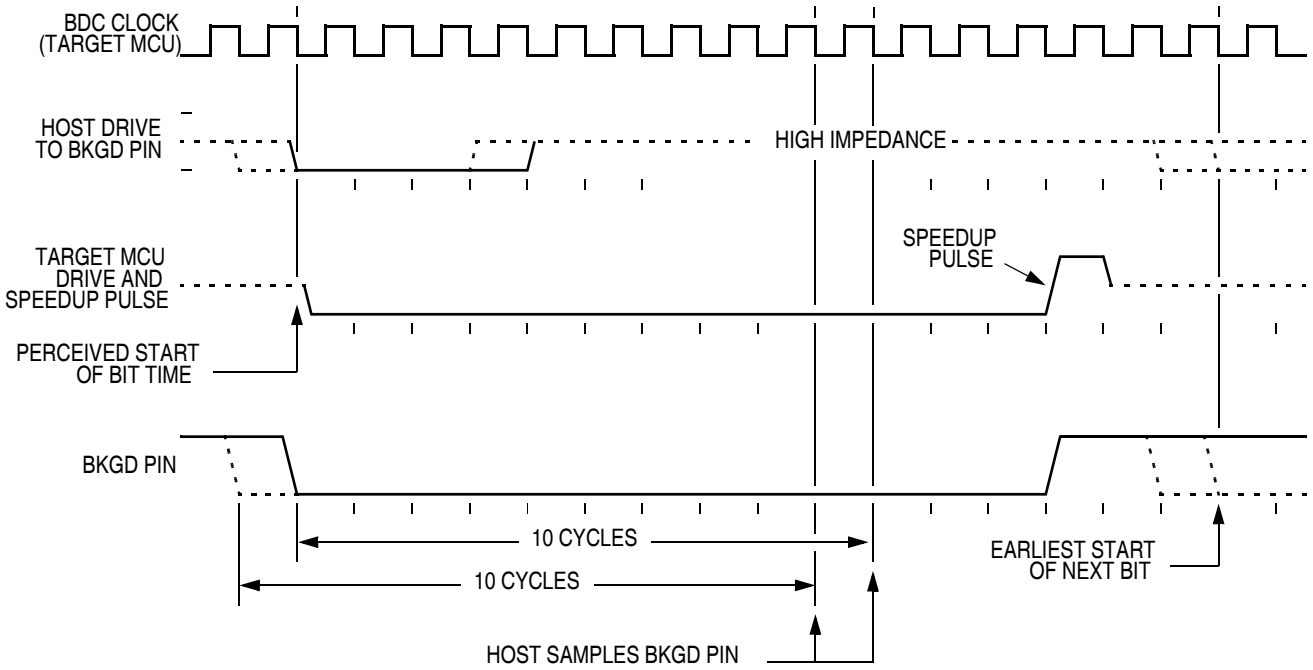


Figure 12-5. BDM Target-to-Host Serial Bit Timing (Logic 0)

12.3.3 SYNC and Serial Communication Timeout

The host initiates a host-to-target serial transmission by generating a falling edge on the BKGD pin. If BKGD is kept low for more than 128 target clock cycles, the target understands that a SYNC command was issued. In this case, the target will keep waiting for a rising edge on BKGD to answer the SYNC request pulse. If the rising edge is not detected, the target will keep waiting indefinitely, without any timeout limit. When a rising edge on BKGD occurs after a valid SYNC request, the BDC will drive the BKGD pin low for exactly 128 BDC cycles.

Consider now the case where the host returns BKGD to logic 1 before 128 cycles. This is interpreted as a valid bit transmission, and not as a SYNC request. The target will keep waiting for another falling edge marking the start of a new bit. If, however, a new falling edge is not detected by the target within 512 clock cycles since the last falling edge, a timeout occurs and the current command is discarded without affecting memory or the operating mode of the MCU. This is referred as a soft-reset to the BDC.

If a read command is issued but the data is not retrieved within 512 serial clock cycles, a soft-reset will occur causing the command to be disregarded. The data is not available for retrieving after the timeout has occurred. A soft-reset is also used to end a READ_BLOCK or WRITE_BLOCK command.

The following describes the actual bit-time requirements for a host to guarantee logic 1 or 0 bit transmission without the target timing out or interpreting the bit as a SYNC command:

- To send a logic 0, BKGD must be kept low for a minimum of 12 BDC cycles and up to 511 BDC cycles except for the first bit of a command sequence, which will be detected as a SYNC request.
- To send a logic 1, BKGD must be held low for at least four BDC cycles, be released by the eighth cycle, and be held high until at least the sixteenth BDC cycle.

- Subsequent bits must occur within 512 BDC cycles of the last bit sent.

12.4 BDC Registers and Control Bits

The BDC contains two non-CPU accessible registers:

- The BDC status and control register (BDCSCR) is an 8-bit register containing control and status bits for the background debug controller.
- The BDC breakpoint register (BDCBKPT) holds a 16-bit breakpoint match address.

These registers are accessed with dedicated serial BDC commands and are not located in the memory space of the target MCU (so they do not have addresses and cannot be accessed by user programs).

Some of the bits in the BDCSCR have write limitations; otherwise, these registers may be read or written at any time. For example, the ENBDM control bit may not be written while the MCU is in active background mode. This prevents the ambiguous condition of the control bit forbidding active background mode while the MCU is already in active background mode. Also, the status bits (BDMACT, WS, and WSF) are read-only status indicators and can never be written by the WRITE_CONTROL serial BDC command.

12.4.1 BDC Status and Control Register (BDCSCR)

This register can be read or written by serial BDC commands (READ_STATUS and WRITE_CONTROL) but is not accessible to user programs because it is not located in the normal memory map of the MCU.

	7	6	5	4	3	2	1	0
R	ENBDM	BDMACT	BKPTEN	FTS	0	WS	WSF	0
W								
Normal Reset	0	0	0	0	0	0	0	0
Reset in Active BDM:	1	1	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-6. BDC Status and Control Register (BDCSCR)

Table 12-1. BDCSCR Register Field Descriptions

Field	Description
7 ENBDM	**Enable BDM (Permit Active Background Mode)** — Typically, this bit is written to 1 by the debug host shortly after the beginning of a debug session or whenever the debug host resets the target and remains 1 until a normal reset clears it. If the application can go into stop mode, this bit is required to be set if debugging capabilities are required. 0 BDM cannot be made active (non-intrusive commands still allowed). 1 BDM can be made active to allow active background mode commands.
6 BDMACT	**Background Mode Active Status** — This is a read-only status bit. 0 BDM not active (user application program running). 1 BDM active and waiting for serial commands.

Table 12-1. BDCSCR Register Field Descriptions (continued)

Field	Description
5 BKPTEN	**BDC Breakpoint Enable** — If this bit is clear, the BDC breakpoint is disabled and the FTS (force tag select) control bit and BDCBKPT match register are ignored 0 BDC breakpoint disabled. 1 BDC breakpoint enabled.
4 FTS	**Force/Tag Select** — When FTS = 1, a breakpoint is requested whenever the CPU address bus matches the BDCBKPT match register. When FTS = 0, a match between the CPU address bus and the BDCBKPT register causes the fetched opcode to be tagged. If this tagged opcode ever reaches the end of the instruction queue, the CPU enters active background mode rather than executing the tagged opcode. 0 Tag opcode at breakpoint address and enter active background mode if CPU attempts to execute that instruction. 1 Breakpoint match forces active background mode at next instruction boundary (address need not be an opcode).
2 WS	**Wait or Stop Status** — When the target CPU is in wait or stop mode, most BDC commands cannot function. However, the BACKGROUND command can be used to force the target CPU out of wait or stop and into active background mode where all BDC commands work. Whenever the host forces the target MCU into active background mode, the host must issue a READ_STATUS command to check that BDMACT = 1 before attempting other BDC commands. 0 Target CPU is running user application code or in active background mode (was not in wait or stop mode when background became active). 1 Target CPU is in wait or stop mode, or a BACKGROUND command was used to change from wait or stop to active background mode.
1 WSF	**Wait or Stop Failure Status** — This status bit is set if a memory access command failed due to the target CPU executing a wait or stop instruction at or about the same time. The usual recovery strategy is to issue a BACKGROUND command to get out of wait or stop mode into active background mode, repeat the command that failed, then return to the user program. (Typically, the host would restore CPU registers and stack values and re-execute the wait or stop instruction.) 0 Memory access did not conflict with a wait or stop instruction. 1 Memory access command failed because the CPU entered wait or stop mode.

12.4.2 BDC Breakpoint Match Register

This 16-bit register holds the 14-bit address for the hardware breakpoint in the BDC. The BKPTEN and FTS control bits in BDCSCR are used to enable and configure the breakpoint logic. Dedicated serial BDC commands (READ_BKPT and WRITE_BKPT) are used to read and write the BDCBKPT register. Breakpoints are normally set while the target MCU is in active background mode before running the user application program. However, because READ_BKPT and WRITE_BKPT are non-intrusive commands, they could be executed even while the user program is running. For additional information about setup and use of the hardware breakpoint logic in the BDC, refer to the RS08 Family Reference Manual.”

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
W																
Any Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-7. BDC Breakpoint Match Register (BDCBKPT)

12.5 RS08 BDC Commands

BDC commands are sent serially from a host computer to the BKGD pin of the target MCU. All commands and data are sent MSB-first using a custom BDC communications protocol. Active background mode commands require that the target MCU is currently in the active background mode while non-intrusive commands may be issued at any time whether the target MCU is in active background mode or running a user application program.

Table 12-2 shows all RS08 BDC commands, a shorthand description of their coding structure, and the meaning of each command.

Coding Structure Nomenclature

The following nomenclature is used in Table 12-2 to describe the coding structure of the BDC commands.

		Commands begin with an 8-bit command code in the host-to-target direction (most significant bit first)
/	=	Separates parts of the command
d	=	Delay 16 to 511 target BDC clock cycles
soft-reset	=	Delay of at least 512 BDC clock cycles from last host falling-edge
AAAA	=	16-bit address in the host-to-target direction[1]
RD	=	Eight bits of read data in the target-to-host direction
WD	=	Eight bits of write data in the host-to-target direction
RD16	=	16 bits of read data in the target-to-host direction
WD16	=	16 bits of write data in the host-to-target direction
SS	=	the contents of BDCSCR in the target-to-host direction (STATUS)
CC	=	Eight bits of write data for BDCSCR in the host-to-target direction (CONTROL)
RBKP	=	16 bits of read data in the target-to-host direction (from BDCBKPT breakpoint register)
WBKP	=	16 bits of write data in the host-to-target direction (for BDCBKPT breakpoint register)

1. The RS08 CPU uses only 14 bits of address and occupies the lower 14 bits of the 16-bit AAAA address field. The values of address bits 15 and 14 in AAAA are truncated and thus do not matter.

Table 12-2. RS08 BDC Command Summary

Command Mnemonic	Active Background Mode/ Non-Intrusive	Coding Structure	Description
SYNC	Non-intrusive	n/a[1]	Request a timed reference pulse to determine target BDC communication speed
BDC_RESET	Any CPU mode	18[2]	Request an MCU reset
BACKGROUND	Non-intrusive	90/d	Enter active background mode if enabled (ignore if ENBDM bit equals 0)
READ_STATUS	Non-intrusive	E4/SS	Read BDC status from BDCSCR
WRITE_CONTROL	Non-intrusive	C4/CC	Write BDC controls in BDCSCR
READ_BYTE	Non-intrusive	E0/AAAA/d/RD	Read a byte from target memory
READ_BYTE_WS	Non-intrusive	E1/AAAA/d/SS/RD	Read a byte and report status
WRITE_BYTE	Non-intrusive	C0/AAAA/WD/d	Write a byte to target memory
WRITE_BYTE_WS	Non-intrusive	C1/AAAA/WD/d/SS	Write a byte and report status
READ_BKPT	Non-intrusive	E2/RBKP	Read BDCBKPT breakpoint register
WRITE_BKPT	Non-intrusive	C2/WBKP	Write BDCBKPT breakpoint register
GO	Active background mode	08/d	Go to execute the user application program starting at the address currently in the PC
TRACE1	Active background mode	10/d	Trace one user instruction at the address in the PC, then return to active background mode
READ_BLOCK	Active background mode	80/AAAA/d/RD[3]	Read a block of data from target memory starting from AAAA continuing until a soft-reset is detected
WRITE_BLOCK	Active background mode	88/AAAA/WD/d[4]	Write a block of data to target memory starting at AAAA continuing until a soft-reset is detected
READ_A	Active background mode	68/d/RD	Read accumulator (A)

Table 12-2. RS08 BDC Command Summary (continued)

Command Mnemonic	Active Background Mode/ Non-Intrusive	Coding Structure	Description
WRITE_A	Active background mode	48/WD/d	Write accumulator (A)
READ_CCR_PC	Active background mode	6B/d/RD16[5]	Read the CCR bits z, c concatenated with the 14-bit program counter (PC) RD16=zc:PC
WRITE_CCR_PC	Active background mode	4B/WD16/d[6]	Write the CCR bits z, c concatenated with the 14-bit program counter (PC) WD16=zc:PC
READ_SPC	Active background mode	6F/d/RD16[7]	Read the 14-bit shadow program counter (SPC) RD16=0:0:SPC
WRITE_SPC	Active background mode	4F/WD16/d[8]	Write 14-bit shadow program counter (SPC) WD16 = x:x:SPC, the two most significant bits shown by “x” are ignored by target

1 The SYNC command is a special operation which does not have a command code.
2 18 was HCS08 BDC command for TAGGO.
3 Each RD requires a delay between host read data byte and next read, command ends when target detects a soft-reset.
4 Each WD requires a delay between host write data byte and next byte, command ends when target detects a soft-reset.
5 HCS08 BDC had separate READ_CCR and READ_PC commands, the RS08 BDC combined this commands.
6 HCS08 BDC had separate WRITE_CCR and WRITE_PC commands, the RS08 BDC combined this commands.
7 6F is READ_SP (read stack pointer) for HCS08 BDC.
8 4F is WRITE_SP (write stack pointer) for HCS08 BDC.

Appendix A
Electrical Characteristics

A.1 Introduction

This chapter contains electrical and timing specifications.

A.2 Absolute Maximum Ratings

Absolute maximum ratings are stress ratings only, and functional operation at the maxima is not guaranteed. Stress beyond the limits specified in Table A-1 may affect device reliability or cause permanent damage to the device. For functional operating conditions, refer to the remaining tables in this chapter.

This device contains circuitry protecting against damage due to high static voltage or electrical fields; however, it is advised that normal precautions be taken to avoid application of any voltages higher than maximum-rated voltages to this high-impedance circuit. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (for instance, either V_{SS} or V_{DD}) or the programmable pull-up resistor associated with the pin is enabled.

Table A-1. Absolute Maximum Ratings

Rating	Symbol	Value	Unit
Supply voltage	V_{DD}	–0.3 to +5.8	V
Maximum current into V_{DD}	I_{DD}	120	mA
Digital input voltage	V_{In}	–0.3 to V_{DD} + 0.3	V
Instantaneous maximum current Single pin limit (applies to all port pins)[1, 2, 3]	I_D	± 25	mA
Storage temperature range	T_{stg}	–55 to 150	°C

1 Input must be current limited to the value specified. To determine the value of the required current-limiting resistor, calculate resistance values for positive (V_{DD}) and negative (V_{SS}) clamp voltages, then use the larger of the two resistance values.

2 All functional non-supply pins are internally clamped to V_{SS} and V_{DD} except the $\overline{\text{RESET}}/V_{PP}$ pin which is internally clamped to V_{SS} only.

3 Power supply must maintain regulation within operating V_{DD} range during instantaneous and operating maximum current conditions. If positive injection current ($V_{In} > V_{DD}$) is greater than I_{DD}, the injection current may flow out of V_{DD} and could result in external power supply going out of regulation. Ensure external V_{DD} load will shunt current greater than maximum injection current. This will be the greatest risk when the MCU is not consuming power. Examples are: if no system clock is present, or if the clock rate is very low which would reduce overall power consumption.

A.3 Thermal Characteristics

This section provides information about operating temperature range, power dissipation, and package thermal resistance. Power dissipation on I/O pins is usually small compared to the power dissipation in on-chip logic and voltage regulator circuits and it is user-determined rather than being controlled by the MCU design. In order to take $P_{I/O}$ into account in power calculations, determine the difference between actual pin voltage and V_{SS} or V_{DD} and multiply by the pin current for each I/O pin. Except in cases of unusually high pin current (heavy loads), the difference between pin voltage and V_{SS} or V_{DD} will be very small.

Table A-2. Thermal Characteristics

Rating	Symbol	Value	Unit
Operating temperature range (packaged)	T_A	T_L to T_H –40 to 85	°C
Maximum junction temperature	T_{JMAX}	105	°C
Thermal resistance [1,2,3,4]			
6-pin DFN			
1s	θ_{JA}	225	°C/W
2s2p		53	
8-pin PDIP			
1s		115	
2s2p		74	
8-pin SOIC			
1s		160	
2s2p		98	

1. Junction temperature is a function of die size, on-chip power dissipation, package thermal resistance, mounting site (board) temperature, ambient temperature, air flow, power dissipation of other components on the board, and board thermal resistance.
2. Junction to Ambient Natural Convection
3. 1s - Single Layer Board, one signal layer
4. 2s2p - Four Layer Board, 2 signal and 2 power layers

The average chip-junction temperature (T_J) in °C can be obtained from:

$$T_J = T_A + (P_D \times \theta_{JA}) \qquad \textit{Eqn. A-1}$$

where:

T_A = Ambient temperature, °C
θ_{JA} = Package thermal resistance, junction-to-ambient, °C/W
$P_D = P_{int} + P_{I/O}$
$P_{int} = I_{DD} \times V_{DD}$, Watts — chip internal power
$P_{I/O}$ = Power dissipation on input and output pins — user determined

For most applications, $P_{I/O} << P_{int}$ and can be neglected. An approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad \textit{Eqn. A-2}$$

Solving Equation A-1 and Equation A-2 for K gives:

$$K = P_D \times (T_A + 273°C) + \theta_{JA} \times (P_D)^2 \quad \textit{Eqn. A-3}$$

where K is a constant pertaining to the particular part. K can be determined from Equation A-3 by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations 1 and 2 iteratively for any value of T_A.

A.4 Electrostatic Discharge (ESD) Protection Characteristics

Although damage from static discharge is much less common on these devices than on early CMOS circuits, normal handling precautions should be used to avoid exposure to static discharge. Qualification tests are performed to ensure that these devices can withstand exposure to reasonable levels of static without suffering any permanent damage. All ESD testing is in conformity with CDF-AEC-Q00 Stress Test Qualification for Automotive Grade Integrated Circuits. (http://www.aecouncil.com/) A device is considered to have failed if, after exposure to ESD pulses, the device no longer meets the device specification requirements. Complete dc parametric and functional testing is performed per the applicable device specification at room temperature followed by hot temperature, unless specified otherwise in the device specification.

Table A-3. ESD Protection Characteristics

Parameter	Symbol	Value	Unit
ESD Target for Machine Model (MM) MM circuit description	V_{THMM}	200	V
ESD Target for Human Body Model (HBM) HBM circuit description	V_{THHBM}	2000	V

A.5 DC Characteristics

This section includes information about power supply requirements, I/O pin characteristics, and power supply current in various operating modes.

Table A-4. DC Characteristics
(Temperature Range = –40 to 85°C Ambient)

Parameter	Symbol	Min	Typical	Max	Unit
Supply voltage (run, wait and stop modes.) $0 < f_{Bus} < 10MHz$	V_{DD}	1.8	—	5.5	V
Minimum RAM retention supply voltage applied to V_{DD}	V_{RAM}	0.8 [1]	—	—	V
Low-voltage Detection threshold (V_{DD} falling) (V_{DD} rising)	V_{LVD}	1.80 1.88	1.86 1.94	1.95 2.03	V
Power on RESET (POR) voltage	V_{POR}	0.9	1.4	1.7	V
Input high voltage ($V_{DD} > 2.3V$) (all digital inputs)	V_{IH}	$0.70 \times V_{DD}$	—	—	V
Input high voltage ($1.8\ V \le V_{DD} \le 2.3\ V$) (all digital inputs)	V_{IH}	$0.85 \times V_{DD}$	—	—	V

Table A-4. DC Characteristics (continued)
(Temperature Range = –40 to 85°C Ambient)

Parameter	Symbol	Min	Typical	Max	Unit
Input low voltage (V_{DD} > 2.3 V) (all digital inputs)	V_{IL}	—	—	0.30 × V_{DD}	V
Input low voltage (1.8 V ≤ V_{DD} ≤ 2.3 V) (all digital inputs)	V_{IL}	—	—	0.30 × V_{DD}	V
Input hysteresis (all digital inputs)	V_{hys}	0.06 × V_{DD}	—	—	V
Input leakage current (per pin) V_{In} = V_{DD} or $V_{SS,}$ all input only pins	$\lvert I_{In}\rvert$	—	0.025	1.0	µA
High impedance (off-state) leakage current (per pin) V_{In} = V_{DD} or V_{SS}, all input/output	$\lvert I_{OZ}\rvert$	—	0.025	1.0	µA
Internal pullup/pulldown resistors[2] (all port pins)	R_{PU}	20	45	65	kΩ
Output high voltage (port A) I_{OH} = –5 mA (V_{DD} ≥ 4.5 V) I_{OH} = –3 mA (V_{DD} ≥ 3 V) I_{OH} = –2 mA (V_{DD} ≥ 1.8 V)	V_{OH}	V_{DD} – 0.8	—	— — —	V
Maximum total I_{OH} for all port pins	$\lvert I_{OHT}\rvert$	—	—	40	mA
Output low voltage (port A) I_{OL} = 5 mA (V_{DD} ≥ 4.5 V) I_{OL} = 3 mA (V_{DD} ≥ 3 V) I_{OL} = 2 mA (V_{DD} ≥ 1.8 V)	V_{OL}	—	—	0.8 0.8 0.8	V
Maximum total I_{OL} for all port pins	I_{OLT}	—	—	40	mA
dc injection current[3, 4, 5 6] V_{In} < V_{SS}, V_{In} > V_{DD} Single pin limit Total MCU limit, includes sum of all stressed pins	$\lvert I_{IC}\rvert$	—	—	 0.2 0.8	 mA mA
Input capacitance (all non-supply pins)	C_{In}	—	—	7	pF

1 This parameter is characterized and not tested on each device.

2 Measurement condition for pull resistors: V_{In} = V_{SS} for pullup and V_{In} = V_{DD} for pulldown.

3 All functional non-supply pins are internally clamped to V_{SS} and V_{DD} except the $\overline{\text{RESET}}$/V_{PP} which is internally clamped to V_{SS} only.

4 Input must be current limited to the value specified. To determine the value of the required current-limiting resistor, calculate resistance values for positive and negative clamp voltages, then use the larger of the two values.

5 Power supply must maintain regulation within operating V_{DD} range during instantaneous and operating maximum current conditions. If positive injection current (V_{In} > V_{DD}) is greater than I_{DD}, the injection current may flow out of V_{DD} and could result in external power supply going out of regulation. Ensure external V_{DD} load will shunt current greater than maximum injection current. This will be the greatest risk when the MCU is not consuming power. Examples are: if no system clock is present, or if clock rate is very low which would reduce overall power consumption.

6 This parameter is characterized and not tested on each device.

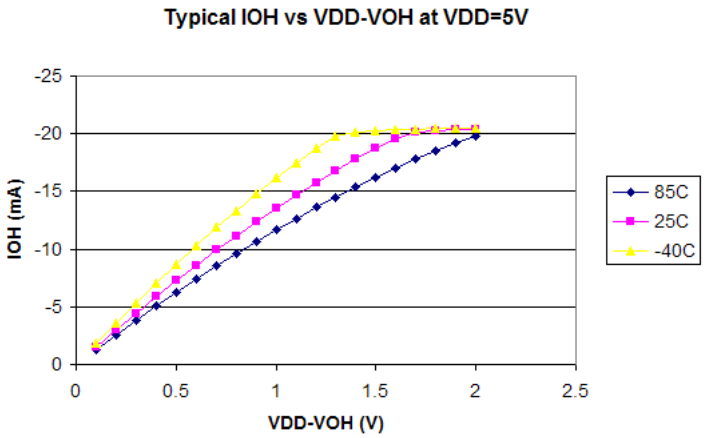


Figure 12-8. Typical IOH vs. VDD-VOH VDD = 5 V

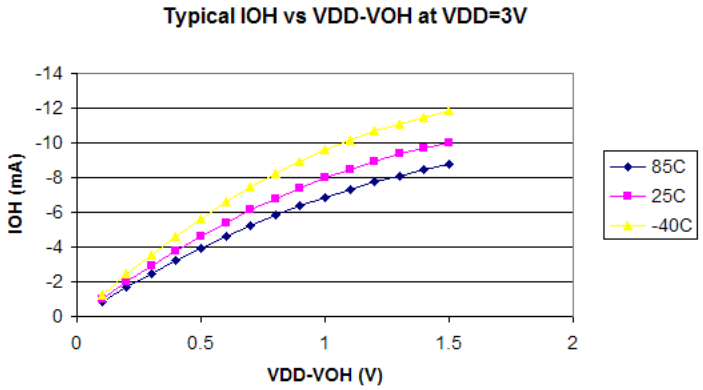


Figure 12-9. Typical IOH vs. VDD-VOH VDD = 3 V

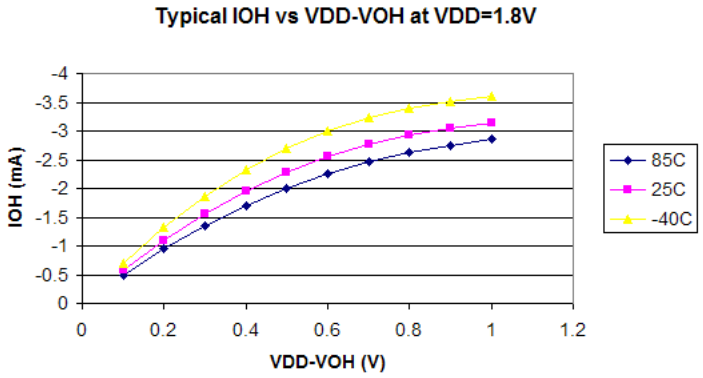


Figure 12-10. Typical IOH vs. VDD-VOH VDD = 1.8 V

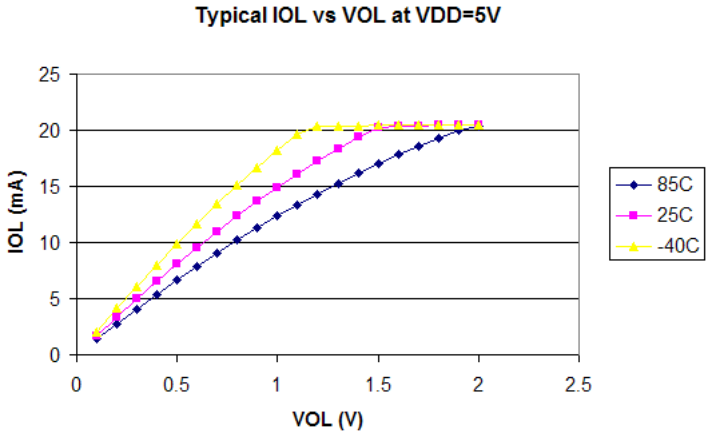


Figure 12-11. Typical I_{OL} vs. V_{OL} VDD = 5 V

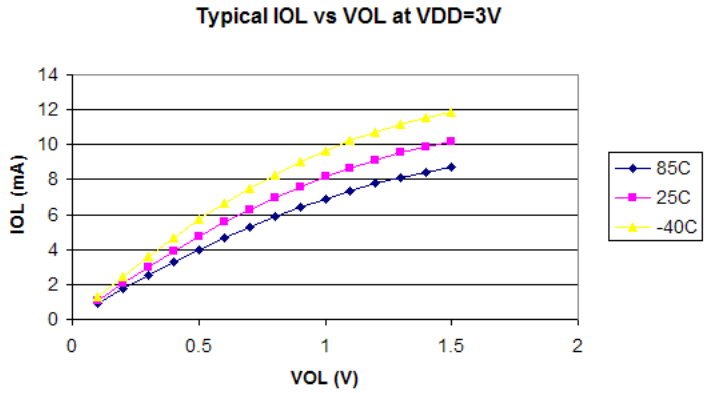


Figure 12-12. Typical I_{OL} vs. V_{OL} VDD = 3 V

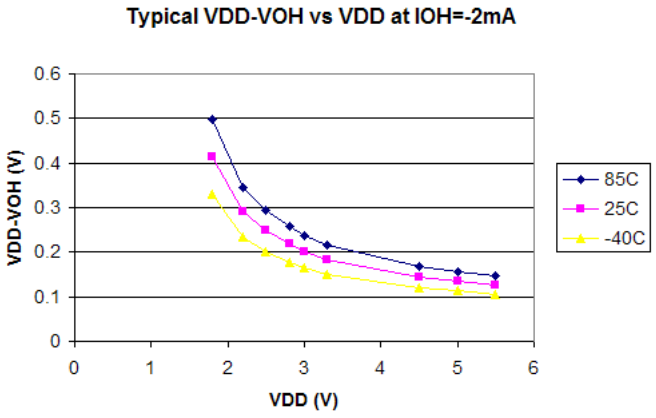


Figure 12-13. Typical V_{DD}-V_{OH} vs. V_{DD} at I_{OH}=2mA

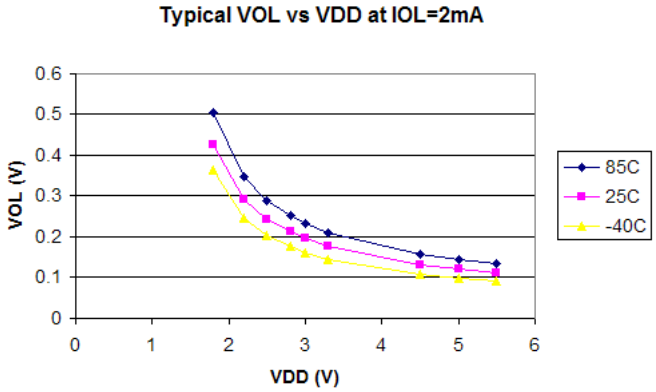


Figure 12-14. Typical V_{OL} vs. V_{DD} at I_{OL}=2mA

A.6 Supply Current Characteristics

Table A-5. Supply Current Characteristics

Parameter	Symbol	V_{DD} (V)	Typical[1]	Max[2]	Temp. (°C)
Run supply current[3] measured at (f_{Bus} = 10 MHz)	RI_{DD10}	5	5.6 mA 5.8 mA	6.5 mA	25 85
		3	4.7 mA 4.8 mA	5.5 mA	25 85
		1.8	2.3 mA 2.4 mA	3 mA	25 85
Run supply current[3] measured at (f_{Bus} = 1.25 MHz)	RI_{DD1}	5	1 mA 1.1 mA	1.5 mA	25 85
		3	0.9 mA 0.95 mA	1.2 mA	25 85
		1.8	0.6 mA 0.62 mA	0.8 mA	25 85
Stop mode supply current	SI_{DD}	5	1 µA 3 µA	2 µA 5 µA	25 85
		3	0.9 µA 2.5 µA	2 µA 5 µA	25 85
		1.8	0.7 µA 2 µA	2 µA 4 µA	25 85
Bandgap buffer adder from stop (BGBE = 1)	—	5	20 µA	30 µA	25 85
		3	20 µA	30 µA	25 85
		1.8	20 µA	30 µA	25 85

Table A-5. Supply Current Characteristics (continued)

Parameter	Symbol	V_{DD} (V)	Typical[1]	Max[2]	Temp. (°C)
ACMP adder from stop (ACME = 1)	—	5	15 μA	20 μA	25
					85
		3	15 μA	20 μA	25
					85
		1.8	15 μA	20 μA	25
					85
RTI adder from stop with 1-kHz clock source enabled[4]	—	5	300 nA	500 nA	25 85
		3	300 nA	500 nA	25 85
		1.8	300 nA	500 nA	25 85
RTI adder from stop with 32-kHz ICS internal clock source reference enabled	—	5	140 μA	165 μA	25 85
		3	140 μA	165 μA	25 85
		1.8	135 μA	160 μA	25 85
LVI adder from stop (LVDE=1 and LVDSE=1)	—	5	70 μA	85 μA	25 85
		3	70 μA	85 μA	25 85
		1.8	65 μA	80 μA	25 85

1 Typicals are measured at 25°C.

2 Maximum value is measured at the nominal V_{DD} voltage times 10% tolerance. Values given here are preliminary estimates prior to completing characterization

3 Does not include any dc loads on port pins

4 Most customers are expected to find that auto-wakeup from stop can be used instead of the higher current wait mode. Wait mode typical is 560 μA at 3 V and 422 μA at 2V with f_{Bus} = 1 MHz.

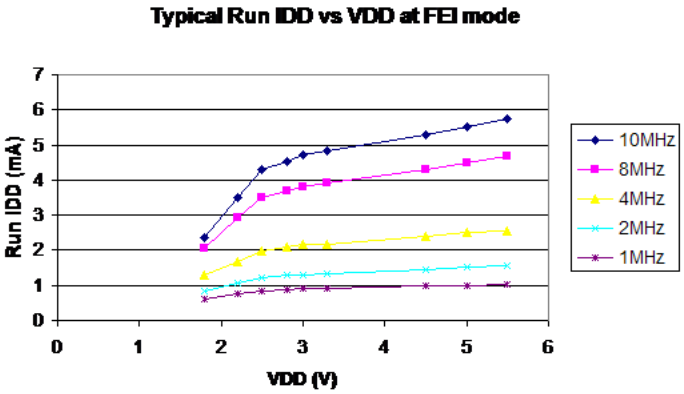


Figure 12-15. Typical Run I_{DD} vs. V_{DD} for FEI mode

A.7 Analog Comparator (ACMP) Electricals

Table A-6. Analog Comparator Electrical Specifications

Characteristic	Symbol	Min	Typical	Max	Unit
Supply voltage	V_{DD}	1.80	—	5.5	V
Analog input voltage	V_{AIN}	V_{SS} – 0.3	—	V_{DD}	V
Analog source impedance	R_{AS}	—	—	10	kΩ
Analog input offset voltage[1]	V_{AIO}	—	20	40	mV
Analog Comparator hysteresis[1]	V_H	3.0	9.0	15.0	mV
Analog Comparator bandgap reference voltage[1]	V_{BG}	1.155	1.190	1.230	V
Supply current (active)[1]	I_{DDAC}	—	20	35	µA
Analog input leakage current[1]	I_{ALKG}	—	—	1.0	µA
Analog Comparator initialization delay[1]	t_{AINIT}	—	—	1.0	µs

1 These data are characterized but not production tested. Measurements are made with the device entered STOP mode.

A.8 Internal Clock Source Characteristics

Table A-7. Internal Clock Source Specifications

Characteristic	Symbol	Min	Typ[1]	Max	Unit
Average internal reference frequency — factory trimmed at V_{DD} = 5 V and temperature = 25°C	f_{int_ft}	—	20	—	MHz
Average internal reference frequency - untrimmed	f_{int_ut}	25	31.25	41.66	kHz
Average internal reference frequency - trimmed	f_{int_t}	31.25	31.25	39.0625	kHz
DCO output frequency range - untrimmed	f_{dco_ut}	12.8	16	21.33	MHz
DCO output frequency range - trimmed	f_{dco_t}	16	16	20	MHz
Resolution of trimmed DCO output frequency at fixed voltage and temperature	$\Delta f_{dco_res_t}$	—	—	±0.2	$\%f_{dco}$
Total deviation of trimmed DCO output frequency over voltage and temperature	Δf_{dco_t}	—	—	±2	$\%f_{dco}$
FLL acquisition time[2,3]	$t_{acquire}$	—	—	1	ms

Table A-7. Internal Clock Source Specifications

Characteristic	Symbol	Min	Typ[1]	Max	Unit
Stop recovery time (FLL wakeup to previous acquired frequency) IREFSTEN=0 IREFSTEN=1	t_wakeup t_{ir_wu} t_{fll_wu}	— —	 100 86	 —	µs

1 Data in typical column was characterized at 3.0 V and 5.0 V, 25°C or is typical recommended value.

2 This parameter is characterized and not tested on each device.

3 This specification applies to any time the FLL reference source or reference divider is changed, trim value changed or changing from FLL disabled (FBILP) to FLL enabled (FEI, FBI).

A.9 AC Characteristics

This section describes ac timing characteristics for each peripheral system.

A.9.1 Control Timing

Table A-8. Control Timing

Parameter	Symbol	Min	Typical	Max	Unit
Bus frequency ($t_{cyc} = 1/f_{Bus}$)	f_{Bus}	dc	—	10	MHz
Real time interrupt internal oscillator period	t_{RTI}	700	1000	1300	µs
External $\overline{\text{RESET}}$ pulse width[1]	t_{extrst}	150	—	—	ns
KBI pulse width[2]	t_{KBIPW}	1.5 t_{cyc}	—	—	ns
KBI pulse width in stop[1]	t_{KBIPWS}	100	—	—	ns
Port rise and fall time (load = 50 pF)[3] Slew rate control disabled (PTxSE = 0) Slew rate control enabled (PTxSE = 1)	t_{Rise}, t_{Fall}	 — —	 11 35	 — —	ns

1 This is the shortest pulse that is guaranteed to pass through the pin input filter circuitry. Shorter pulses may or may not be recognized.

2 This is the minimum pulse width that is guaranteed to pass through the pin synchronization circuitry. Shorter pulses may or may not be recognized. In stop mode, the synchronizer is bypassed so shorter pulses can be recognized in that case.

3 Timing is shown with respect to 20% V_{DD} and 80% V_{DD} levels. Temperature range –40°C to 85°C.

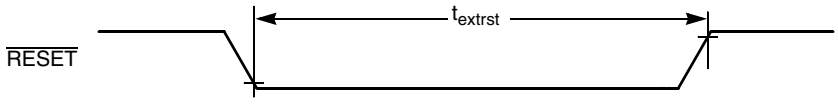


Figure A-1. Reset Timing

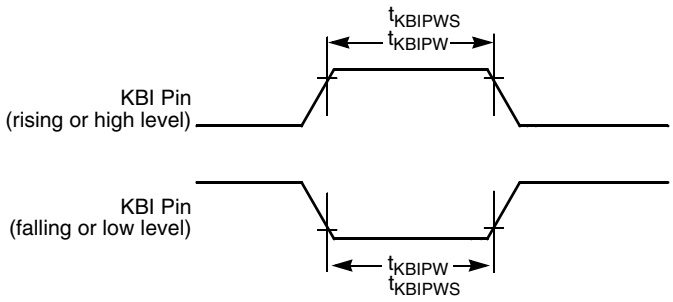


Figure A-2. KBI Pulse Width

A.10 FLASH Specifications

This section provides details about program/erase times and program-erase endurance for the FLASH memory.

For detailed information about program/erase operations, see Chapter 4, “Memory.”

Table A-9. FLASH Characteristics

Characteristic	Symbol	Min	Typical[1]	Max	Unit
Supply voltage for program/erase	V_{DD}	2.7	—	5.5	V
Program/Erase voltage	V_{PP}	11.8	12	12.2	V
V_{PP} current Program Mass erase	 I_{VPP_prog} I_{VPP_erase}	 — —	 — —	 200 100	 µA µA
Supply voltage for read operation $0 < f_{Bus} < 10$ MHz	V_{Read}	1.8	—	5.5	V
Byte program time	t_{prog}	20	—	40	µs
Mass erase time	t_{me}	500	—	—	ms
Cumulative program HV time[2]	t_{hv}	—	—	8	ms
Total cumulative HV time (total of t_{me} & t_{hv} applied to device)	t_{hv_total}	—	—	2	hours
HVEN to program setup time	t_{pgs}	10	—	—	µs
PGM/MASS to HVEN setup time	t_{nvs}	5	—	—	µs
HVEN hold time for PGM	t_{nvh}	5	—	—	µs
HVEN hold time for MASS	t_{nvh1}	100	—	—	µs
V_{PP} to PGM/MASS setup time	t_{vps}	20	—	—	ns
HVEN to V_{PP} hold time	t_{vph}	20	—	—	ns
V_{PP} rise time[3]	t_{vrs}	200	—	—	ns
Recovery time	t_{rcv}	1	—	—	µs
Program/erase endurance T_L to T_H = –40°C to + 85°C	—	1000	—	—	cycles
Data retention	t_{D_ret}	15	100	—	years

1 Typicals are measured at 25°C.

2 t_{hv} is the cumulative high voltage programming time to the same row before next erase. Same address can not be programmed more than twice before next erase.

3 Fast V_{PP} rise time may potentially trigger the ESD protection structure, which may result in over current flowing into the pad and cause permanent damage to the pad. External filtering for the V_{PP} power source is recommended. An example VPP filter is shown in Figure A-3.

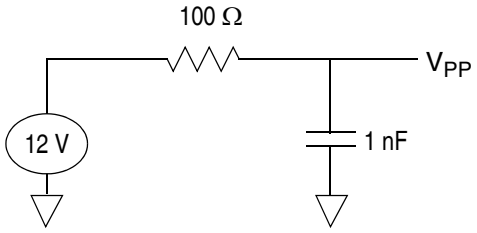
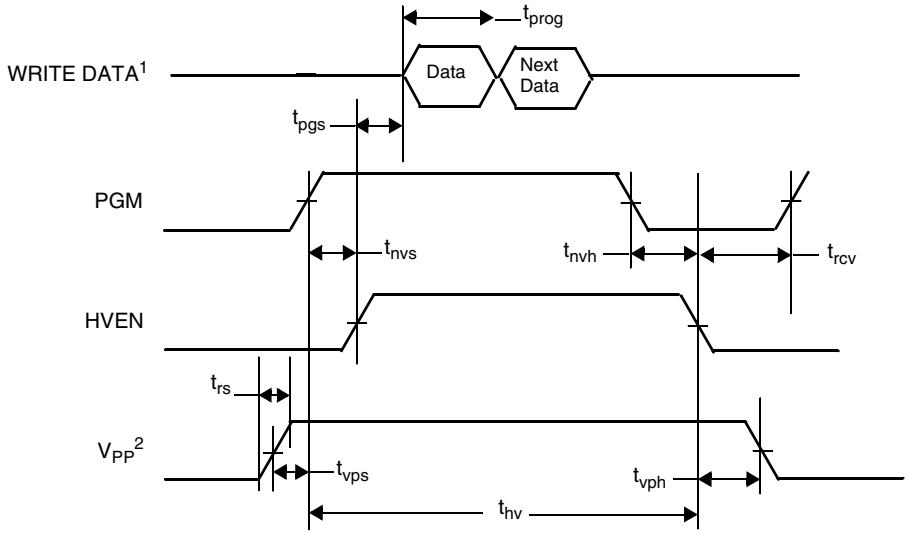
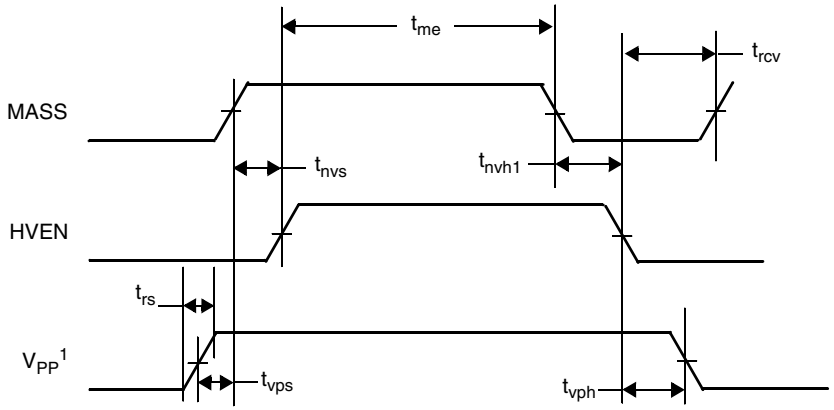


Figure A-3. Example V_{PP} Filtering



[1] Next Data applies if programming multiple bytes in a single row, reference 4.6.2, “Flash Programming Procedure”.
[2] V_{DD} must be at a valid operating voltage before voltage is applied or removed from the V_{PP} pin.

Figure A-4. Flash Program Timing



[1] V_{DD} must be at a valid operating voltage before voltage is applied or removed from the V_{PP} pin.

Figure A-5. Flash Mass Erase Timing

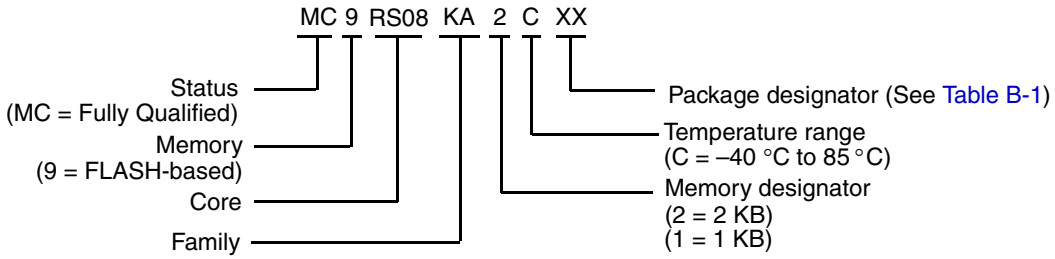
Appendix B
Ordering Information and Mechanical Drawings

B.1 Ordering Information

This section contains ordering numbers for MC9RS08KA2 Series devices. See below for an example of the device numbering system.

Table B-1. Device Numbering System

Device Number	Memory		Package		
	FLASH	RAM	Type	Designator	Document No.
MC9RS08KA2 MC9RS08KA1	2 KB 1 KB	63 bytes	6 DFN	DB	98ARL10602D
			8 PDIP	PC	98ASB42420B
			8 NB-SOIC	SC	98ASB42564B



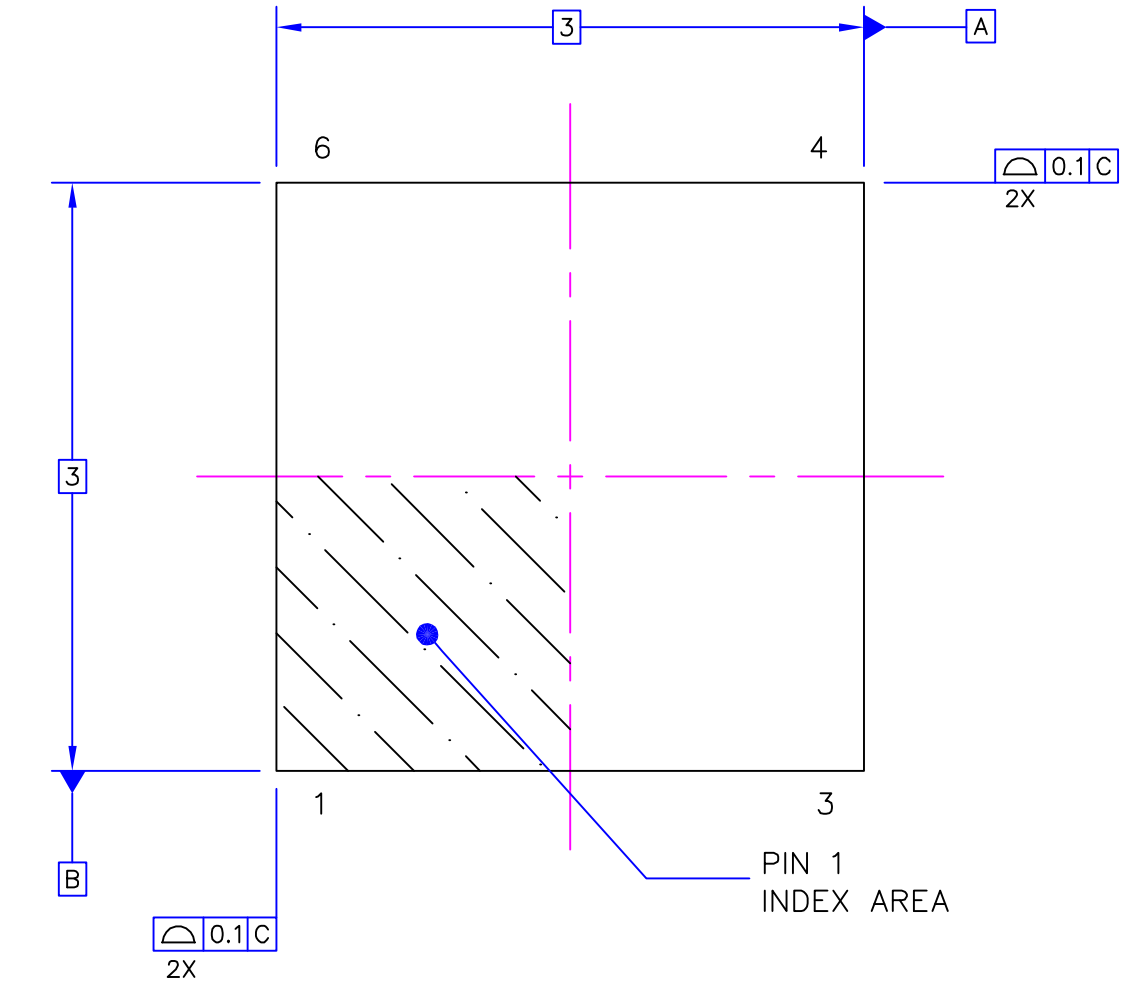
B.2 Mechanical Drawings

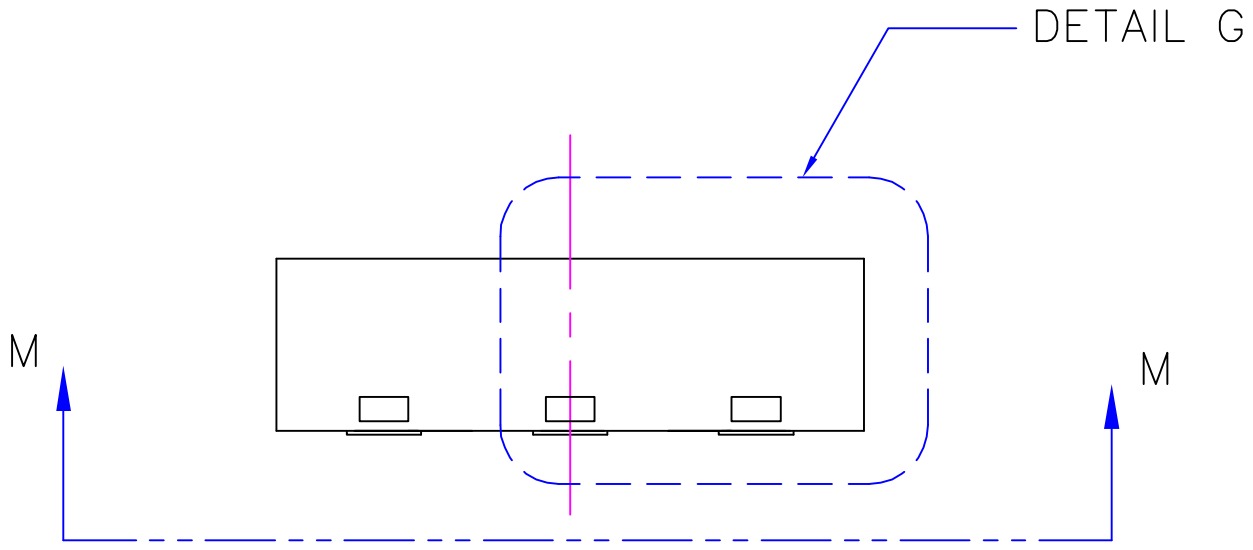
This following pages contain mechanical specifications for MC9RS08KA2 Series package options:

- 6-pin DFN (dual flat no-lead)
- 8-pin PDIP (plastic dual in-line pin)
- 8-pin NB-SOIC (narrow body small outline integrated circuit)

freescale semiconductor
© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.

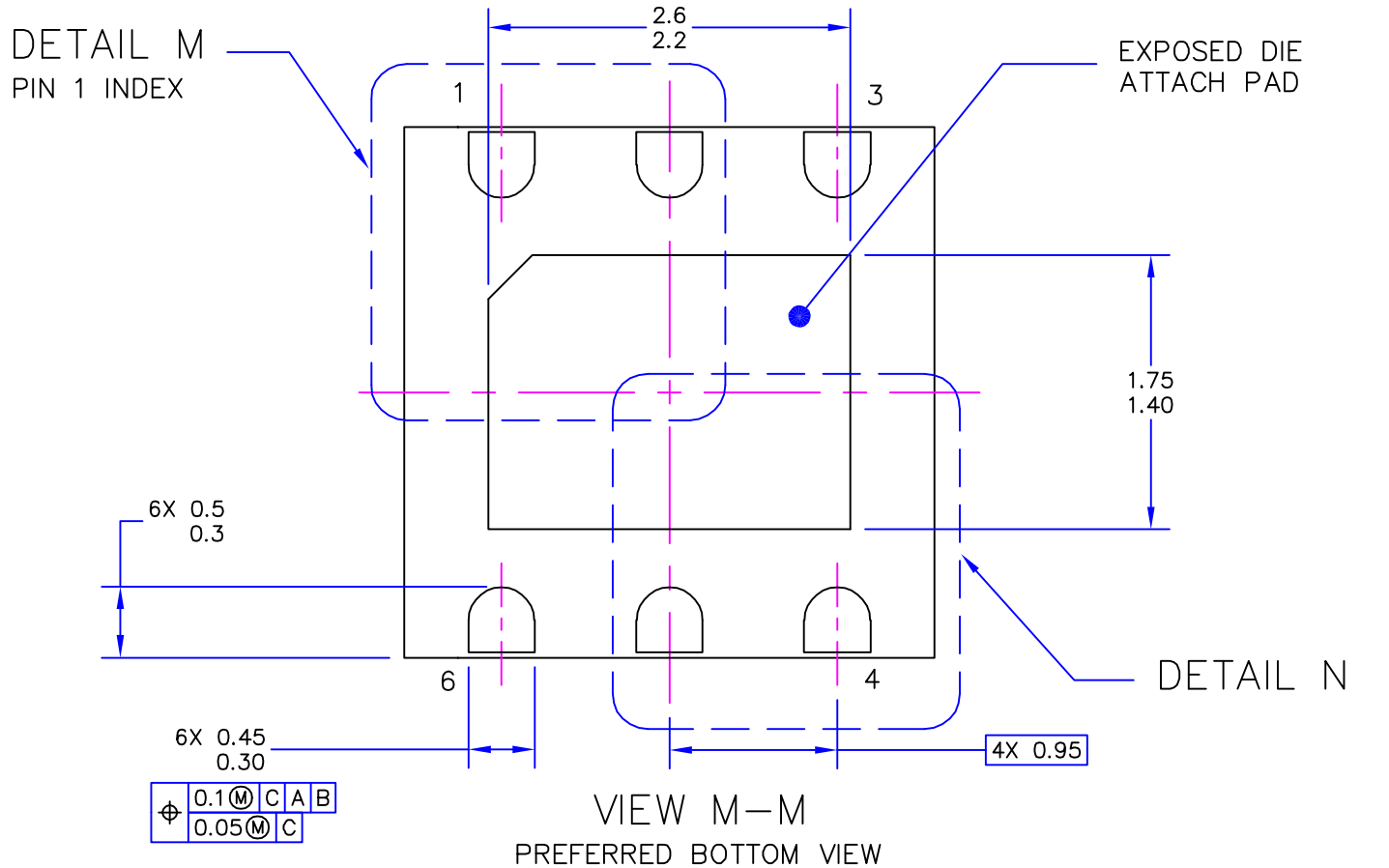
MECHANICAL OUTLINES DICTIONARY	DOCUMENT NO: 98ARL10602D
	PAGE: 1677
DO NOT SCALE THIS DRAWING	REV: C



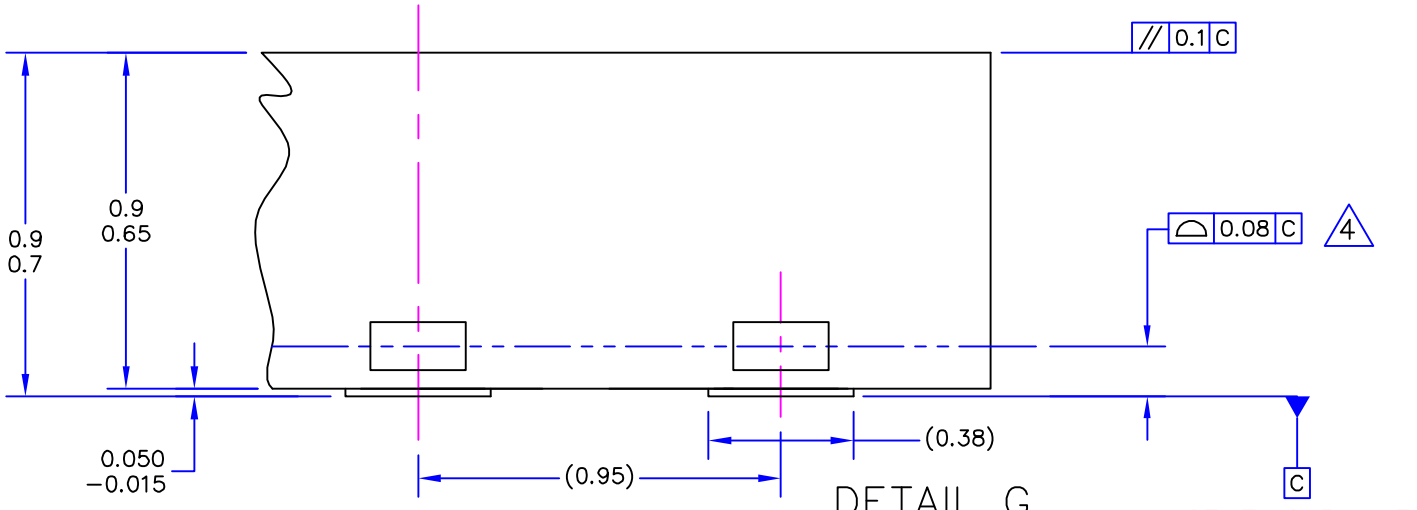


TITLE: THERMALLY ENHANCED DUAL FLAT NON-LEADED PACKAGE (DFN) 6 TERMINAL, 0.95 PITCH (3 X 3 X 0.8)	CASE NUMBER: 1677-02	
	STANDARD: FREESCALE STD	
	PACKAGE CODE: 6197	SHEET: 1 OF 6

© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.



VIEW M–M
PREFERRED BOTTOM VIEW
(NON–EXPOSED LEAD END TYPE)



DETAIL G
PREFERRED SIDE VIEW
(NON–EXPOSED LEAD END TYPE)

SEATING PLANE

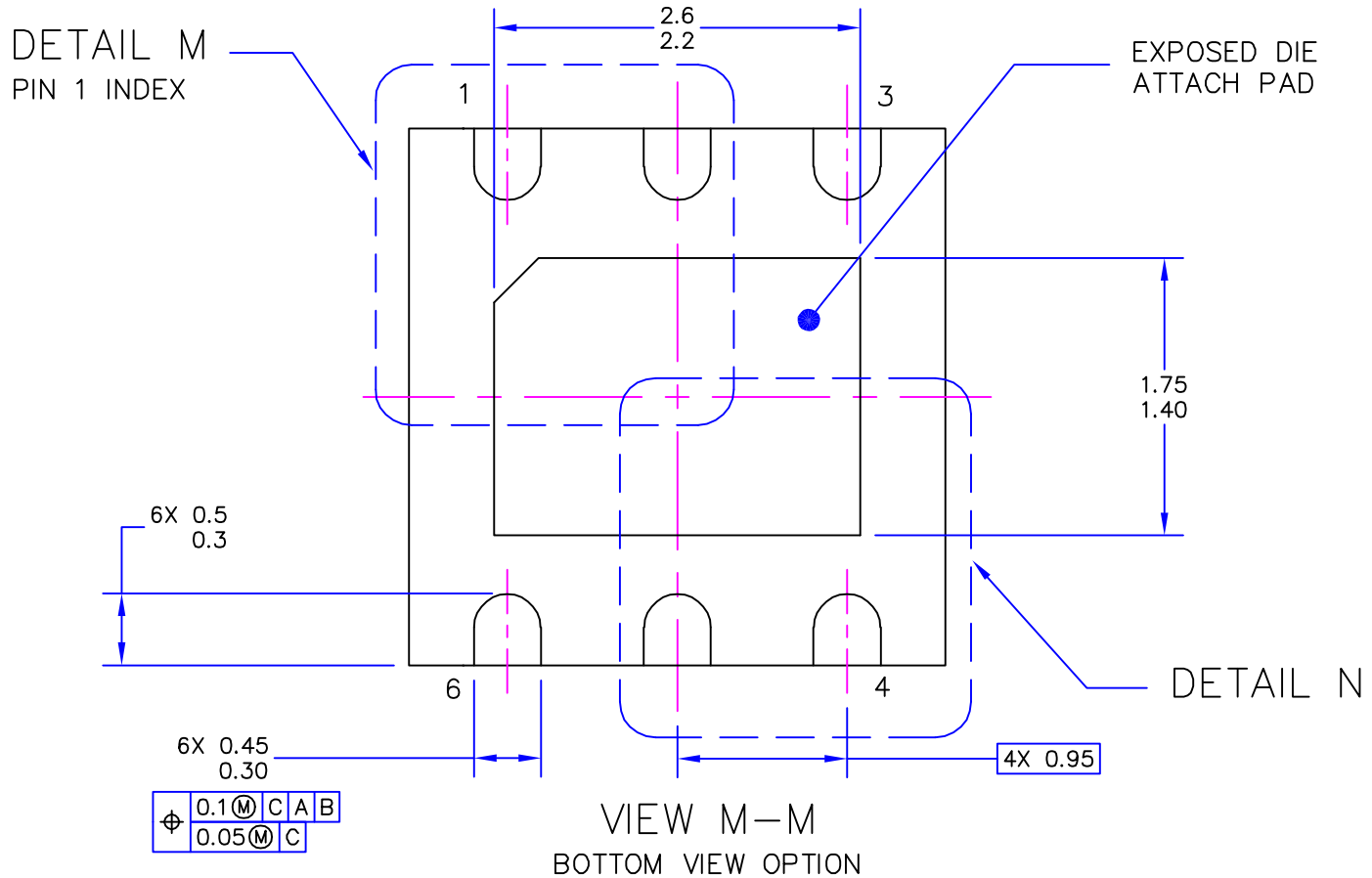
TITLE: THERMALLY ENHANCED DUAL FLAT NON–LEADED PACKAGE (DFN) 6 TERMINAL, 0.95 PITCH (3 X 3 X 0.8)

CASE NUMBER: 1677–02

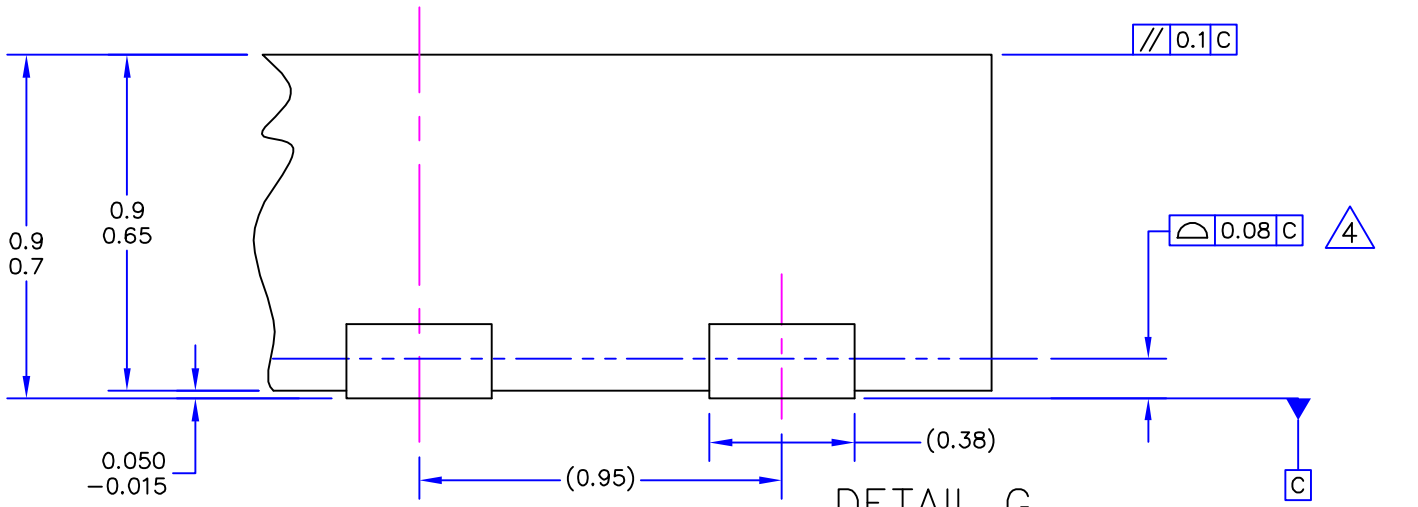
STANDARD: FREESCALE STD

PACKAGE CODE: 6197

© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.



VIEW M–M
BOTTOM VIEW OPTION
(EXPOSED LEAD END TYPE)



DETAIL G
SIDE VIEW OPTION
(EXPOSED LEAD END TYPE)

SEATING PLANE

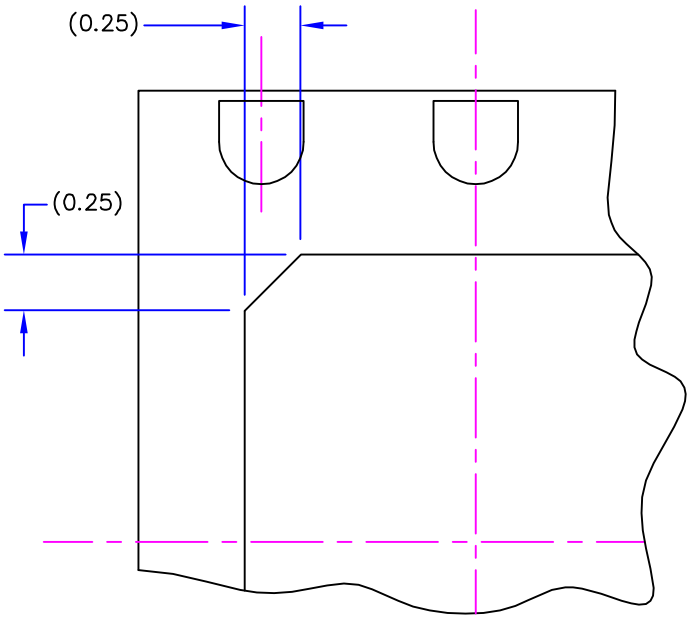
TITLE: THERMALLY ENHANCED DUAL FLAT NON–LEADED PACKAGE (DFN) 6 TERMINAL, 0.95 PITCH (3 X 3 X 0.8)

CASE NUMBER: 1677–02

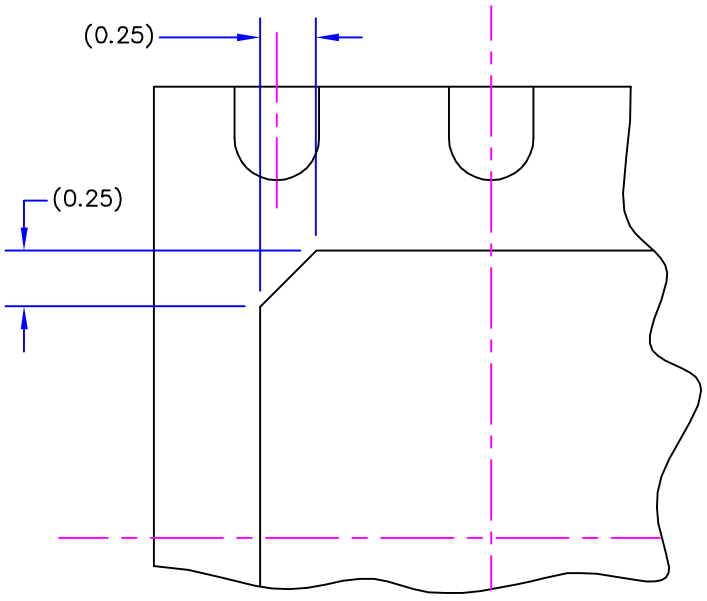
STANDARD: FREESCALE STD

PACKAGE CODE: 6197

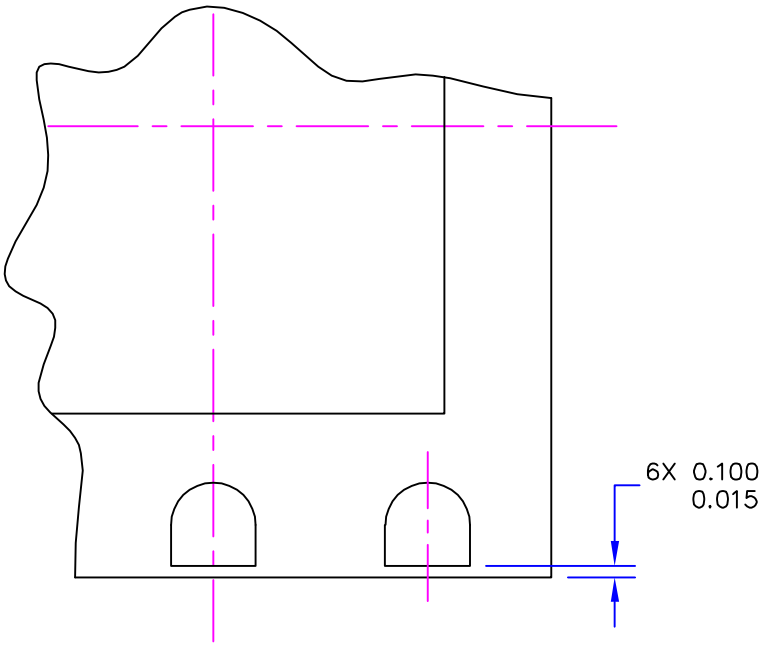
freescale semiconductor
© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.



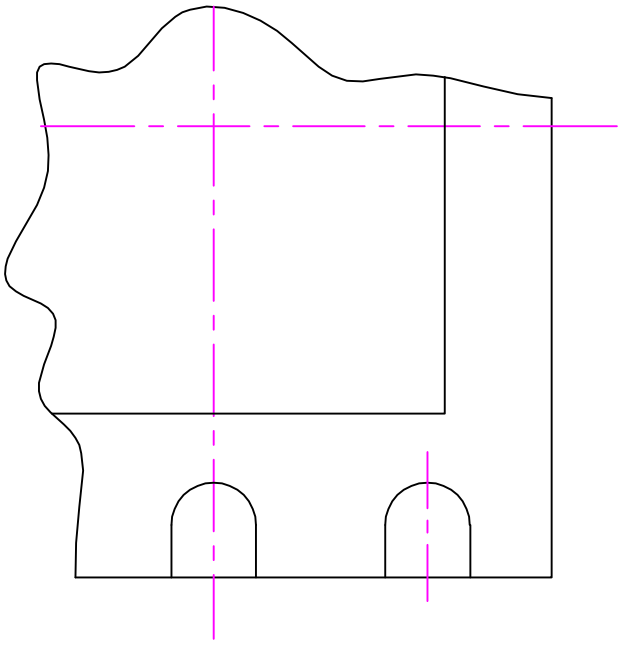
DETAIL M
PREFERRED BACKSIDE PIN 1 INDEX



DETAIL M
BACKSIDE PIN 1 INDEX OPTION



DETAIL N
PREFERRED CORNER CONFIGURATION



DETAIL N
CORNER CONFIGURATION OPTION

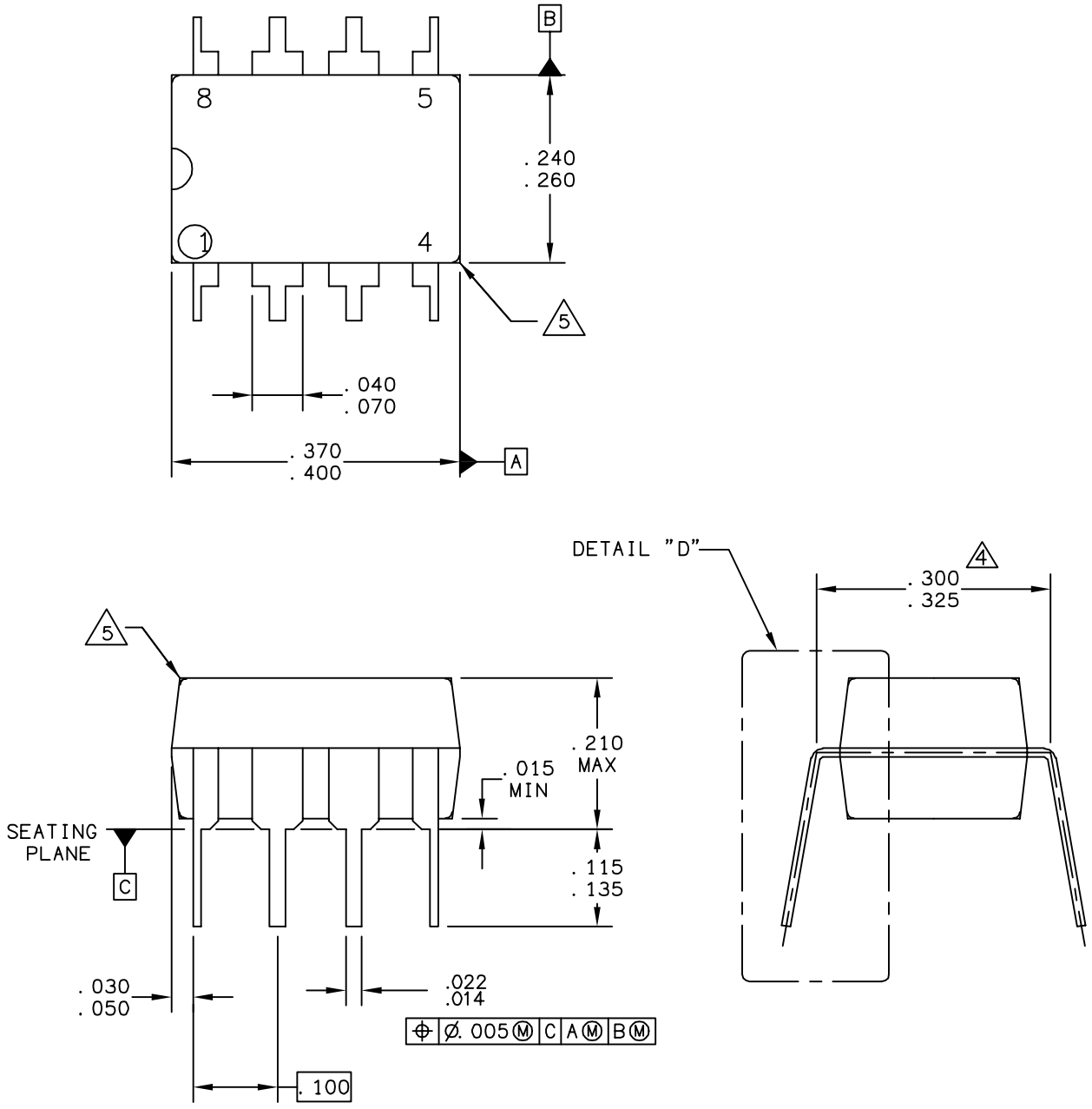
TITLE: THERMALLY ENHANCED DUAL FLAT NON-LEADED PACKAGE (DFN) 6 TERMINAL, 0.95 PITCH (3 X 3 X 0.8)

CASE NUMBER: 1677-02
STANDARD: FREESCALE STD
PACKAGE CODE: 6197
SHEET: 4 OF 6

freescale semiconductor
© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.
ELECTRONIC VERSIONS ARE UNCONTROLLED EXCEPT WHEN ACCESSED DIRECTLY FROM THE DOCUMENT CONTROL REPOSITORY. PRINTED VERSIONS ARE UNCONTROLLED EXCEPT WHEN STAMPED "CONTROLLED COPY" IN RED.

NOTES:

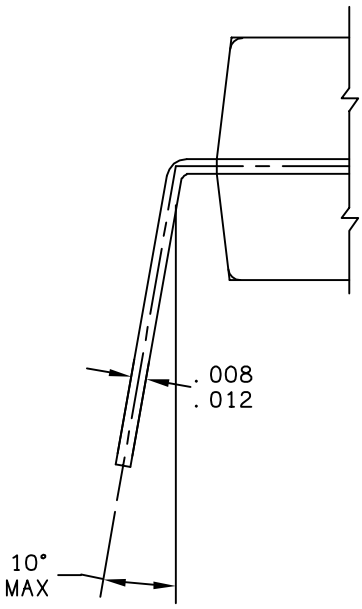
1. ALL DIMENSIONS ARE IN MILLIMETERS.
2. INTERPRET DIMENSIONS AND TOLERANCES PER ASME Y14.5M–1994.
3. THE COMPLETE JEDEC DESIGNATOR FOR THIS PACKAGE IS: HV–PDSO–N.
4. COPLANARITY APPLIES TO LEADS AND DIE ATTACH PAD.
5. MIN. METAL GAP SHOULD BE 0.2 MM.



© FREESCALE SEMICONDUCTOR, INC.
ALL RIGHTS RESERVED.

MECHANICAL OUTLINE	PRINT VERSION NOT TO SCALE

TITLE:	DOCUMENT NO: 98ASB42420B	REV: N
8 LD PDIP	CASE NUMBER: 626-06	19 MAY 2005
	STANDARD: NON-JEDEC	



DETAIL "D"

© FREESCALE SEMICONDUCTOR, INC.
ALL RIGHTS RESERVED.

MECHANICAL OUTLINE	PRINT VERSION NOT TO SCALE

TITLE:	DOCUMENT NO: 98ASB42420B	REV: N
8 LD PDIP	CASE NUMBER: 626-06	19 MAY 2005
	STANDARD: NON-JEDEC	

NOTES:

1. DIMENSIONING AND TOLERANCING PER ASME Y14.5M – 1994.
2. ALL DIMENSIONS ARE IN INCHES.
3. 626-03 TO 626-06 OBSOLETE. NEW STANDARD 626-07.
4. DIMENSION TO CENTER OF LEAD WHEN FORMED PARALLEL.
5. PACKAGE CONTOUR OPTIONAL (ROUND OR SQUARE CONERS).

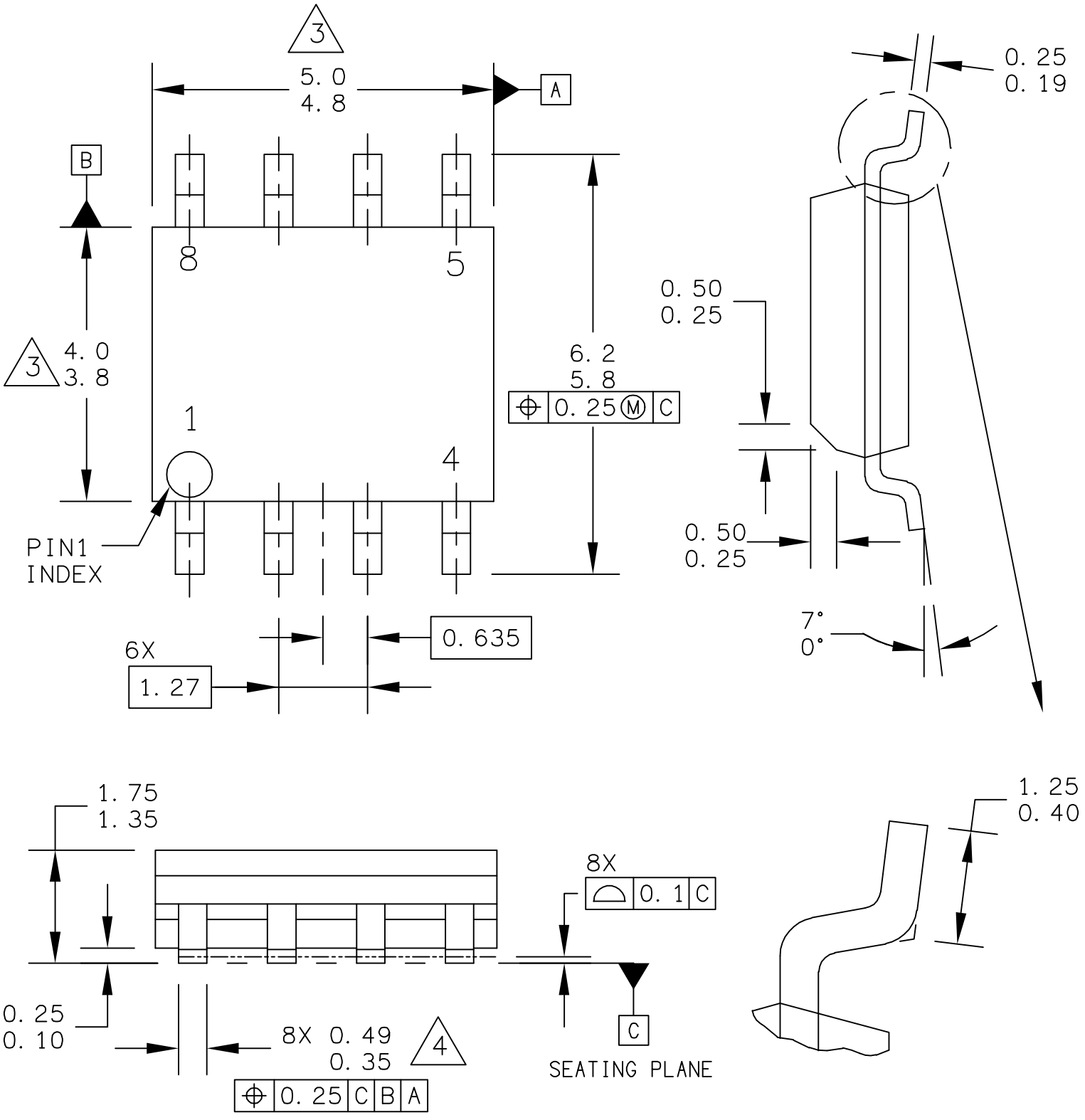
STYLE 1:

PIN	1.	AC IN	5.	GROUND
	2.	DC + IN	6.	OUTPUT
	3.	DC – IN	7.	AUXILIARY
	4.	AC IN	8.	VCC

© FREESCALE SEMICONDUCTOR, INC.
ALL RIGHTS RESERVED.

MECHANICAL OUTLINE	PRINT VERSION NOT TO SCALE

TITLE:	DOCUMENT NO: 98ASB42420B	REV: N
8 LD PDIP	CASE NUMBER: 626-06	19 MAY 2005
	STANDARD: NON-JEDEC	



© FREESCALE SEMICONDUCTOR, INC.
ALL RIGHTS RESERVED.

MECHANICAL OUTLINE	PRINT VERSION NOT TO SCALE

TITLE:	DOCUMENT NO: 98ASB42564B	REV: V
8LD SOIC NARROW BODY	CASE NUMBER: 751-07	20 NOV 2007
	STANDARD: JEDEC MS-012AA	

NOTES:

1. DIMENSIONS ARE IN MILLIMETERS.

2. DIMENSIONING AND TOLERANCING PER ASME Y14.5M–1994.

3. DIMENSION DOES NOT INCLUDE MOLD PROTRUSION. MAXIMUM MOLD PROTRUSION 0.15 PER SIDE.

4. DIMENSION DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL BE 0.127 TOTAL IN EXCESS OF THE DIMENSION AT MAXIMUM MATERIAL CONDITION.

© FREESCALE SEMICONDUCTOR, INC.
ALL RIGHTS RESERVED.

MECHANICAL OUTLINE	PRINT VERSION NOT TO SCALE

TITLE:	DOCUMENT NO: 98ASB42564B	REV: V
8LD SOIC NARROW BODY	CASE NUMBER: 751–07	20 NOV 2007
	STANDARD: JEDEC MS–012AA	

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or +1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: MC9RS08KA2
Rev. 4
12/2008

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals”, must be validated for each customer application by customer’s technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

RoHS-compliant and/or Pb-free versions of Freescale products have the functionality and electrical characteristics as their non-RoHS-compliant and/or non-Pb-free counterparts. For further information, see http://www.freescale.com or contact your Freescale sales representative.

For information on Freescale’s Environmental Products program, go to http://www.freescale.com/epp.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.

© Freescale Semiconductor, Inc. 2006-2008. All rights reserved.

freescale™
semiconductor