



High-Performance 8-Bit Microcontrollers

Z8 Encore!® 4K Series

Product Specification

PS024302-0904

PRELIMINARY



This publication is subject to replacement by a later edition. To determine whether a later edition exists, or to request copies of publications, contact:

ZiLOG Worldwide Headquarters

532 Race Street
San Jose, CA 95126
Telephone: 408.558.8500
Fax: 408.558.8300
www.ZiLOG.com

Document Disclaimer

ZiLOG is a registered trademark of ZiLOG Inc. in the United States and in other countries. All other products and/or service names mentioned herein may be trademarks of the companies with which they are associated.

©2004 by ZiLOG, Inc. All rights reserved. Information in this publication concerning the devices, applications, or technology described is intended to suggest possible uses and may be superseded. ZILOG, INC. DOES NOT ASSUME LIABILITY FOR OR PROVIDE A REPRESENTATION OF ACCURACY OF THE INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED IN THIS DOCUMENT. ZiLOG ALSO DOES NOT ASSUME LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT RELATED IN ANY MANNER TO USE OF INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED HEREIN OR OTHERWISE. Devices sold by ZiLOG, Inc. are covered by warranty and limitation of liability provisions appearing in the ZiLOG, Inc. Terms and Conditions of Sale. ZiLOG, Inc. makes no warranty of merchantability or fitness for any purpose Except with the express written approval of ZiLOG, use of information, devices, or technology as critical components of life support systems is not authorized. No licenses are conveyed, implicitly or otherwise, by this document under any intellectual property rights.



Table of Contents

| | |
|--|------------|
| List of Figures | x |
| List of Tables | xii |
| Overview | 1 |
| Features | 1 |
| Part Selection Guide | 2 |
| Block Diagram | 2 |
| CPU and Peripheral Overview | 4 |
| eZ8 CPU Features | 4 |
| General Purpose I/O | 4 |
| Flash Controller | 4 |
| Internal Precision Oscillator | 5 |
| 10-Bit Analog-to-Digital Converter | 5 |
| UART | 5 |
| Timers | 5 |
| Interrupt Controller | 5 |
| Reset Controller | 5 |
| On-Chip Debugger | 5 |
| Pin Description | 7 |
| Overview | 7 |
| Available Packages | 7 |
| Pin Configurations | 7 |
| Signal Descriptions | 9 |
| Pin Characteristics | 10 |
| Address Space | 13 |
| Overview | 13 |
| Register File | 13 |
| Program Memory | 14 |
| Data Memory | 15 |
| Flash Information Area | 15 |
| Register Map | 16 |
| Reset, STOP Mode Recovery and Low Voltage Detection | 20 |
| Overview | 20 |
| Reset Types | 20 |



| | |
|---|-----------|
| Reset Sources | 21 |
| Power-On Reset | 22 |
| Voltage Brown-Out Reset | 22 |
| Watch-Dog Timer Reset | 23 |
| External Reset Input | 24 |
| External Reset Indicator | 24 |
| STOP Mode Recovery | 24 |
| STOP Mode Recovery Using Watch-Dog Timer Time-Out | 25 |
| STOP Mode Recovery Using a GPIO Port Pin Transition | 25 |
| STOP Mode Recovery Using the External RESET Pin | 26 |
| Reset Register Definitions | 26 |
| Low-Power Modes | 28 |
| Overview | 28 |
| STOP Mode | 28 |
| HALT Mode | 29 |
| Peripheral-Level Power Control | 29 |
| Power Control Register Definitions | 29 |
| General-Purpose I/O | 31 |
| Overview | 31 |
| GPIO Port Availability By Device | 31 |
| Architecture | 32 |
| GPIO Alternate Functions | 32 |
| Shared Debug Pin | 33 |
| 5V Tolerance | 33 |
| External Clock Setup | 33 |
| GPIO Interrupts | 38 |
| GPIO Control Register Definitions | 38 |
| Port A–C Address Registers | 39 |
| Port A–C Control Registers | 39 |
| Port A–C Data Direction Sub-Registers | 40 |
| Port A–C Alternate Function Sub-Registers | 40 |
| Port A–C Input Data Registers | 44 |
| Port A–C Output Data Register | 45 |
| Interrupt Controller | 46 |
| Overview | 46 |
| Interrupt Vector Listing | 46 |
| Architecture | 48 |



| | |
|---|-----------|
| Operation | 48 |
| Master Interrupt Enable | 48 |
| Interrupt Vectors and Priority | 49 |
| Interrupt Assertion | 49 |
| Software Interrupt Assertion | 50 |
| Interrupt Control Register Definitions | 50 |
| Interrupt Request 0 Register | 50 |
| Interrupt Request 1 Register | 51 |
| Interrupt Request 2 Register | 52 |
| IRQ0 Enable High and Low Bit Registers | 52 |
| IRQ1 Enable High and Low Bit Registers | 54 |
| IRQ2 Enable High and Low Bit Registers | 55 |
| Interrupt Edge Select Register | 56 |
| Interrupt Control Register | 56 |
| Timers | 58 |
| Overview | 58 |
| Architecture | 58 |
| Operation | 59 |
| Timer Operating Modes | 59 |
| Reading the Timer Count Values | 71 |
| Timer Pin Signal Operation | 71 |
| Timer Control Register Definitions | 72 |
| Timer 0–1 High and Low Byte Registers | 72 |
| Timer Reload High and Low Byte Registers | 72 |
| Timer 0-1 PWM High and Low Byte Registers | 73 |
| Timer 0–1 Control Registers | 74 |
| Watch-Dog Timer | 79 |
| Overview | 79 |
| Operation | 79 |
| Watch-Dog Timer Refresh | 80 |
| Watch-Dog Timer Time-Out Response | 80 |
| Watch-Dog Timer Reload Unlock Sequence | 81 |
| Watch-Dog Timer Control Register Definitions | 82 |
| Watch-Dog Timer Control Register | 82 |
| Watch-Dog Timer Reload Upper, High and Low Byte Registers | 82 |
| UART | 84 |
| Overview | 84 |
| Architecture | 84 |



| | |
|---|------------|
| Operation | 85 |
| Data Format | 85 |
| Transmitting Data using the Polled Method | 86 |
| Transmitting Data using the Interrupt-Driven Method | 87 |
| Receiving Data using the Polled Method | 88 |
| Receiving Data using the Interrupt-Driven Method | 89 |
| Clear To Send (CTS) Operation | 90 |
| MULTIPROCESSOR (9-bit) Mode | 90 |
| External Driver Enable | 91 |
| UART Interrupts | 92 |
| UART Baud Rate Generator | 94 |
| UART Control Register Definitions | 95 |
| UART Transmit Data Register | 95 |
| UART Receive Data Register | 96 |
| UART Status 0 Register | 96 |
| UART Status 1 Register | 97 |
| UART Control 0 and Control 1 Registers | 98 |
| UART Address Compare Register | 100 |
| UART Baud Rate High and Low Byte Registers | 101 |
| Infrared Encoder/Decoder | 103 |
| Overview | 103 |
| Architecture | 103 |
| Operation | 103 |
| Transmitting IrDA Data | 104 |
| Receiving IrDA Data | 105 |
| Infrared Encoder/Decoder Control Register Definitions | 106 |
| Analog-to-Digital Converter | 107 |
| Overview | 107 |
| Architecture | 107 |
| Operation | 108 |
| Data Format | 108 |
| Automatic Powerdown | 109 |
| Single-Shot Conversion | 109 |
| Continuous Conversion | 110 |
| ADC Control Register Definitions | 111 |
| ADC Control Register 0 | 111 |
| ADC Control/Status Register 1 | 112 |
| ADC Data High Byte Register | 113 |
| ADC Data Low Bits Register | 113 |



| | |
|--|------------|
| Flash Memory | 115 |
| Overview | 115 |
| Flash Information Area | 116 |
| Operation | 117 |
| Flash Operation Timing Using the Flash Frequency Registers | 119 |
| Flash Code Protection Against External Access | 119 |
| Flash Code Protection Against Accidental Program and Erasure | 119 |
| Byte Programming | 121 |
| Page Erase | 121 |
| Mass Erase | 121 |
| Flash Controller Bypass | 122 |
| Flash Control Register Definitions | 122 |
| Flash Control Register | 122 |
| Flash Status Register | 123 |
| Flash Page Select Register | 123 |
| Flash Sector Protect Register | 124 |
| Flash Frequency High and Low Byte Registers | 125 |
| Flash Option Bits | 126 |
| Overview | 126 |
| Operation | 126 |
| Option Bit Configuration By Reset | 126 |
| Option Bit Types | 126 |
| Flash Option Bit Control Register Definitions | 127 |
| Trim Bit Address Register | 127 |
| Trim Bit Data Register | 127 |
| Flash Option Bit Address Space | 128 |
| Flash Program Memory Address 0000H | 128 |
| Flash Program Memory Address 0001H | 129 |
| Trim Bit Address Space | 129 |
| Trim Bit Address 0000H - Reserved | 129 |
| Trim Bit Address 0001H - Reserved | 129 |
| Trim Bit Address 0002H | 129 |
| Trim Bit Address 0003H - Reserved | 130 |
| Trim Bit Address 0004H - Reserved | 130 |
| On-Chip Debugger | 131 |
| Overview | 131 |
| Architecture | 131 |
| Operation | 132 |



| | |
|---|------------|
| OCD Interface | 132 |
| DEBUG Mode | 133 |
| OCD Data Format | 134 |
| OCD Auto-Baud Detector/Generator | 134 |
| OCD Serial Errors | 135 |
| Breakpoints | 135 |
| Runtime Counter | 135 |
| On-Chip Debugger Commands | 136 |
| On-Chip Debugger Control Register Definitions | 140 |
| OCD Control Register | 140 |
| OCD Status Register | 142 |
| Oscillator Control | 143 |
| Overview | 143 |
| Operation | 143 |
| System Clock Selection | 143 |
| Oscillator Control Register Definitions | 144 |
| Internal Precision Oscillator | 146 |
| Overview | 146 |
| Operation | 146 |
| eZ8 CPU Instruction Set | 147 |
| Assembly Language Programming Introduction | 147 |
| Assembly Language Syntax | 148 |
| eZ8 CPU Instruction Notation | 148 |
| eZ8 CPU Instruction Classes | 150 |
| eZ8 CPU Instruction Summary | 155 |
| Opcode Maps | 163 |
| Electrical Characteristics | 167 |
| Absolute Maximum Ratings | 167 |
| DC Characteristics | 168 |
| AC Characteristics | 170 |
| On-Chip Peripheral AC and DC Electrical Characteristics | 171 |
| General Purpose I/O Port Input Data Sample Timing | 173 |
| General Purpose I/O Port Output Timing | 175 |
| On-Chip Debugger Timing | 176 |
| UART Timing | 177 |
| Packaging | 179 |



| | |
|-------------------------------------|------------|
| Ordering Information | 185 |
| Customer Feedback Form | 193 |
| Index | 195 |



List of Figures

| | |
|--|-----|
| Figure 1. 4K Series Block Diagram | 3 |
| Figure 2. Z8F04x3, Z8F02x3, and Z8F01x3 in 8-Pin SOIC or PDIP Package .. | 8 |
| Figure 3. Z8F04x3, Z8F02x3, and Z8F01x3 in 20-Pin SOIC, SSOP or PDIP Package | 8 |
| Figure 4. Z8F04x3, Z8F02x3, and Z8F01x3k in 28-Pin SOIC, SSOP or PDIP Package | 8 |
| Figure 5. Power-On Reset Operation | 22 |
| Figure 6. Voltage Brown-Out Reset Operation | 23 |
| Figure 7. GPIO Port Pin Block Diagram | 32 |
| Figure 8. Interrupt Controller Block Diagram | 48 |
| Figure 9. Timer Block Diagram | 59 |
| Figure 10. UART Block Diagram | 85 |
| Figure 11. UART Asynchronous Data Format without Parity | 86 |
| Figure 12. UART Asynchronous Data Format with Parity | 86 |
| Figure 13. UART Asynchronous MULTIPROCESSOR Mode Data Format ... | 90 |
| Figure 14. UART Driver Enable Signal Timing (shown with 1 Stop Bit and Parity) | 92 |
| Figure 15. UART Receiver Interrupt Service Routine Flow | 94 |
| Figure 16. Infrared Data Communication System Block Diagram | 103 |
| Figure 17. Infrared Data Transmission | 104 |
| Figure 18. IrDA Data Reception | 105 |
| Figure 19. Analog-to-Digital Converter Block Diagram | 108 |
| Figure 20. Flash Memory Arrangement | 116 |
| Figure 21. Flash Controller Operation Flow Chart | 118 |
| Figure 22. On-Chip Debugger Block Diagram | 131 |
| Figure 23. Interfacing the On-Chip Debugger's DBG Pin with an RS-232 Interface (1) | 132 |
| Figure 24. Interfacing the On-Chip Debugger's DBG Pin with an RS-232 Interface (2) | 133 |



| | |
|---|-----|
| Figure 25. OCD Data Format | 134 |
| Figure 26. Opcode Map Cell Description | 163 |
| Figure 27. First Opcode Map | 165 |
| Figure 28. Second Opcode Map after 1FH | 166 |
| Figure 29. ICC Versus System Clock Frequency | 169 |
| Figure 30. Port Input Sample Timing | 174 |
| Figure 31. GPIO Port Output Timing | 175 |
| Figure 32. On-Chip Debugger Timing | 176 |
| Figure 33. UART Timing With CTS | 177 |
| Figure 34. UART Timing Without CTS | 178 |
| Figure 35. 8-Pin Plastic Dual Inline Package (PDIP) | 179 |
| Figure 36. 8-Pin Small Outline Integrated Circuit Package (SOIC) | 180 |
| Figure 37. 20-Pin Plastic Dual Inline Package (PDIP) | 181 |
| Figure 38. 20-Pin Small Shrink Outline Package (SSOP) | 182 |
| Figure 39. 28-Pin Plastic Dual Inline Package (PDIP) | 183 |
| Figure 40. 28-Pin Small Outline Integrated Circuit Package (SOIC) | 184 |



List of Tables

| | |
|---|----|
| Table 1. 4K Series Family Part Selection Guide | 2 |
| Table 2. 4K Series Package Options | 7 |
| Table 3. Signal Descriptions. | 9 |
| Table 4. Pin Characteristics (20- and 28-pin Devices). | 11 |
| Table 5. Pin Characteristics (8-Pin Devices) | 12 |
| Table 6. 4K Series Program Memory Maps | 14 |
| Table 7. 4K Series Flash Memory Information Area Map | 15 |
| Table 8. Register File Address Map. | 16 |
| Table 9. Reset and STOP Mode Recovery Characteristics and Latency. | 20 |
| Table 10. Reset Sources and Resulting Reset Type | 21 |
| Table 11. STOP Mode Recovery Sources and Resulting Action | 25 |
| Table 12. Reset Status Register (RSTSTAT). | 26 |
| Table 13. Power Control Register 0 (PWRCTL0). | 30 |
| Table 14. Port Availability by Device and Package Type. | 31 |
| Table 15. Port Alternate Function Mapping (Non 8-Pin Parts). | 34 |
| Table 16. Port Alternate Function Mapping (8-Pin Parts). | 37 |
| Table 17. GPIO Port Registers and Sub-Registers | 38 |
| Table 18. Port A–C GPIO Address Registers (PxADDR). | 39 |
| Table 19. Port A–C Control Registers (PxCTL) | 40 |
| Table 20. Port A–C Data Direction Sub-Registers (PxDD) | 40 |
| Table 21. Port A–C Alternate Function Sub-Registers (PxAF). | 41 |
| Table 22. Port A–C Output Control Sub-Registers (PxOC) | 41 |
| Table 23. Port A–C High Drive Enable Sub-Registers (PxHDE) | 42 |
| Table 24. Port A–C STOP Mode Recovery Source Enable Sub-Registers (PxSMRE) | 42 |
| Table 25. Port A–C Pull-Up Enable Sub-Registers (PxPUE). | 43 |
| Table 26. Port A–C Alternate Function Set 1 Sub-Registers (PxAFS1). | 43 |
| Table 27. Port A–C Alternate Function Set 2 Sub-Registers (PxAFS2). | 44 |



| | |
|--|----|
| Table 28. Port A–C Input Data Registers (PxIN) | 44 |
| Table 29. Port A–C Output Data Register (PxOUT). | 45 |
| Table 30. Trap and Interrupt Vectors in Order of Priority | 47 |
| Table 31. Interrupt Request 0 Register (IRQ0) | 51 |
| Table 32. Interrupt Request 1 Register (IRQ1) | 52 |
| Table 33. Interrupt Request 2 Register (IRQ2) | 52 |
| Table 34. IRQ0 Enable and Priority Encoding | 53 |
| Table 35. IRQ0 Enable High Bit Register (IRQ0ENH) | 53 |
| Table 36. IRQ0 Enable Low Bit Register (IRQ0ENL). | 53 |
| Table 37. IRQ1 Enable and Priority Encoding | 54 |
| Table 38. IRQ1 Enable High Bit Register (IRQ1ENH) | 54 |
| Table 39. IRQ1 Enable Low Bit Register (IRQ1ENL). | 54 |
| Table 40. IRQ2 Enable and Priority Encoding | 55 |
| Table 41. IRQ2 Enable High Bit Register (IRQ2ENH) | 55 |
| Table 42. IRQ2 Enable Low Bit Register (IRQ2ENL). | 55 |
| Table 43. Interrupt Edge Select Register (IRQES). | 56 |
| Table 44. Interrupt Control Register (IRQCTL) | 56 |
| Table 45. Timer 0–1 High Byte Register (TxH) | 72 |
| Table 46. Timer 0–1 Low Byte Register (TxL) | 72 |
| Table 47. Timer 0–1 Reload High Byte Register (TxRH). | 73 |
| Table 48. Timer 0–1 Reload Low Byte Register (TxRL). | 73 |
| Table 49. Timer 0–1 PWM High Byte Register (TxPWMH) | 73 |
| Table 50. Timer 0–1 Control Register 0 (TxCTL0). | 74 |
| Table 51. Timer 0–1 PWM Low Byte Register (TxPWML). | 74 |
| Table 52. Timer 0–1 Control Register 1 (TxCTL1). | 75 |
| Table 53. Watch-Dog Timer Approximate Time-Out Delays | 80 |
| Table 54. Watch-Dog Timer Control Register (WDTCTL) | 82 |
| Table 55. Watch-Dog Timer Reload Upper Byte Register (WDTU). | 83 |
| Table 56. Watch-Dog Timer Reload High Byte Register (WDTH) | 83 |
| Table 57. Watch-Dog Timer Reload Low Byte Register (WDTL). | 83 |



| | |
|--|-----|
| Table 58. UART Transmit Data Register (U0TXD) | 95 |
| Table 59. UART Receive Data Register (U0RXD). | 96 |
| Table 60. UART Status 0 Register (U0STAT0) | 96 |
| Table 61. UART Status 1 Register (U0STAT1) | 97 |
| Table 62. UART Control 0 Register (U0CTL0). | 98 |
| Table 63. UART Control 1 Register (U0CTL1). | 99 |
| Table 64. UART Address Compare Register (U0ADDR). | 101 |
| Table 65. UART Baud Rate High Byte Register (U0BRH). | 101 |
| Table 66. UART Baud Rate Low Byte Register (U0BRL) | 101 |
| Table 67. UART Baud Rates | 102 |
| Table 68. ADC Control Register 0 (ADCCTL0) | 111 |
| Table 69. ADC Control/Status Register 1 (ADCCTL1). | 112 |
| Table 70. ADC Data High Byte Register (ADCD_H) | 113 |
| Table 71. ADC Data Low Bits Register (ADCD_L). | 113 |
| Table 72. 4K Series Flash Memory Configurations | 115 |
| Table 73. Flash Code Protection Using the Flash Option Bits | 120 |
| Table 74. Flash Control Register (FCTL) | 122 |
| Table 75. Flash Status Register (FSTAT) | 123 |
| Table 76. Flash Page Select Register (FPS) | 124 |
| Table 77. Flash Sector Protect Register (FPROT). | 124 |
| Table 78. Flash Frequency High Byte Register (FFREQH) | 125 |
| Table 79. Flash Frequency Low Byte Register (FFREQL). | 125 |
| Table 80. Trim Bit Address Register (TRMADR) | 127 |
| Table 81. Trim Bit Data Register (TRMDR) | 127 |
| Table 82. Flash Option Bits at Program Memory Address 0000H | 128 |
| Table 83. Flash Options Bits at Program Memory Address 0001H | 129 |
| Table 84. Trim Option Bits at 0002H (TIPO) | 129 |
| Table 85. OCD Baud-Rate Limits. | 134 |
| Table 86. OCD Control Register (OCDCTL) | 141 |
| Table 87. OCD Status Register (OCDSTAT) | 142 |



| | |
|--|-----|
| Table 88. Oscillator Configuration and Selection | 143 |
| Table 89. Oscillator Control Register (OSCCTL) | 144 |
| Table 90. Assembly Language Syntax Example 1 | 148 |
| Table 91. Assembly Language Syntax Example 2 | 148 |
| Table 92. Notational Shorthand | 149 |
| Table 93. Additional Symbols | 150 |
| Table 94. Arithmetic Instructions | 151 |
| Table 95. Bit Manipulation Instructions | 152 |
| Table 96. Block Transfer Instructions | 152 |
| Table 97. CPU Control Instructions | 152 |
| Table 98. Logical Instructions | 153 |
| Table 99. Load Instructions | 153 |
| Table 100. Program Control Instructions | 154 |
| Table 101. Rotate and Shift Instructions | 154 |
| Table 102. eZ8 CPU Instruction Summary | 155 |
| Table 103. Opcode Map Abbreviations | 164 |
| Table 104. Absolute Maximum Ratings | 167 |
| Table 105. DC Characteristics | 168 |
| Table 106. AC Characteristics | 170 |
| Table 107. Power-On Reset and Voltage Brown-Out Electrical Characteristics and Timing | 171 |
| Table 108. Flash Memory Electrical Characteristics and Timing | 172 |
| Table 109. Watch-Dog Timer Electrical Characteristics and Timing | 172 |
| Table 110. Analog-to-Digital Converter Electrical Characteristics and Timing | 173 |
| Table 111. GPIO Port Input Timing | 174 |
| Table 112. GPIO Port Output Timing | 175 |
| Table 113. On-Chip Debugger Timing | 176 |
| Table 114. UART Timing With CTS | 177 |
| Table 115. UART Timing Without CTS | 178 |



Overview

The Z8 Encore![®] MCU family of products are the first in a line of ZiLOG[®] microcontroller products based upon the 8-bit eZ8 CPU. The Z8 Encore![®] 4K Series products, hereafter referred to collectively the Z8 Encore![®] or the PartName, expand upon ZiLOG's extensive line of 8-bit microcontrollers. The Flash in-circuit programming capability allows for faster development time and program changes in the field. The new eZ8 CPU is upward compatible with existing Z8[®] instructions. The rich peripheral set of the PartName makes it suitable for a variety of applications including motor control, security systems, home appliances, personal electronic devices, and sensors.

Features

- 5MHz eZ8 CPU
- 1KB, 2KB or 4KB Flash memory with in-circuit programming capability
- 256B, 512B or 1KB register RAM
- 6 to 24 I/O pins depending upon package
- Internal Precision Oscillator
- Full-duplex UART
- Infrared Data Association (IrDA)-compliant infrared encoder/decoders, integrated with UART
- Two enhanced 16-bit timers with capture, compare, and PWM capability
- Watch-Dog Timer (WDT) with dedicated internal RC oscillator
- On-Chip Debugger
- Optional 8-channel, 10-bit Analog-to-Digital Converter (ADC)
- Up to 20 vectored interrupts
- Voltage Brown-out Protection (VBO)
- Power-On Reset (POR)
- 2.7 to 3.6V operating voltage
- Up to thirteen 5V-tolerant input pins
- 0° to +70°C and -40° to +105°C for operating temperature ranges



Part Selection Guide

Table 1 identifies the basic features and package styles available for each device within the PartName product line.

Table 1. 4K Series Family Part Selection Guide

| Part Number | Flash (KB) | RAM (B) | EEPROM (B) | I/O | ADC Inputs | Packages |
|-------------|------------|---------|------------|------|------------|---------------------|
| Z8F0423 | 4 | 1024 | 128 | 6–22 | 4–8 | 8-, 20- and 28-pins |
| Z8F0413 | 4 | 1024 | 128 | 6–24 | 0 | 8-, 20- and 28-pins |
| Z8F0223 | 2 | 512 | 64 | 6–22 | 4–8 | 8-, 20- and 28-pins |
| Z8F0213 | 2 | 512 | 64 | 6–24 | 0 | 8-, 20- and 28-pins |
| Z8F0123 | 1 | 256 | 16 | 6–22 | 4–8 | 8-, 20- and 28-pins |
| Z8F0113 | 1 | 256 | 16 | 6–24 | 0 | 8-, 20- and 28-pins |

Block Diagram

Figure 1 illustrates the block diagram of the architecture of the PartName devices.

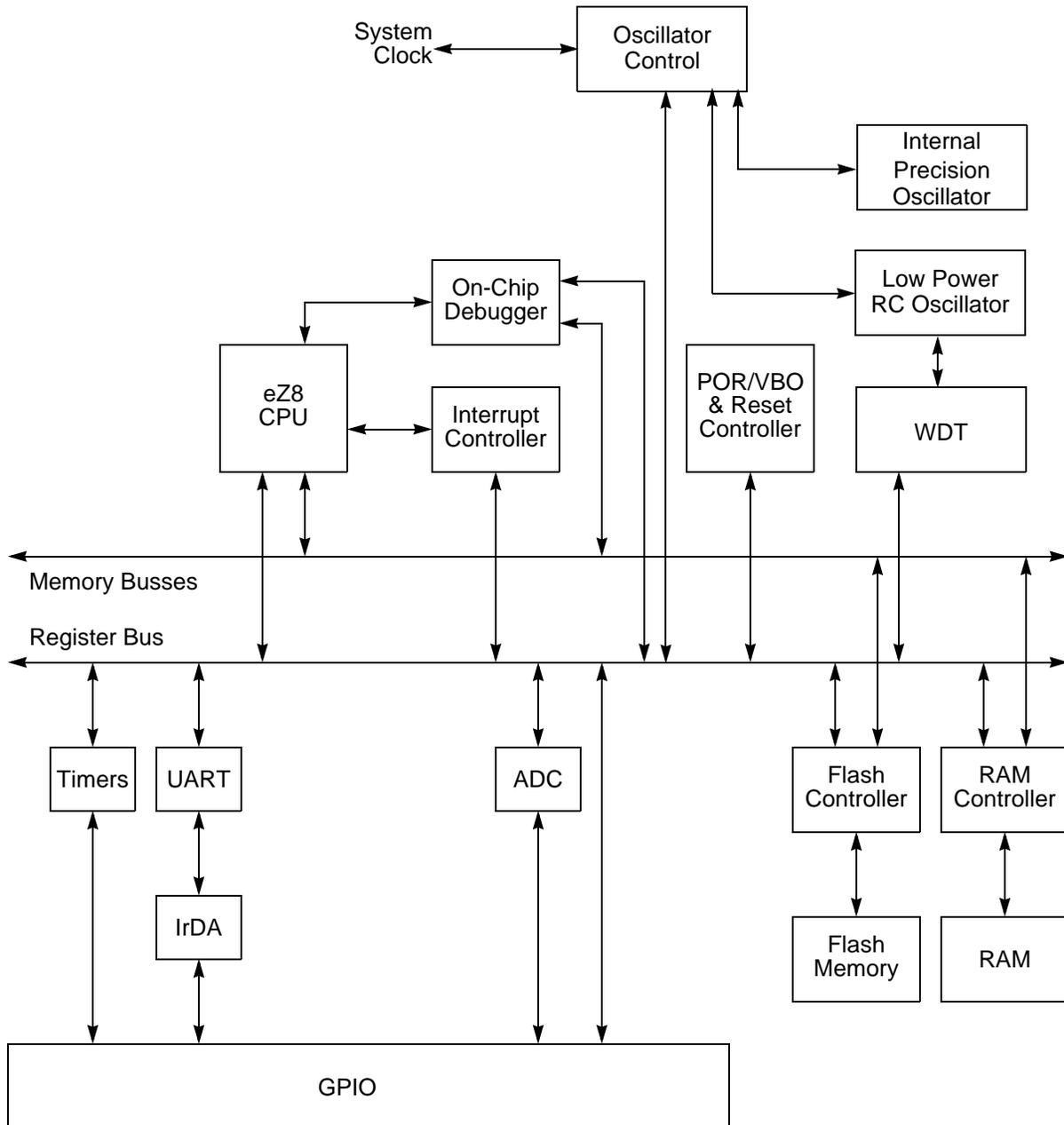


Figure 1. 4K Series Block Diagram



CPU and Peripheral Overview

eZ8 CPU Features

The eZ8 CPU, ZiLOG®'s latest 8-bit Central Processing Unit (CPU), meets the continuing demand for faster and more code-efficient microcontrollers. The eZ8 CPU executes a superset of the original Z8® instruction set. The eZ8 CPU features include:

- Direct register-to-register architecture allows each register to function as an accumulator, improving execution time and decreasing the required program memory
- Software stack allows much greater depth in subroutine calls and interrupts than hardware stacks
- Compatible with existing Z8® code
- Expanded internal Register File allows access of up to 4KB
- New instructions improve execution efficiency for code developed using higher-level programming languages, including C
- Pipelined instruction fetch and execution
- New instructions for improved performance including BIT, BSWAP, BTJ, CPC, LDC, LDCI, LEA, MULT, and SRL
- New instructions support 12-bit linear addressing of the Register File
- Up to 10 MIPS operation
- C-Compiler friendly
- 2 to 9 clock cycles per instruction

For more information regarding the eZ8 CPU, refer to the *eZ8 CPU User Manual* available for download at www.zilog.com.

General Purpose I/O

The Z8 Encore! PartName features 6 to 24 port pins (Ports A–C) for general purpose I/O (GPIO). The number of GPIO pins available is a function of package. Each pin is individually programmable.

Flash Controller

The Flash Controller programs and erases Flash memory. The Flash Controller supports protection against accidental program and erasure.



Internal Precision Oscillator

The Internal Precision Oscillator (IPO) is a trimmable clock source that requires no external components.

10-Bit Analog-to-Digital Converter

The optional Analog-to-Digital Converter (ADC) converts an analog input signal to a 10-bit binary number. The ADC accepts inputs from 8 different analog input pins in both single-ended and differential modes.

UART

The UART is full-duplex and capable of handling asynchronous data transfers. The UART supports 8- and 9-bit data modes and selectable parity. The UART also supports multi-drop address processing in hardware.

Timers

Two enhanced 16-bit reloadable timers can be used for timing/counting events or for motor control operations. These timers provide a 16-bit programmable reload counter and operate in One-Shot, Continuous, Gated, Capture, Capture Restart, Compare, Capture and Compare, PWM Single Output and PWM Dual Output modes.

Interrupt Controller

The Z8 Encore!® 4K Series products support up to 20 interrupts. These interrupts consist of 8 internal peripheral interrupts and 12 general-purpose I/O pin interrupt sources. The interrupts have 3 levels of programmable interrupt priority.

Reset Controller

The 4K Series products can be reset using the $\overline{\text{RESET}}$ pin, power-on reset, Watch-Dog Timer (WDT) time-out, STOP mode exit, or Voltage Brown-Out (VBO) warning signal. The $\overline{\text{RESET}}$ pin is bi-directional, meaning it functions as reset source as well as a reset indicator.

On-Chip Debugger

The 4K Series products feature an integrated On-Chip Debugger (OCD). The OCD provides a rich set of debugging capabilities, such as reading and writing registers, program-



ming Flash memory, setting breakpoints and executing code. A single-pin interface provides communication to the OCD.

Pin Description

Overview

The PartName products are available in a variety of packages styles and pin configurations. This chapter describes the signals and available pin configurations for each of the package styles. For information regarding the physical package specifications, refer to the chapter “Packaging” on page 179.

Available Packages

Table 2 identifies the package styles that are available for each device in the PartName product line.

Table 2. 4K Series Package Options

| Part Number | ADC | 8-pin PDIP | 8-pin SOIC | 20-pin PDIP | 20-pin SOIC | 20-pin SSOP | 28-pin PDIP | 28-pin SOIC | 28-pin SSOP |
|-------------|-----|------------|------------|-------------|-------------|-------------|-------------|-------------|-------------|
| Z8F0423 | Yes | X | X | X | X | X | X | X | X |
| Z8F0413 | No | X | X | X | X | X | X | X | X |
| Z8F0223 | Yes | X | X | X | X | X | X | X | X |
| Z8F0213 | No | X | X | X | X | X | X | X | X |
| Z8F0123 | Yes | X | X | X | X | X | X | X | X |
| Z8F0113 | No | X | X | X | X | X | X | X | X |

Pin Configurations

Figure 2 through Figure 4 illustrate the pin configurations for all of the packages available in the 4K Series. Refer to Table 3 for a description of the signals. The analog input alternate functions (ANAx) are not available on the Z8F041A, Z8F021A, and Z8F011A devices. The analog supply pins (AV_{DD} and AV_{SS}) are also not available on these parts, and are replaced by PB6 and PB7.

At reset, all Port A, B and C pins default to an input state. In addition, any alternate functionality is not enabled, so the pins function as general purpose input ports until programmed otherwise.

The pin configurations listed are preliminary and subject to change based on manufacturing limitations.

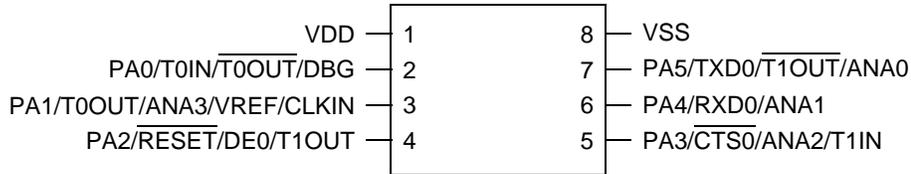


Figure 2. Z8F04x3, Z8F02x3, and Z8F01x3 in 8-Pin SOIC or PDIP Package

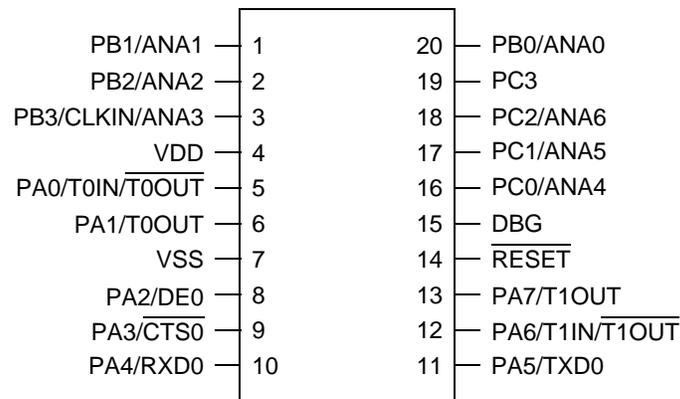


Figure 3. Z8F04x3, Z8F02x3, and Z8F01x3 in 20-Pin SOIC, SSOP or PDIP Package

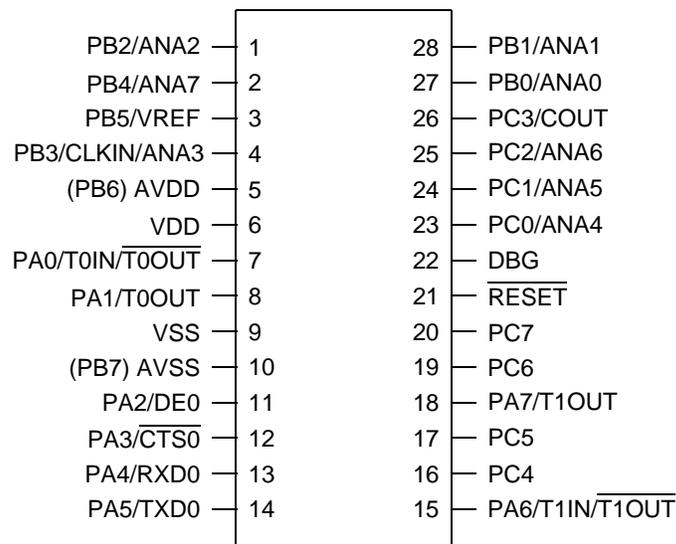


Figure 4. Z8F04x3, Z8F02x3, and Z8F01x3k in 28-Pin SOIC, SSOP or PDIP Package



Signal Descriptions

Table 3 describes the PartName signals. Refer to the section “Pin Configurations” on page 7 to determine the signals available for the specific package styles.

Table 3. Signal Descriptions

| Signal Mnemonic | I/O | Description |
|---|-----|---|
| General-Purpose I/O Ports A–D | | |
| PA[7:0] | I/O | Port A. These pins are used for general-purpose I/O. |
| PB[7:0] | I/O | Port B. These pins are used for general-purpose I/O. PB6 and PB7 are available only in those devices without an ADC. |
| PC[7:0] | I/O | Port C. These pins are used for general-purpose I/O. |
| Note: PB6 and PB7 are only available in 28-pin packages without ADC. In 28-pin packages with ADC, they are replaced by AV _{DD} and AV _{SS} . | | |
| UART Controllers | | |
| TXD0 | O | Transmit Data. This signal is the transmit output from the UART and IrDA. |
| RXD0 | I | Receive Data. This signal is the receive input for the UART and IrDA. |
| $\overline{\text{CTS0}}$ | I | Clear To Send. This signal is the flow control input for the UART. |
| DE | O | Driver Enable. This signal allows automatic control of external RS-485 drivers. This signal is approximately the inverse of the TXE (Transmit Empty) bit in the UART Status 0 register. The DE signal may be used to ensure the external RS-485 driver is enabled when data is transmitted by the UART. |
| Timers | | |
| T0OUT/T1OUT | O | Timer Output 0–1. These signals are output from the timers. |
| $\overline{\text{T0OUT}}/\overline{\text{T1OUT}}$ | O | Timer Complement Output 0–1. These signals are output from the timers in PWM Dual Output mode. |
| T0IN/T1IN | I | Timer Input 0–1. These signals are used as the <u>capture</u> , gating and counter inputs. The T0IN signal is multiplexed T0OUT signals. |
| Analog | | |
| ANA[7:0] | I | Analog Port. These signals are used as inputs to the analog-to-digital converter (ADC). |
| VREF | I/O | Analog-to-digital converter reference voltage input. |
| Clock Input | | |
| CLKIN | I | Clock Input Signal. This pin inputs a TTL-level signal to be used as the system clock. |

Table 3. Signal Descriptions (Continued)

| Signal Mnemonic | I/O | Description |
|---|-----|---|
| On-Chip Debugger | | |
| DBG | I/O | Debug. This signal is the control and data input and output to and from the On-Chip Debugger. |
|  Caution: | | The DBG pin is open-drain and requires an external pull-up resistor to ensure proper operation. |
| Reset | | |
| RESET | I/O | RESET. Generates a Reset when asserted (driven Low). Also serves as a reset indicator; the Z8 Encore!® forces this pin low when in reset. This pin is open-drain and features an enabled internal pull-up resistor. |
| Power Supply | | |
| V _{DD} | I | Digital Power Supply. |
| AV _{DD} | I | Analog Power Supply. |
| V _{SS} | I | Digital Ground. |
| AV _{SS} | I | Analog Ground. |
| Note: The AV _{DD} and AV _{SS} signals are available only in 28-pin packages with ADC. They are replaced by PB6 and PB7 on 28-pin packages without ADC. | | |

Pin Characteristics

Table 4 provides detailed information about the characteristics for each pin available on the 4K Series 20- and 28-pin devices. Data in Table 4 is sorted alphabetically by the pin symbol mnemonic.

Table 5 provides detailed information about the characteristics for each pin available on the 4K Series 8-pin devices,

- **Note:** All six I/O pins on the 8-pin packages are 5V-tolerant (unless the pull-up devices are enabled). The column in Table 4 below describes 5V-tolerance for the 20 and 28-pin packages only.



Table 4. Pin Characteristics (20- and 28-pin Devices)

| Symbol Mnemonic | Direction | Reset Direction | Active Low or Active High | Tristate Output | Internal Pull-up or Pull-down | Schmitt Trigger Input | Open Drain Output | 5V Tolerance |
|-----------------|-----------|-----------------|---------------------------|-----------------|-------------------------------|-----------------------|-------------------|--------------|
| AVDD | N/A | N/A | N/A | N/A | N/A | N/A | N/A | No |
| AVSS | N/A | N/A | N/A | N/A | N/A | N/A | N/A | No |
| DBG | I/O | I | N/A | Yes | No | Yes | Yes | Yes |
| PA[7:0] | I/O | I | N/A | Yes | Programmable Pull-up | Yes | Yes, Programmable | PA[7:2] only |
| PB[7:0] | I/O | I | N/A | Yes | Programmable Pull-up | Yes | Yes, Programmable | PB[7:6] only |
| PC[7:0] | I/O | I | N/A | Yes | Programmable Pull-up | Yes | Yes, Programmable | PC[7:3] only |
| RESET | I/O | I/O | Low | N/A | Pull-up | Yes | N/A | Yes |
| VDD | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A |
| VSS | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A |

► **Note:** PB6 and PB7 are available only in those devices without ADC.



Table 5. Pin Characteristics (8-Pin Devices)

| Symbol Mnemonic | Direction | Reset Direction | Active Low or Active High | Tristate Output | Internal Pull-up or Pull-down | Schmitt Trigger Input | Open Drain Output | 5V Tolerance |
|---------------------------------|-----------|--|---------------------------|-----------------|---|-----------------------|---|------------------------------|
| PA0/DBG | I/O | I (but can change during reset if key sequence detected) | N/A | Yes | Programmable Pull-up | Yes | Yes, Programmable | Yes, unless pull-ups enabled |
| PA1 | I/O | I | N/A | Yes | Programmable Pull-up | Yes | Yes, Programmable | Yes, unless pull-ups enabled |
| $\overline{\text{RESET}}$ / PA2 | I/O | I/O | N/A | Yes | Programmable for PA2; always on for $\overline{\text{RESET}}$ | Yes | programmable for PA2; always on for $\overline{\text{RESET}}$ | Yes, unless pull-ups enabled |
| PA[5:3] | I/O | I | N/A | Yes | Programmable Pull-up | Yes | Yes, Programmable | Yes, unless pull-ups enabled |
| VDD | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A |
| VSS | N/A | N/A | N/A | N/A | N/A | N/A | N/A | N/A |



Address Space

Overview

The eZ8 CPU can access three distinct address spaces:

- The Register File contains addresses for the general-purpose registers and the eZ8 CPU, peripheral, and general-purpose I/O port control registers.
- The Program Memory contains addresses for all memory locations having executable code and/or data.
- The Data Memory contains addresses for all memory locations that contain data only.

These three address spaces are covered briefly in the following subsections. For more detailed information regarding the eZ8 CPU and its address space, refer to the *eZ8 CPU User Manual* available for download at www.zilog.com.

Register File

The Register File address space in the Z8 Encore!® MCU is 4KB (4096 bytes). The Register File is composed of two sections: control registers and general-purpose registers. When instructions are executed, registers defined as sources are read, and registers defined as destinations are written. The architecture of the eZ8 CPU allows all general-purpose registers to function as accumulators, address pointers, index registers, stack areas, or scratch pad memory.

The upper 256 bytes of the 4KB Register File address space are reserved for control of the eZ8 CPU, the on-chip peripherals, and the I/O ports. These registers are located at addresses from F00H to FFFH. Some of the addresses within the 256B control register section are reserved (unavailable). Reading from a reserved Register File address returns an undefined value. Writing to reserved Register File addresses is not recommended and can produce unpredictable results.

The on-chip RAM always begins at address 000H in the Register File address space. The 4K Series devices contain 256B to 1KB of on-chip RAM. Reading from Register File addresses outside the available RAM addresses (and not within the control register address space) returns an undefined value. Writing to these Register File addresses produces no effect.



Program Memory

The eZ8 CPU supports 64KB of Program Memory address space. The PartName devices contain 1KB to 4KB of on-chip Flash memory in the Program Memory address space, depending on the device. Reading from Program Memory addresses outside the available Flash memory addresses returns FFH. Writing to these unimplemented Program Memory addresses produces no effect. Table 6 describes the Program Memory Maps for the Part-Name products.

Table 6. 4K Series Program Memory Maps

| Program Memory Address (Hex) | Function |
|-------------------------------------|--------------------------|
| Z8F042A and Z8F041A Products | |
| 0000–0001 | Flash Option Bits |
| 0002–0003 | Reset Vector |
| 0004–0005 | WDT Interrupt Vector |
| 0006–0007 | Illegal Instruction Trap |
| 0008–0037 | Interrupt Vectors* |
| 003E–0FFF | Program Memory |
| Z8F022A and Z8F021A Products | |
| 0000–0001 | Flash Option Bits |
| 0002–0003 | Reset Vector |
| 0004–0005 | WDT Interrupt Vector |
| 0006–0007 | Illegal Instruction Trap |
| 0008–0037 | Interrupt Vectors* |
| 003E–07FF | Program Memory |
| Z8F012A and Z8F011A Products | |
| 0000–0001 | Flash Option Bits |
| 0002–0003 | Reset Vector |
| 0004–0005 | WDT Interrupt Vector |
| 0006–0007 | Illegal Instruction Trap |
| 0008–0037 | Interrupt Vectors* |
| 003E–03FF | Program Memory |

* See **Table 30 on page 47** for a list of the interrupt vectors.

Data Memory

The 4K Series does not use the eZ8 CPU's 64KB Data Memory address space.

Flash Information Area

Table 7 describes the 4K Series Flash Information Area. This 128B Information Area is accessed by setting bit 7 of the Flash Page Select Register to 1. When access is enabled, the Flash Information Area is mapped into the Program Memory and overlays the 128 bytes at addresses FE00H to FF7FH. When the Information Area access is enabled, all reads from these Program Memory addresses return the Information Area data rather than the Program Memory data. Access to the Flash Information Area is read-only.

Table 7. 4K Series Flash Memory Information Area Map

| Program Memory Address (Hex) | Function |
|------------------------------|--|
| FE00–FE3F | ZiLOG Option Bits |
| FE40–FE53 | Part Number 20-character ASCII alphanumeric code Left justified and filled with FH |
| FE54–FE5F | Reserved |
| FE60–FE7F | ZiLOG Calibration Data |
| FE80–FFFF | Reserved |



Register Map

Table 8 provides the address map for the Register File of the PartName devices. Not all devices and package styles in the PartName support the ADC, or all of the GPIO Ports. Consider registers for unimplemented peripherals as Reserved.

Table 8. Register File Address Map

| Address (Hex) | Register Description | Mnemonic | Reset (Hex) | Page # |
|--------------------------------|-----------------------------------|----------|-------------|--------|
| General Purpose RAM | | | | |
| Z8F0423/Z8F0413 Devices | | | | |
| 000–3FF | General-Purpose Register File RAM | — | XX | |
| 400–EFF | Reserved | — | XX | |
| Z8F0223/Z8F0213 Devices | | | | |
| 000–1FF | General-Purpose Register File RAM | — | XX | |
| 200–EFF | Reserved | — | XX | |
| Z8F0123/Z8F0113 Devices | | | | |
| 000–0FF | General-Purpose Register File RAM | — | XX | |
| 100–EFF | Reserved | — | XX | |
| Timer 0 | | | | |
| F00 | Timer 0 High Byte | T0H | 00 | 72 |
| F01 | Timer 0 Low Byte | T0L | 01 | 72 |
| F02 | Timer 0 Reload High Byte | T0RH | FF | 73 |
| F03 | Timer 0 Reload Low Byte | T0RL | FF | 73 |
| F04 | Timer 0 PWM High Byte | T0PWMH | 00 | 73 |
| F05 | Timer 0 PWM Low Byte | T0PWML | 00 | 74 |
| F06 | Timer 0 Control 0 | T0CTL0 | 00 | 74 |
| F07 | Timer 0 Control 1 | T0CTL1 | 00 | 75 |
| Timer 1 | | | | |
| F08 | Timer 1 High Byte | T1H | 00 | 72 |
| F09 | Timer 1 Low Byte | T1L | 01 | 72 |
| F0A | Timer 1 Reload High Byte | T1RH | FF | 73 |
| F0B | Timer 1 Reload Low Byte | T1RL | FF | 73 |
| F0C | Timer 1 PWM High Byte | T1PWMH | 00 | 73 |
| F0D | Timer 1 PWM Low Byte | T1PWML | 00 | 74 |
| XX=Undefined | | | | |



Table 8. Register File Address Map (Continued)

| Address (Hex) | Register Description | Mnemonic | Reset (Hex) | Page # |
|--|---------------------------|----------|-------------|--------|
| F0E | Timer 1 Control 0 | T1CTL0 | 00 | 74 |
| F0F | Timer 1 Control 1 | T1CTL1 | 00 | 72 |
| F10–F3F | Reserved | — | XX | |
| UART 0 | | | | |
| F40 | UART0 Transmit Data | U0TXD | XX | 95 |
| | UART0 Receive Data | U0RXD | XX | 96 |
| F41 | UART0 Status 0 | U0STAT0 | 0000011Xb | 96 |
| F42 | UART0 Control 0 | U0CTL0 | 00 | 98 |
| F43 | UART0 Control 1 | U0CTL1 | 00 | 99 |
| F44 | UART0 Status 1 | U0STAT1 | 00 | 97 |
| F45 | UART0 Address Compare | U0ADDR | 00 | 101 |
| F46 | UART0 Baud Rate High Byte | U0BRH | FF | 101 |
| F47 | UART0 Baud Rate Low Byte | U0BRL | FF | 101 |
| F48–F6F | Reserved | — | XX | |
| Analog-to-Digital Converter (ADC) | | | | |
| F70 | ADC Control 0 | ADCCTL0 | 00 | 111 |
| F71 | ADC Control 1 | ADCCTL1 | 80 | 111 |
| F72 | ADC Data High Byte | ADCD_H | XX | 113 |
| F73 | ADC Data Low Bits | ADCD_L | XX | 113 |
| F74–F7F | Reserved | — | XX | |
| Low Power Control | | | | |
| F80 | Power Control 0 | PWRCTL0 | 80 | 30 |
| F81 | Reserved | — | XX | |
| F82–F85 | Reserved | — | XX | |
| Oscillator Control | | | | |
| F86 | Oscillator Control | OSCCTL | A0 | 144 |
| F87–FBF | Reserved | — | XX | |
| Interrupt Controller | | | | |
| FC0 | Interrupt Request 0 | IRQ0 | 00 | 51 |
| FC1 | IRQ0 Enable High Bit | IRQ0ENH | 00 | 53 |
| FC2 | IRQ0 Enable Low Bit | IRQ0ENL | 00 | 53 |
| FC3 | Interrupt Request 1 | IRQ1 | 00 | 52 |
| FC4 | IRQ1 Enable High Bit | IRQ1ENH | 00 | 54 |
| FC5 | IRQ1 Enable Low Bit | IRQ1ENL | 00 | 54 |
| XX=Undefined | | | | |



Table 8. Register File Address Map (Continued)

| Address (Hex) | Register Description | Mnemonic | Reset (Hex) | Page # |
|------------------------------|-----------------------------------|----------|-------------|--------|
| FC6 | Interrupt Request 2 | IRQ2 | 00 | 52 |
| FC7 | IRQ2 Enable High Bit | IRQ2ENH | 00 | 55 |
| FC8 | IRQ2 Enable Low Bit | IRQ2ENL | 00 | 55 |
| FC9–FCC | Reserved | — | XX | |
| FCD | Interrupt Edge Select | IRQES | 00 | 56 |
| FCE | Reserved | | | |
| FCF | Interrupt Control | IRQCTL | 00 | 56 |
| GPIO Port A | | | | |
| FD0 | Port A Address | PAADDR | 00 | 38 |
| FD1 | Port A Control | PACTL | 00 | 40 |
| FD2 | Port A Input Data | PAIN | XX | 40 |
| FD3 | Port A Output Data | PAOUT | 00 | 40 |
| GPIO Port B | | | | |
| FD4 | Port B Address | PBADDR | 00 | 38 |
| FD5 | Port B Control | PBCTL | 00 | 40 |
| FD6 | Port B Input Data | PBIN | XX | 40 |
| FD7 | Port B Output Data | PBOUT | 00 | 40 |
| GPIO Port C | | | | |
| FD8 | Port C Address | PCADDR | 00 | 38 |
| FD9 | Port C Control | PCCTL | 00 | 40 |
| FDA | Port C Input Data | PCIN | XX | 40 |
| FDB | Port C Output Data | PCOUT | 00 | 40 |
| FDC–FEF | Reserved | — | XX | |
| Watch-Dog Timer (WDT) | | | | |
| FF0 | Reset Status | RSTSTAT | XX | 82 |
| | Watch-Dog Timer Control | WDTCTL | XX | 82 |
| FF1 | Watch-Dog Timer Reload Upper Byte | WDTU | FF | 83 |
| FF2 | Watch-Dog Timer Reload High Byte | WDTH | FF | 83 |
| FF3 | Watch-Dog Timer Reload Low Byte | WDTL | FF | 83 |
| FF4–FF5 | Reserved | — | XX | |
| Trim Bit Control | | | | |
| FF6 | Trim Bit Address | TRMADR | 00 | 127 |
| FF7 | Trim Data | TRMDR | XX | 127 |
| XX=Undefined | | | | |



Table 8. Register File Address Map (Continued)

| Address (Hex) | Register Description | Mnemonic | Reset (Hex) | Page # |
|--------------------------------|---------------------------------------|----------|-------------|--|
| Flash Memory Controller | | | | |
| FF8 | Flash Control | FCTL | 00 | 122 |
| FF8 | Flash Status | FSTAT | 00 | 123 |
| FF9 | Flash Page Select | FPS | 00 | 124 |
| | Flash Sector Protect | FPROT | 00 | 124 |
| FFA | Flash Programming Frequency High Byte | FFREQH | 00 | 125 |
| FFB | Flash Programming Frequency Low Byte | FFREQL | 00 | 125 |
| eZ8 CPU | | | | |
| FFC | Flags | — | XX | Refer to the <i>eZ8 CPU User Manual</i> |
| FFD | Register Pointer | RP | XX | |
| FFE | Stack Pointer High Byte | SPH | XX | |
| FFF | Stack Pointer Low Byte | SPL | XX | |
| XX=Undefined | | | | |



Reset, STOP Mode Recovery and Low Voltage Detection

Overview

The Reset Controller within the PartName controls Reset and STOP Mode Recovery operation and provides indication of low supply voltage conditions. In typical operation, the following events cause a Reset:

- Power-On Reset (POR)
- Voltage Brown-Out (VBO)
- Watch-Dog Timer time-out (when configured by the WDT_RES Flash Option Bit to initiate a reset)
- External $\overline{\text{RESET}}$ pin assertion (when the alternate RESET function is enabled by the GPIO register)
- On-Chip Debugger initiated Reset (OCDCTL[0] set to 1)

When the device is in STOP mode, a STOP Mode Recovery is initiated by either of the following:

- Watch-Dog Timer time-out
- GPIO Port input pin transition on an enabled STOP Mode Recovery source

Reset Types

The PartName provides two types of Reset operation (Systems Reset, and STOP Mode Recovery). The reset type is a function of both the current operating mode of the Part-Name device and the source of the reset. Table 9 lists the types of Reset and their operating characteristics.

Table 9. Reset and STOP Mode Recovery Characteristics and Latency

| Reset Characteristics and Latency | | | |
|-----------------------------------|--|---------|---|
| Reset Type | Control Registers | eZ8 CPU | Reset Latency (Delay) |
| System Reset | Reset (as applicable) | Reset | 66 Internal Precision Oscillator Cycles |
| STOP Mode Recovery | Unaffected, except WDT_CTL and OSC_CTL registers | Reset | 66 Internal Precision Oscillator Cycles |



During a System Reset or STOP Mode Recovery, the PartName device is held in Reset for 66 cycles of the Internal Precision Oscillator. When a reset occurs because of a low voltage condition or power on reset, this delay is measured from the time that the supply voltage first exceeds the POR level (discussed later in this chapter). If the external pin reset remains asserted at the end of the reset period, the device remains in reset until the pin is deasserted.

At the beginning of Reset, all GPIO pins are configured as inputs with pull-up resistor disabled. During Reset, the eZ8 CPU and on-chip peripherals are idle; however, Watch-Dog Timer oscillator remains running.

Upon Reset, control registers within the Register File that have a defined Reset value are loaded with their reset values. Other control registers (including the Stack Pointer, Register Pointer, and Flags) and general-purpose RAM are undefined following Reset. The eZ8 CPU fetches the Reset vector at Program Memory addresses 0002H and 0003H and loads that value into the Program Counter. Program execution begins at the Reset vector address.

Because the control registers are re-initialized by a system reset, the system clock after reset is always the IPO. User software must reconfigure the oscillator control block, such that the correct system clock source is enabled and selected.

Reset Sources

Table 10 lists the possible sources of a system reset.

Table 10. Reset Sources and Resulting Reset Type

| Operating Mode | Reset Source | Special Conditions |
|----------------------|---|--|
| NORMAL or HALT modes | Power-On Reset / Voltage Brown-Out | Reset delay begins after supply voltage exceeds POR level |
| | Watch-Dog Timer time-out when configured for Reset | None |
| | $\overline{\text{RESET}}$ pin assertion | All reset pulses less than three system clocks in width are ignored. |
| | On-Chip Debugger initiated Reset (OCDCTL[0] set to 1) | System Reset, except the On-Chip Debugger is unaffected by the reset |
| STOP mode | Power-On Reset / Voltage Brown-Out | Reset delay begins after supply voltage exceeds POR level |
| | $\overline{\text{RESET}}$ pin assertion | All reset pulses less than the specified analog delay are ignored. See “Electrical Characteristics” on page 167. |
| | DBG pin driven Low | None |

Power-On Reset

Each device in the 4K Series contains an internal Power-On Reset (POR) circuit. The POR circuit monitors the supply voltage and holds the device in the Reset state until the supply voltage reaches a safe operating level. After the supply voltage exceeds the POR voltage threshold (V_{POR}), the device is held in the Reset state until the POR Counter has timed out.

After the 4K Series device exits the Power-On Reset state, the eZ8 CPU fetches the Reset vector. Following Power-On Reset, the POR status bit in the Watch-Dog Timer Control (WDTCTL) register is set to 1.

Figure 5 illustrates Power-On Reset operation. Refer to the “Electrical Characteristics” on page 167 for the POR threshold voltage (V_{POR}).

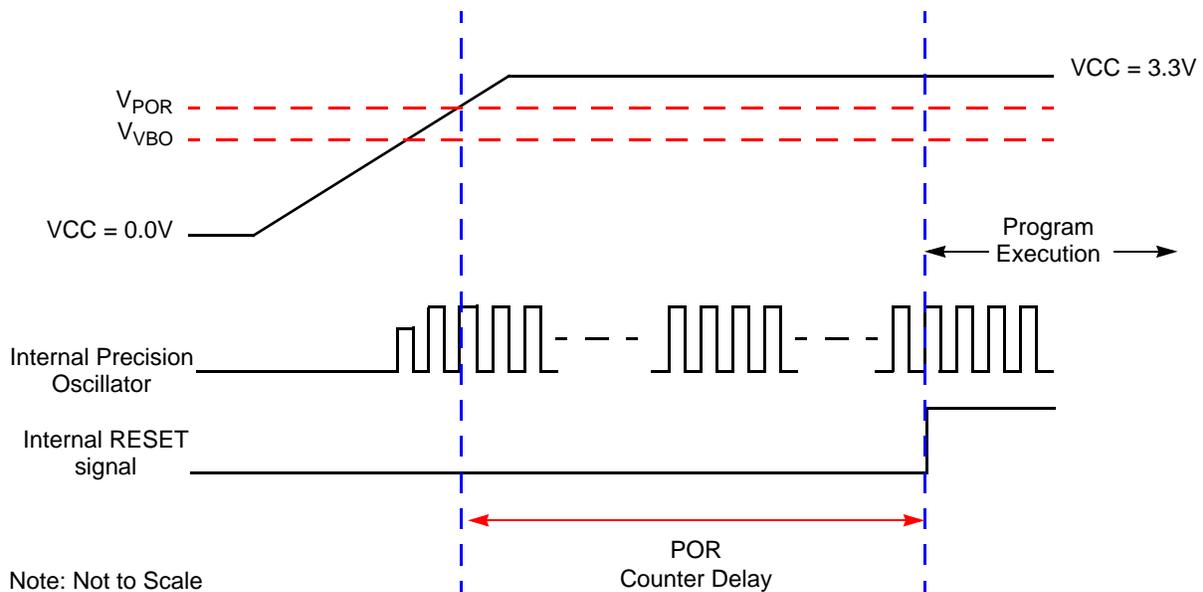


Figure 5. Power-On Reset Operation

Voltage Brown-Out Reset

The devices in the 4K Series provide low Voltage Brown-Out (VBO) protection. The VBO circuit senses when the supply voltage drops to an unsafe level (below the VBO threshold voltage) and forces the device into the Reset state. While the supply voltage remains below the Power-On Reset voltage threshold (V_{POR}), the VBO block holds the device in the Reset.

After the supply voltage again exceeds the Power-On Reset voltage threshold, the device progresses through a full System Reset sequence, as described in the Power-On Reset sec-

tion. Following Power-On Reset, the POR status bit in the Reset Status (RSTSTAT) register is set to 1. Figure 6 illustrates Voltage Brown-Out operation. Refer to the chapter “Electrical Characteristics” on page 167 for the VBO and POR threshold voltages (V_{VBO} and V_{POR}).

The Voltage Brown-Out circuit can be either enabled or disabled during STOP mode. Operation during STOP mode is set by the VBO_AO Flash Option Bit. Refer to the Flash Option Bits chapter for information about configuring VBO_AO.

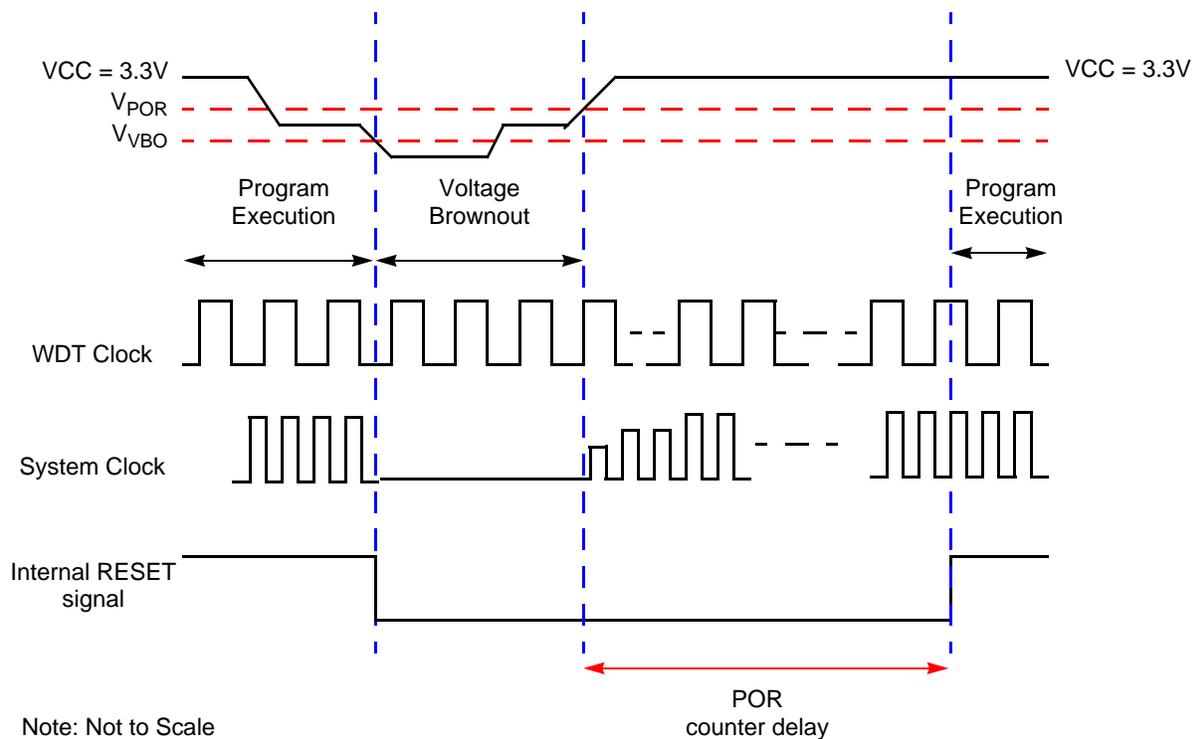


Figure 6. Voltage Brown-Out Reset Operation

The POR level is greater than the VBO level by the specified hysteresis value. This ensures that the device undergoes a power on reset after recovering from a VBO condition.

Watch-Dog Timer Reset

If the device is in NORMAL or STOP mode, the Watch-Dog Timer can initiate a System Reset at time-out if the WDT_RES Flash Option Bit is programmed to 1. This is the unprogrammed state of the WDT_RES Flash Option Bit. If the bit is programmed to 0, it configures the Watch-Dog Timer to cause an interrupt, not a System Reset, at time-out.

The WDT status bit in the WDT Control register is set to signify that the reset was initiated by the Watch-Dog Timer.

External Reset Input

The $\overline{\text{RESET}}$ pin has a Schmitt-triggered input and an internal pull-up resistor. Once the $\overline{\text{RESET}}$ pin is asserted for a minimum of 4 system clock cycles, the device progresses through the System Reset sequence. Because of the possible asynchronicity of the system clock and reset signals, the required reset duration may be as short as three clock periods and as long as four. A reset pulse three clock cycles in duration might trigger a reset; a pulse four cycles in duration always triggers a reset.

While the $\overline{\text{RESET}}$ input pin is asserted Low, the 4K Series devices remain in the Reset state. If the $\overline{\text{RESET}}$ pin is held Low beyond the System Reset time-out, the device exits the Reset state on the system clock rising edge following $\overline{\text{RESET}}$ pin deassertion. Following a System Reset initiated by the external $\overline{\text{RESET}}$ pin, the EXT status bit in the Watch-Dog Timer Control (WDTCTL) register is set to 1.

External Reset Indicator

During System Reset the $\overline{\text{RESET}}$ pin functions as an open-drain (active low) reset mode indicator. This reset output feature allows an 4K Series device to reset other components to which it is connected, even if that reset is caused by internal sources such as POR, VBO or WDT events.

After an internal reset event occurs, the internal circuitry begins driving the $\overline{\text{RESET}}$ pin Low. The $\overline{\text{RESET}}$ pin is held Low by the internal circuitry until the appropriate delay listed in Table 9 has elapsed.

STOP Mode Recovery

STOP mode is entered by execution of a STOP instruction by the eZ8 CPU. Refer to the chapter “**Low-Power Modes**” on page 28 for detailed STOP mode information. During STOP Mode Recovery, the CPU is held in reset for 66 IPO cycles.

STOP Mode Recovery does not affect onchip registers other than the Watch-Dog Timer Control register (WDTCTL) and the Oscillator Control register (OSCCTL). After any STOP Mode Recovery, the IPO is enabled and selected as the system clock. If another system clock source is required or IPO disabling is required, the STOP Mode Recovery code must reconfigure the oscillator control block such that the correct system clock source is enabled and selected.

The eZ8 CPU fetches the Reset vector at Program Memory addresses 0002H and 0003H and loads that value into the Program Counter. Program execution begins at the Reset vec-

tor address. Following STOP Mode Recovery, the STOP bit in the Watch-Dog Timer Control Register is set to 1. Table 11 lists the STOP Mode Recovery sources and resulting actions. The text following provides more detailed information about each of the STOP Mode Recovery sources.

Table 11. STOP Mode Recovery Sources and Resulting Action

| Operating Mode | STOP Mode Recovery Source | Action |
|----------------|---|--|
| STOP mode | Watch-Dog Timer time-out when configured for Reset | STOP Mode Recovery |
| | Watch-Dog Timer time-out when configured for interrupt | STOP Mode Recovery followed by interrupt (if interrupts are enabled) |
| | Data transition on any GPIO Port pin enabled as a STOP Mode Recovery source | STOP Mode Recovery |
| | Assertion of external $\overline{\text{RESET}}$ Pin | System Reset |
| | Debug Pin driven Low | System Reset |

STOP Mode Recovery Using Watch-Dog Timer Time-Out

If the Watch-Dog Timer times out during STOP mode, the device undergoes a STOP Mode Recovery sequence. In the Watch-Dog Timer Control register, the WDT and STOP bits are set to 1. If the Watch-Dog Timer is configured to generate an interrupt upon time-out and the 4K Series device is configured to respond to interrupts, the eZ8 CPU services the Watch-Dog Timer interrupt request following the normal STOP Mode Recovery sequence.

STOP Mode Recovery Using a GPIO Port Pin Transition

Each of the GPIO Port pins may be configured as a STOP Mode Recovery input source. On any GPIO pin enabled as a STOP Mode Recovery source, a change in the input pin value (from High to Low or from Low to High) initiates STOP Mode Recovery. In the Watch-Dog Timer Control register, the STOP bit is set to 1.



Caution: In STOP mode, the GPIO Port Input Data registers (PxIN) are disabled. The Port Input Data registers record the Port transition only if the signal stays on the Port pin through the end of the STOP Mode Recovery delay. As a result, short pulses on the Port pin can initiate STOP Mode Recovery without being written to the Port Input Data register or without initiating an interrupt (if enabled for that pin).



STOP Mode Recovery Using the External $\overline{\text{RESET}}$ Pin

When the 4K Series device is in STOP Mode and the external $\overline{\text{RESET}}$ pin is driven Low, a system reset occurs. Because of a glitch filter operating on the $\overline{\text{RESET}}$ pin, the Low pulse must be greater than the minimum width specified, or it is ignored. See “Electrical Characteristics” on page 167 for details.

Reset Register Definitions

Reset Status Register

The Reset Status (RSTSTAT) register is a read-only register that indicates the source of the most recent Reset event, indicates a STOP Mode Recovery event, and indicates a Watch-Dog Timer time-out. Reading this register resets the upper four bits to 0.

This register shares its address with the Watch-Dog Timer control register, which is write-only (Table 12).

Table 12. Reset Status Register (RSTSTAT)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------------------------|------|-----|-----|----------|---|---|---|
| FIELD | POR | STOP | WDT | EXT | Reserved | | | |
| RESET | See descriptions below | | | 0 | 0 | 0 | 0 | 0 |
| R/W | R | R | R | R | R | R | R | R |
| ADDR | FF0H | | | | | | | |

| Reset or STOP Mode Recovery Event | POR | STOP | WDT | EXT |
|---|-----|------|-----|-----|
| Power-On Reset | 1 | 0 | 0 | 0 |
| Reset using $\overline{\text{RESET}}$ pin assertion | 0 | 0 | 0 | 1 |
| Reset using Watch-Dog Timer time-out | 0 | 0 | 1 | 0 |
| Reset using the On-Chip Debugger (OCTCTL[1] set to 1) | 1 | 0 | 0 | 0 |
| Reset from STOP Mode using DBG Pin driven Low | 1 | 0 | 0 | 0 |
| STOP Mode Recovery using GPIO pin transition | 0 | 1 | 0 | 0 |
| STOP Mode Recovery using Watch-Dog Timer time-out | 0 | 1 | 1 | 0 |

POR—Power-On Reset Indicator

If this bit is set to 1, a Power-On Reset event occurred. This bit is reset to 0 if a WDT time-out or STOP Mode Recovery occurs. This bit is also reset to 0 when the register is read.

STOP—STOP Mode Recovery Indicator

If this bit is set to 1, a STOP Mode Recovery occurred. If the STOP and WDT bits are both



set to 1, the STOP Mode Recovery occurred because of a WDT time-out. If the STOP bit is 1 and the WDT bit is 0, the STOP Mode Recovery was not caused by a WDT time-out. This bit is reset by a Power-On Reset or a WDT time-out that occurred while not in STOP mode. Reading this register also resets this bit.

WDT—Watch-Dog Timer Time-Out Indicator

If this bit is set to 1, a WDT time-out occurred. A Power-On Reset resets this pin. A STOP Mode Recovery from a change in an input pin also resets this bit. Reading this register resets this bit. This read must occur before clearing the WDT interrupt.

EXT—External Reset Indicator

If this bit is set to 1, a Reset initiated by the external $\overline{\text{RESET}}$ pin occurred. A Power-On Reset or a STOP Mode Recovery from a change in an input pin resets this bit. Reading this register resets this bit.

Reserved—Must be 0.



Low-Power Modes

Overview

The 4K Series products contain power-saving features. The highest level of power reduction is provided by the STOP mode. The next lower level of power reduction is provided by the HALT mode.

Further power savings can be implemented by disabling individual peripheral blocks while in Normal mode.

STOP Mode

Executing the eZ8 CPU's STOP instruction places the device into STOP mode. In STOP mode, the operating characteristics are:

- Internal Precision Oscillator is stopped
- System clock is stopped
- eZ8 CPU is stopped
- Program counter (PC) stops incrementing
- Watch-Dog Timer's internal RC oscillator continues to operate if enabled by the Oscillator Control Register
- If enabled, the Watch-Dog Timer logic continues to operate
- If enabled for operation in STOP mode by the associated Flash Option Bit, the Voltage-brown out protection circuit continues to operate
- Transimpedance amplifier in the ADC block continues to operate if enabled by the Power Control Register to do so; all other portions of the ADC are disabled
- All other on-chip peripherals are idle

To minimize current in STOP mode, all GPIO pins that are configured as digital inputs must be driven to one of the supply rails (V_{CC} or GND). The device can be brought out of STOP mode using STOP Mode Recovery. For more information about STOP Mode Recovery refer to “Reset, STOP Mode Recovery and Low Voltage Detection” on page 20.



HALT Mode

Executing the eZ8 CPU's HALT instruction places the device into HALT mode. In HALT mode, the operating characteristics are:

- Primary oscillator is enabled and continues to operate
- System clock is enabled and continues to operate
- eZ8 CPU is stopped
- Program counter (PC) stops incrementing
- Watch-Dog Timer's internal RC oscillator continues to operate
- If enabled, the Watch-Dog Timer continues to operate
- All other on-chip peripherals continue to operate

The eZ8 CPU can be brought out of HALT mode by any of the following operations:

- Interrupt
- Watch-Dog Timer time-out (interrupt or reset)
- Power-on reset
- Voltage-brown out reset
- External $\overline{\text{RESET}}$ pin assertion

To minimize current in HALT mode, all GPIO pins that are configured as inputs must be driven to one of the supply rails (V_{CC} or GND).

Peripheral-Level Power Control

In addition to the STOP and Halt modes, it is possible to disable each peripheral on each of the 4K Series devices. Disabling a given peripheral minimizes its power consumption.

Power Control Register Definitions

Power Control Register 0

Each bit of the following registers disables a peripheral block, either by gating its system clock input or by removing power from the block.



Table 13. Power Control Register 0 (PWRCTL0)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------------|----------|----------|----------|----------|----------|----------|----------|----------|
| FIELD | Reserved | | | VBO | Reserved | ADC | Reserved | |
| RESET | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W |
| ADDR | F80H | | | | | | | |

Reserved—Must be 0.

VBO—Voltage Brown-Out Detector Disable

This bit and the VBO_AO Flash option bit must both enable the VBO for the VBO to be active.

0 = VBO Enabled

1 = VBO Disabled

ADC—Analog-to-Digital Converter Disable

0 = Analog-to-Digital Converter Enabled

1 = Analog-to-Digital Converter Disabled

Reserved—Must be 0.



General-Purpose I/O

Overview

The 4K Series products support a maximum of 25 port pins (Ports A–C) for general-purpose input/output (GPIO) operations. Each port contains control and data registers. The GPIO control registers determine data direction, open-drain, output drive current, programmable pull-ups, STOP Mode Recovery functionality, and alternate pin functions. Each port pin is individually programmable.

GPIO Port Availability By Device

Table 14 lists the port pins available with each device and package type.

Table 14. Port Availability by Device and Package Type

| Devices | Package | 10-Bit ADC | Port A | Port B | Port C | Total I/O |
|--|---------|------------|--------|--------|--------|-----------|
| Z8F0423SB, Z8F0423PB Z8F0223SB, Z8F0223PB Z8F0123SB, Z8F0123PB | 8-pin | Yes | [5:0] | No | No | 6 |
| Z8F0413SB, Z8F0413PB Z8F0213SB, Z8F0213PB Z8F0113SB, Z8F011vPB | 8-pin | No | [5:0] | No | No | 6 |
| Z8F0423PH, Z8F0423HH Z8F0223PH, Z8F0223HH Z8F0123PH, Z8F0123HH | 20-pin | Yes | [7:0] | [3:0] | [3:0] | 16 |
| Z8F0413PH, Z8F0413HH Z8F0213PH, Z8F0213HH Z8F0113PH, Z8F0113HH | 20-pin | No | [7:0] | [3:0] | [3:0] | 16 |
| Z8F0423PJ, Z8F0423SJ Z8F0223PJ, Z8F0223SJ Z8F0123PJ, Z8F0123SJ | 28-pin | Yes | [7:0] | [5:0] | [7:0] | 22 |
| Z8F0413PJ, Z8F0413SJ Z8F0213PJ, Z8F0213SJ Z8F0113PJ, Z8F0113SJ | 28-pin | No | [7:0] | [7:0] | [7:0] | 24 |

Architecture

Figure 7 illustrates a simplified block diagram of a GPIO port pin. In this figure, the ability to accommodate alternate functions and variable port current drive strength is not illustrated.

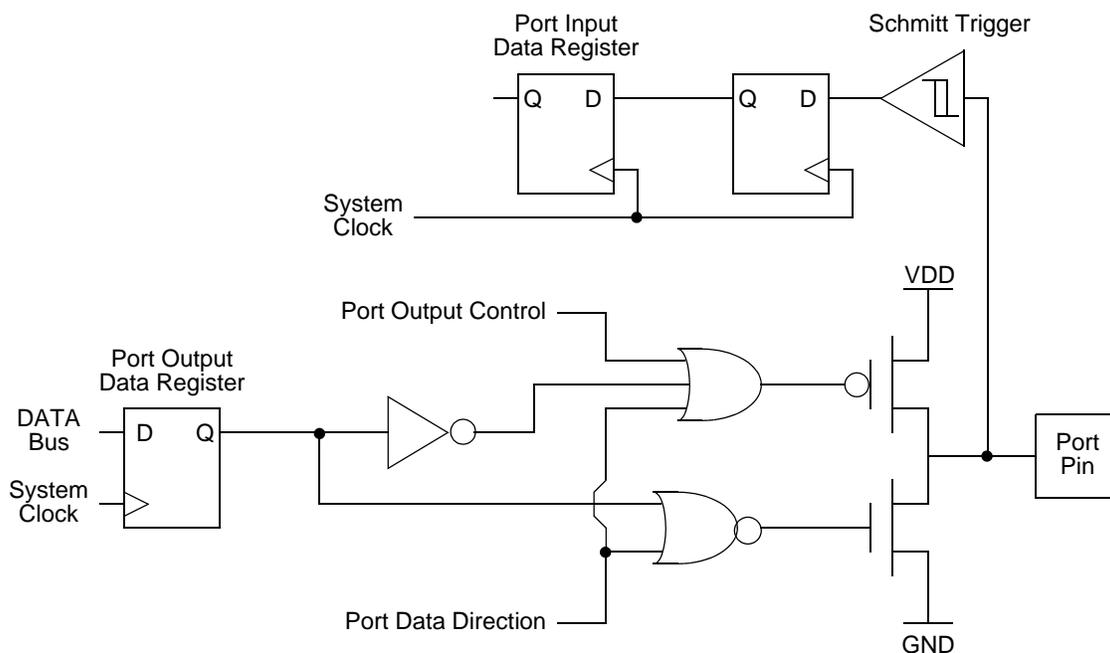


Figure 7. GPIO Port Pin Block Diagram

GPIO Alternate Functions

Many of the GPIO port pins can be used for general-purpose I/O and access to on-chip peripheral functions such as the timers and serial communication devices. The Port A–C Alternate Function sub-registers configure these pins for either General-Purpose I/O or alternate function operation. When a pin is configured for alternate function, control of the port pin direction (input/output) is passed from the Port A–C Data Direction registers to the alternate function assigned to this pin. Table 15 on page 34 lists the alternate functions possible with each port pin. The alternate function associated at a pin is defined through Alternate Function Sets sub-registers AFS1 and AFS2.

PA0 and PA6 contain two different timer functions, a timer input and a complementary timer output. Both of these functions require the same GPIO configuration, the selection between the two is based on the timer mode. See “Timers” on page 58 for more details.

Shared Debug Pin

On the 8-pin version of this device only, the Debug pin shares function with the PortA0 GPIO pin. This pin performs as a general purpose input pin on power-up, but the debug logic monitors this pin during the reset sequence to determine if the unlock sequence occurs. If the unlock sequence is present, the debug function is unlocked and the pin no longer functions as a GPIO pin. If it is not present, the debug feature is disabled until/ unless another reset event occurs. For more details, see “On-Chip Debugger” on page 131

5V Tolerance

All six I/O pins on the 8-pin devices are 5V-tolerant, unless the programmable pull-ups are enabled. If the pull-ups are enabled and inputs higher than V_{DD} are applied to these parts, excessive current flows through those pull-up devices and can damage the chip.

- **Note:** In the 20- and 28-pin versions of this device, any pin which shares functionality with an ADC is **not** 5V-tolerant, including PB[5:0] and PC[2:0]. All other signal pins are 5V-tolerant, and can safely handle inputs higher than V_{DD} even with the pull-ups enabled.

External Clock Setup

For systems using an external TTL drive, PB3 is the clock source for 20- and 28-pin devices. In this case, configure PB3 for alternate function CLKIN. Write the Oscillator Control Register (page 144) such that the external oscillator is selected as the system clock. For 8-pin devices use PA1 instead of PB3.



Table 15. Port Alternate Function Mapping (Non 8-Pin Parts)

| Port | Pin | Mnemonic | Alternate Function Description | Alternate Function Set Register AFS1 |
|--------|----------|----------------|---|--------------------------------------|
| Port A | PA0 | T0IN/T0OUT | Timer 0 Input/Timer 0 Output Complement | N/A |
| | | Reserved | | |
| | PA1 | T0OUT | Timer 0 Output | |
| | | Reserved | | |
| | PA2 | DE0 | UART 0 Driver Enable | |
| | | Reserved | | |
| | PA3 | CTS0 | UART 0 Clear to Send | |
| | | Reserved | | |
| | PA4 | RXD0/IRRX0 | UART 0 / IrDA 0 Receive Data | |
| | | Reserved | | |
| | PA5 | TXD0/IRTX0 | UART 0 / IrDA 0 Transmit Data | |
| | | Reserved | | |
| | PA6 | T1IN/T1OUT | Timer 1 Input/Timer 1 Output Complement | |
| | | Reserved | | |
| PA7 | T1OUT | Timer 1 Output | | |
| | Reserved | | | |

Note: Because there is only a single alternate function for each Port A pin, the Alternate Function Set registers are not implemented for Port A. Enabling alternate function selections as described in “Port A–C Alternate Function Sub-Registers” on page 40 automatically enables the associated alternate function.



Table 15. Port Alternate Function Mapping (Continued)(Non 8-Pin Parts)

| Port | Pin | Mnemonic | Alternate Function Description | Alternate Function Set Register AFS1 |
|---------------|-----|-------------|-------------------------------------|--------------------------------------|
| Port B | PB0 | Reserved | | AFS1[0]: 0 |
| | | ANA0/AMPOUT | ADC Analog Input/Transamp Output | AFS1[0]: 1 |
| | PB1 | Reserved | | AFS1[1]: 0 |
| | | ANA1/AMPINN | ADC Analog Input/Transamp Input (N) | AFS1[1]: 1 |
| | PB2 | Reserved | | AFS1[2]: 0 |
| | | ANA2/AMPINP | ADC Analog Input/Transamp Input (P) | AFS1[2]: 1 |
| | PB3 | CLKIN | External Clock Input | AFS1[3]: 0 |
| | | ANA3 | ADC Analog Input | AFS1[3]: 1 |
| | PB4 | Reserved | | AFS1[4]: 0 |
| | | ANA7 | ADC Analog Input | AFS1[4]: 1 |
| | PB5 | Reserved | | AFS1[5]: 0 |
| | | VREF | ADC Voltage Reference | AFS1[5]: 1 |
| | PB6 | Reserved | | AFS1[6]: 0 |
| | | Reserved | | AFS1[6]: 1 |
| | PB7 | Reserved | | AFS1[7]: 0 |
| | | Reserved | | AFS1[7]: 1 |

Note: Because there are at most two choices of alternate function for any pin of Port B, the Alternate Function Set register AFS2 is implemented but not used to select the function. Also, alternate function selection as described in “Port A–C Alternate Function Sub-Registers” on page 40 must also be enabled.



Table 15. Port Alternate Function Mapping (Continued)(Non 8-Pin Parts)

| Port | Pin | Mnemonic | Alternate Function Description | Alternate Function Set Register AFS1 |
|---------------|------------|-----------------|---------------------------------------|---|
| Port C | PC0 | Reserved | | AFS1[0]: 0 |
| | | ANA4 | ADC | AFS1[0]: 1 |
| | PC1 | Reserved | | AFS1[1]: 0 |
| | | ANA5 | ADC | AFS1[1]: 1 |
| | PC2 | Reserved | | AFS1[2]: 0 |
| | | ANA6 | ADC Analog Input | AFS1[2]: 1 |
| | PC3 | Reserved | | AFS1[3]: 0 |
| | | Reserved | | AFS1[3]: 1 |
| | PC4 | Reserved | | AFS1[4]: 0 |
| | | Reserved | | AFS1[4]: 1 |
| | PC5 | Reserved | | AFS1[5]: 0 |
| | | Reserved | | AFS1[5]: 1 |
| | PC6 | Reserved | | AFS1[6]: 0 |
| | | Reserved | | AFS1[6]: 1 |
| | PC7 | Reserved | | AFS1[7]: 0 |
| | | Reserved | | AFS1[7]: 1 |

Note: Because there are at most two choices of alternate function for any pin of Port C, the Alternate Function Set register AFS2 is implemented but not used to select the function.



Table 16. Port Alternate Function Mapping (8-Pin Parts)

| Port | Pin | Mnemonic | Alternate Function Description | Alternate Function Select Register AFS2 | Alternate Function Select Register AFS1 |
|--------|----------|---------------------------|--------------------------------|---|---|
| Port A | PA0 | T0IN | Timer 0 Input | AFS2[0]: 0 | AFS1[0]: 0 |
| | | Reserved | | AFS2[0]: 0 | AFS1[0]: 1 |
| | | DBG | OCD Control and Data I/O | AFS2[0]: 1 | AFS1[0]: 0 |
| | | $\overline{\text{T0OUT}}$ | Timer 0 Output Complement | AFS2[0]: 1 | AFS1[0]: 1 |
| | PA1 | T0OUT | Timer 0 Output | AFS2[1]: 0 | AFS1[1]: 0 |
| | | Reserved | | AFS2[1]: 0 | AFS1[1]: 1 |
| | | CLKIN | External Clock Input | AFS2[1]: 1 | AFS1[1]: 0 |
| | | ANA3 | ADC Analog Input/VREF | AFS2[1]: 1 | AFS1[1]: 1 |
| | PA2 | DE0 | UART 0 Driver Enable | AFS2[2]: 0 | AFS1[2]: 0 |
| | | $\overline{\text{RESET}}$ | External Reset | AFS2[2]: 0 | AFS1[2]: 1 |
| | | T1OUT | Timer 1 Output | AFS2[2]: 1 | AFS1[2]: 0 |
| | | Reserved | | AFS2[2]: 1 | AFS1[2]: 1 |
| | PA3 | $\overline{\text{CTS0}}$ | UART 0 Clear to Send | AFS2[3]: 0 | AFS1[3]: 0 |
| | | COUT | Comparator Output | AFS2[3]: 0 | AFS1[3]: 1 |
| | | T1IN | Timer 1 Input | AFS2[3]: 1 | AFS1[3]: 0 |
| | | ANA2 | ADC Analog Input | AFS2[3]: 1 | AFS1[3]: 1 |
| | PA4 | RXD0 | UART 0 Receive Data | AFS2[4]: 0 | AFS1[4]: 0 |
| | | Reserved | | AFS2[4]: 0 | AFS1[4]: 1 |
| | | Reserved | | AFS2[4]: 1 | AFS1[4]: 0 |
| | | ANA1 | ADC Analog Input | AFS2[4]: 1 | AFS1[4]: 1 |
| PA5 | TXD0 | UART 0 Transmit Data | AFS2[5]: 0 | AFS1[5]: 0 | |
| | T1OUT | Timer 1 Output Complement | AFS2[5]: 0 | AFS1[5]: 1 | |
| | Reserved | | AFS2[5]: 1 | AFS1[5]: 0 | |
| | ANA0 | ADC Analog Input | AFS2[5]: 1 | AFS1[5]: 1 | |

GPIO Interrupts

Many of the GPIO port pins can be used as interrupt sources. Some port pins can be configured to generate an interrupt request on either the rising edge or falling edge of the pin input signal. Other port pin interrupt sources generate an interrupt when any edge occurs (both rising and falling). Refer to the chapter “**Interrupt Controller**” on page 46 for more information about interrupts using the GPIO pins.

GPIO Control Register Definitions

Four registers for each Port provide access to GPIO control, input data, and output data. Table 17 lists these Port registers. Use the Port A–C Address and Control registers together to provide access to sub-registers for Port configuration and control.

Table 17. GPIO Port Registers and Sub-Registers

| Port Register Mnemonic | Port Register Name |
|-----------------------------------|---|
| PxADDR | Port A–C Address Register (Selects sub-registers) |
| PxCTL | Port A–C Control Register (Provides access to sub-registers) |
| PxIN | Port A–C Input Data Register |
| PxOUT | Port A–C Output Data Register |
| Port Sub-Register Mnemonic | Port Register Name |
| PxDD | Data Direction |
| PxAF | Alternate Function |
| PxOC | Output Control (Open-Drain) |
| PxHDE | High Drive Enable |
| PxSMRE | STOP Mode Recovery Source Enable |
| PxPUE | Pull-up Enable |
| PxAFS1 | Alternate Function Set 1 |
| PxAFS2 | Alternate Function Set 2 |



Port A–C Address Registers

The Port A–C Address registers select the GPIO Port functionality accessible through the Port A–C Control registers. The Port A–C Address and Control registers combine to provide access to all GPIO Port controls (Table 18).

Table 18. Port A–C GPIO Address Registers (PxADDR)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------------------|-----|-----|-----|-----|-----|-----|-----|
| FIELD | PADDR[7:0] | | | | | | | |
| RESET | 00H | | | | | | | |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | FD0H, FD4H, FD8H | | | | | | | |

PADDR[7:0]—Port Address

The Port Address selects one of the sub-registers accessible through the Port Control register.

| PADDR[7:0] | Port Control sub-register accessible using the Port A–C Control Registers |
|------------|--|
| 00H | No function. Provides some protection against accidental Port reconfiguration. |
| 01H | Data Direction |
| 02H | Alternate Function |
| 03H | Output Control (Open-Drain) |
| 04H | High Drive Enable |
| 05H | STOP Mode Recovery Source Enable. |
| 06H | Pull-up Enable |
| 07H | Alternate Function Set 1 |
| 08H | Alternate Function Set 2 |
| 09H–FFH | No function |

Port A–C Control Registers

The Port A–C Control registers set the GPIO port operation. The value in the corresponding Port A–C Address register determines which sub-register is read from or written to by a Port A–C Control register transaction (Table 19).



Table 19. Port A–C Control Registers (PxCTL)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------------------|-----|-----|-----|-----|-----|-----|-----|
| FIELD | PCTL | | | | | | | |
| RESET | 00H | | | | | | | |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | FD1H, FD5H, FD9H | | | | | | | |

PCTL[7:0]—Port Control

The Port Control register provides access to all sub-registers that configure the GPIO Port operation.

Port A–C Data Direction Sub-Registers

The Port A–C Data Direction sub-register is accessed through the Port A–C Control register by writing 01H to the Port A–C Address register (Table 20).

Table 20. Port A–C Data Direction Sub-Registers (PxDD)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---|-----|-----|-----|-----|-----|-----|-----|
| FIELD | DD7 | DD6 | DD5 | DD4 | DD3 | DD2 | DD1 | DD0 |
| RESET | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | If 01H in Port A–C Address Register, accessible through the Port A–C Control Register | | | | | | | |

DD[7:0]—Data Direction

These bits control the direction of the associated port pin. Port Alternate Function operation overrides the Data Direction register setting.

0 = Output. Data in the Port A–C Output Data register is driven onto the port pin.

1 = Input. The port pin is sampled and the value written into the Port A–C Input Data Register. The output driver is tristated.

Port A–C Alternate Function Sub-Registers

The Port A–C Alternate Function sub-register (Table 21) is accessed through the Port A–C Control register by writing 02H to the Port A–C Address register. The Port A–C Alternate Function sub-registers enable the alternate function selection on pins. If disabled, pins function as GPIO. If enabled, select one of four alternate functions using alternate function set subregisters 1 and 2 as described in the “**Port A–C Alternate Function Set 1 Sub-Registers**” on page 43 and “**Port A–C Alternate Function Set 2 Sub-Registers**” on page 44. Refer to the “**GPIO Alternate Functions**” on page 32 to determine the alternate function associated with each port pin.



Caution: Do not enable alternate functions for GPIO port pins for which there is no associated alternate function. Failure to follow this guideline can result in unpredictable operation.

Table 21. Port A–C Alternate Function Sub-Registers (PxAF)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------------|---|----------|----------|----------|----------|----------|----------|----------|
| FIELD | AF7 | AF6 | AF5 | AF4 | AF3 | AF2 | AF1 | AF0 |
| RESET | 00H (Ports A–C) | | | | | | | |
| R/W | R/W | | | | | | | |
| ADDR | If 02H in Port A–C Address Register, accessible through the Port A–C Control Register | | | | | | | |

AF[7:0]—Port Alternate Function enabled
 0 = The port pin is in normal mode and the DDx bit in the Port A–C Data Direction sub-register determines the direction of the pin.
 1 = The alternate function selected through Alternate Function Set sub-registers is enabled. Port pin operation is controlled by the alternate function.

Port A–C Output Control Sub-Registers

The Port A–C Output Control sub-register (Table 22) is accessed through the Port A–C Control register by writing 03H to the Port A–C Address register. Setting the bits in the Port A–C Output Control sub-registers to 1 configures the specified port pins for open-drain operation. These sub-registers affect the pins directly and, as a result, alternate functions are also affected.

Table 22. Port A–C Output Control Sub-Registers (PxOC)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------------|---|----------|----------|----------|----------|----------|----------|----------|
| FIELD | POC7 | POC6 | POC5 | POC4 | POC3 | POC2 | POC1 | POC0 |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | If 03H in Port A–C Address Register, accessible through the Port A–C Control Register | | | | | | | |

POC[7:0]—Port Output Control
 These bits function independently of the alternate function bit and always disable the drains if set to 1.
 0 = The drains are enabled for any output mode (unless overridden by the alternate function).
 1 = The drain of the associated pin is disabled (open-drain mode).



Port A–C High Drive Enable Sub-Registers

The Port A–C High Drive Enable sub-register (Table 23) is accessed through the Port A–C Control register by writing 04H to the Port A–C Address register. Setting the bits in the Port A–C High Drive Enable sub-registers to 1 configures the specified port pins for high current output drive operation. The Port A–C High Drive Enable sub-register affects the pins directly and, as a result, alternate functions are also affected.

Table 23. Port A–C High Drive Enable Sub-Registers (PxHDE)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---|-------|-------|-------|-------|-------|-------|-------|
| FIELD | PHDE7 | PHDE6 | PHDE5 | PHDE4 | PHDE3 | PHDE2 | PHDE1 | PHDE0 |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | If 04H in Port A–C Address Register, accessible through the Port A–C Control Register | | | | | | | |

PHDE[7:0]—Port High Drive Enabled

0 = The Port pin is configured for standard output current drive.

1 = The Port pin is configured for high output current drive.

Port A–C STOP Mode Recovery Source Enable Sub-Registers

The Port A–C STOP Mode Recovery Source Enable sub-register (Table 24) is accessed through the Port A–C Control register by writing 05H to the Port A–C Address register. Setting the bits in the Port A–C STOP Mode Recovery Source Enable sub-registers to 1 configures the specified Port pins as a STOP Mode Recovery source. During STOP Mode, any logic transition on a Port pin enabled as a STOP Mode Recovery source initiates STOP Mode Recovery.

Table 24. Port A–C STOP Mode Recovery Source Enable Sub-Registers (PxSMRE)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---|--------|--------|--------|--------|--------|--------|--------|
| FIELD | PSMRE7 | PSMRE6 | PSMRE5 | PSMRE4 | PSMRE3 | PSMRE2 | PSMRE1 | PSMRE0 |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | If 05H in Port A–C Address Register, accessible through the Port A–C Control Register | | | | | | | |

PSMRE[7:0]—Port STOP Mode Recovery Source Enabled

0 = The Port pin is not configured as a STOP Mode Recovery source. Transitions on this pin during STOP mode do not initiate STOP Mode Recovery.

1 = The Port pin is configured as a STOP Mode Recovery source. Any logic transition on this pin during STOP mode initiates STOP Mode Recovery.



Port A–C Pull-up Enable Sub-Registers

The Port A–C Pull-up Enable sub-register (Table 25) is accessed through the Port A–C Control register by writing 06H to the Port A–C Address register. Setting the bits in the Port A–C Pull-up Enable sub-registers enables a weak internal resistive pull-up on the specified Port pins.

Table 25. Port A–C Pull-Up Enable Sub-Registers (PxPUE)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---|-------|-------|-------|-------|-------|-------|-------|
| FIELD | PPUE7 | PPUE6 | PPUE5 | PPUE4 | PPUE3 | PPUE2 | PPUE1 | PPUE0 |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | If 06H in Port A–C Address Register, accessible through the Port A–C Control Register | | | | | | | |

PPUE[7:0]—Port Pull-up Enabled

0 = The weak pull-up on the Port pin is disabled.

1 = The weak pull-up on the Port pin is enabled.

Port A–C Alternate Function Set 1 Sub-Registers

The Port A–C Alternate Function Set1 sub-register (Table 26) is accessed through the Port A–C Control register by writing 07H to the Port A–C Address register. The Alternate Function Set 1 sub-registers selects the alternate function available at a port pin. Alternate Functions selected by setting or clearing bits of this register are defined in “GPIO Alternate Functions” on page 32.

► **Note:** Alternate function selection on port pins must also be enabled as described in “Port A–C Alternate Function Sub-Registers” on page 40.

Table 26. Port A–C Alternate Function Set 1 Sub-Registers (PxAFS1)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---|--------|--------|--------|--------|--------|--------|--------|
| FIELD | PAFS17 | PAFS16 | PAFS15 | PAFS14 | PAFS13 | PAFS12 | PAFS11 | PAFS10 |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | If 07H in Port A–C Address Register, accessible through the Port A–C Control Register | | | | | | | |

PAFS1[7:0]—Port Alternate Function Set 1

0 = Port Alternate Function selected as defined in Table 14 in the **GPIO Alternate Functions** section.

1 = Port Alternate Function selected as defined in Table 14 in the **GPIO Alternate Functions** section.



Port A–C Alternate Function Set 2 Sub-Registers

The Port A–C Alternate Function Set 2 sub-register (Table 27) is accessed through the Port A–C Control register by writing 08H to the Port A–C Address register. The Alternate Function Set 2 sub-registers selects the alternate function available at a port pin. Alternate Functions selected by setting or clearing bits of this register is defined in Table 14 in the section “GPIO Alternate Functions” on page 32.

► **Note:** Alternate function selection on port pins must also be enabled as described in “Port A–C Alternate Function Sub-Registers” on page 40.

Table 27. Port A–C Alternate Function Set 2 Sub-Registers (PxAFS2)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---|--------|--------|--------|--------|--------|--------|--------|
| FIELD | PAFS27 | PAFS26 | PAFS25 | PAFS24 | PAFS23 | PAFS22 | PAFS21 | PAFS20 |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | If 08H in Port A–C Address Register, accessible through the Port A–C Control Register | | | | | | | |

PAFS2[7:0]—Port Alternate Function Set 2

0 = Port Alternate Function selected as defined in Table 14 **GPIO Alternate Functions** section.

1 = Port Alternate Function selected as defined in Table 14 **GPIO Alternate Functions** section.

Port A–C Input Data Registers

Reading from the Port A–C Input Data registers (Table 28) returns the sampled values from the corresponding port pins. The Port A–C Input Data registers are read-only. The value returned for any unused ports is 0. Unused ports include those missing on the 8- and 28-pin packages, as well as those missing on the ADC-enabled 28-pin packages.

Table 28. Port A–C Input Data Registers (PxIN)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------------------|------|------|------|------|------|------|------|
| FIELD | PIN7 | PIN6 | PIN5 | PIN4 | PIN3 | PIN2 | PIN1 | PIN0 |
| RESET | X | X | X | X | X | X | X | X |
| R/W | R | R | R | R | R | R | R | R |
| ADDR | FD2H, FD6H, FDAH | | | | | | | |

PIN[7:0]—Port Input Data

Sampled data from the corresponding port pin input.

0 = Input data is logical 0 (Low).

1 = Input data is logical 1 (High).



Port A–C Output Data Register

The Port A–C Output Data register (Table 29) controls the output data to the pins.

Table 29. Port A–C Output Data Register (PxOUT)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------------|------------------|----------|----------|----------|----------|----------|----------|----------|
| FIELD | POUT7 | POUT6 | POUT5 | POUT4 | POUT3 | POUT2 | POUT1 | POUT0 |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | FD3H, FD7H, FDBH | | | | | | | |

POUT[7:0]—Port Output Data

These bits contain the data to be driven to the port pins. The values are only driven if the corresponding pin is configured as an output and the pin is not configured for alternate function operation.

0 = Drive a logical 0 (Low).

1 = Drive a logical 1 (High). High value is not driven if the drain has been disabled by setting the corresponding Port Output Control register bit to 1.

Interrupt Controller

Overview

The interrupt controller on the 4K Series products prioritizes the interrupt requests from the on-chip peripherals and the GPIO port pins. The features of the interrupt controller include the following:

- 20 unique interrupt vectors:
 - 14 GPIO port pin interrupt sources (two are shared)
 - 10 on-chip peripheral interrupt sources (two are shared)
- Flexible GPIO interrupts
 - 8 selectable rising and falling edge GPIO interrupts
 - 4 dual-edge interrupts
- 3 levels of individually programmable interrupt priority
- Watch-Dog Timer can be configured to generate an interrupt

Interrupt requests (IRQs) allow peripheral devices to suspend CPU operation in an orderly manner and force the CPU to start an interrupt service routine (ISR). Usually this interrupt service routine is involved with the exchange of data, status information, or control information between the CPU and the interrupting peripheral. When the service routine is completed, the CPU returns to the operation from which it was interrupted.

The eZ8 CPU supports both vectored and polled interrupt handling. For polled interrupts, the interrupt controller has no effect on operation. Refer to the *eZ8 CPU User Manual* for more information regarding interrupt servicing by the eZ8 CPU. The *eZ8 CPU User Manual* is available for download at www.zilog.com.

Interrupt Vector Listing

Table 30 lists all of the interrupts available in order of priority. The interrupt vector is stored with the most significant byte (MSB) at the even Program Memory address and the least significant byte (LSB) at the following odd Program Memory address.

- **Note:** Some port interrupts are not available on the 8- and 20-pin packages. The ADC interrupt is unavailable on devices not containing an ADC.



Table 30. Trap and Interrupt Vectors in Order of Priority

| Priority | Program Memory Vector Address | Interrupt or Trap Source |
|-----------------|--------------------------------------|---|
| Highest | 0002H | Reset (not an interrupt) |
| | 0004H | Watch-Dog Timer (see Watch-Dog Timer chapter) |
| | 0006H | Illegal Instruction Trap (not an interrupt) |
| | 0008H | Reserved |
| | 000AH | Timer 1 |
| | 000CH | Timer 0 |
| | 000EH | UART 0 receiver |
| | 0010H | UART 0 transmitter |
| | 0012H | Reserved |
| | 0014H | Reserved |
| | 0016H | ADC |
| | 0018H | Port A7, selectable rising or falling input edge |
| | 001AH | Port A6, selectable rising or falling input edge |
| | 001CH | Port A5, selectable rising or falling input edge |
| | 001EH | Port A4, selectable rising or falling input edge |
| | 0020H | Port A3 or Port D3, selectable rising or falling input edge |
| | 0022H | Port A2 or Port D2, selectable rising or falling input edge |
| | 0024H | Port A1, selectable rising or falling input edge |
| | 0026H | Port A0, selectable rising or falling input edge |
| | 0028H | Reserved |
| | 002AH | Reserved |
| | 002CH | Reserved |
| | 002EH | Reserved |
| | 0030H | Port C3, both input edges |
| | 0032H | Port C2, both input edges |
| | 0034H | Port C1, both input edges |
| Lowest | 0036H | Port C0, both input edges |
| | 0038H | Reserved |

Table 30. Trap and Interrupt Vectors in Order of Priority (Continued)

| Priority | Program Memory Vector Address | Interrupt or Trap Source |
|----------|-------------------------------|--------------------------|
| | 003AH | Reserved |
| | 003CH | Reserved |

Architecture

Figure 8 illustrates the interrupt controller block diagram.

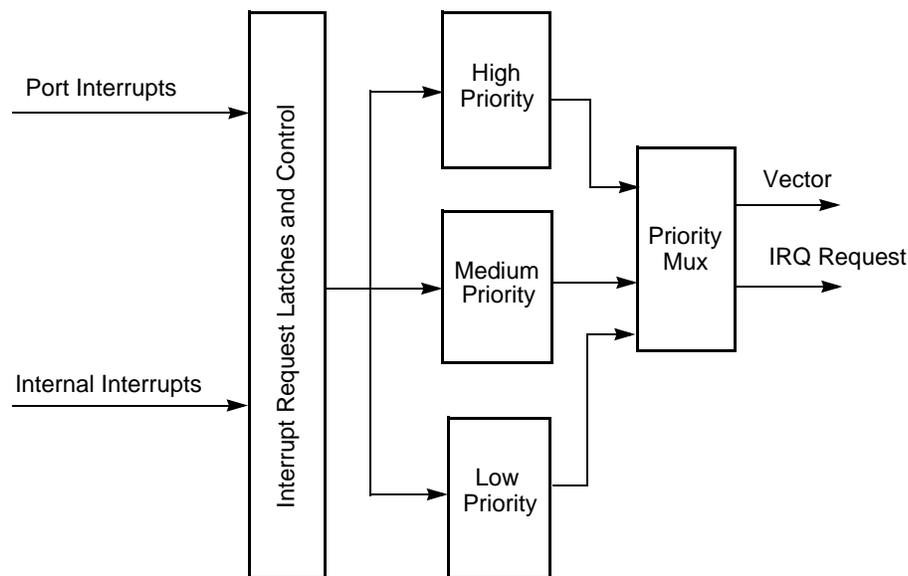


Figure 8. Interrupt Controller Block Diagram

Operation

Master Interrupt Enable

The master interrupt enable bit (IRQE) in the Interrupt Control register globally enables and disables interrupts.

Interrupts are globally enabled by any of the following actions:

- Execution of an EI (Enable Interrupt) instruction

- Execution of an IRET (Return from Interrupt) instruction
- Writing a 1 to the IRQE bit in the Interrupt Control register

Interrupts are globally disabled by any of the following actions:

- Execution of a DI (Disable Interrupt) instruction
- eZ8 CPU acknowledgement of an interrupt service request from the interrupt controller
- Writing a 0 to the IRQE bit in the Interrupt Control register
- Reset
- Execution of a Trap instruction
- Illegal Instruction Trap

Interrupt Vectors and Priority

The interrupt controller supports three levels of interrupt priority. Level 3 is the highest priority, Level 2 is the second highest priority, and Level 1 is the lowest priority. If all of the interrupts are enabled with identical interrupt priority (all as Level 2 interrupts, for example), the interrupt priority is assigned from highest to lowest as specified in Table 30 on page 47. Level 3 interrupts are always assigned higher priority than Level 2 interrupts which, in turn, always are assigned higher priority than Level 1 interrupts. Within each interrupt priority level (Level 1, Level 2, or Level 3), priority is assigned as specified in Table 30, above. Reset, Watch-Dog Timer interrupt (if enabled) and Illegal Instruction Traps always command (level 3) priority.

Interrupt Assertion

Interrupt sources assert their interrupt requests for only a single system clock period (single pulse). When the interrupt request is acknowledged by the eZ8 CPU, the corresponding bit in the Interrupt Request register is cleared until the next interrupt occurs. Writing a 0 to the corresponding bit in the Interrupt Request register likewise clears the interrupt request.



Caution: The following coding style that clears bits in the Interrupt Request registers is **NOT** recommended. All incoming interrupts received between execution of the first LDX command and the final LDX command are lost.

Poor coding style that can result in lost interrupt requests:

```
LDX r0, IRQ0
AND r0, MASK
LDX IRQ0, r0
```

**Caution:**

To avoid missing interrupts, use the following coding style to clear bits in the Interrupt Request 0 register:

Good coding style that avoids lost interrupt requests:

```
ANDX IRQ0, MASK
```

Software Interrupt Assertion

Program code can generate interrupts directly. Writing a 1 to the correct bit in the Interrupt Request register triggers an interrupt (assuming that interrupt is enabled). When the interrupt request is acknowledged by the eZ8 CPU, the bit in the Interrupt Request register is automatically cleared to 0.

**Caution:**

The following coding style used to generate software interrupts by setting bits in the Interrupt Request registers is **NOT** recommended. All incoming interrupts received between execution of the first LDX command and the final LDX command are lost.

Poor coding style that can result in lost interrupt requests:

```
LDX r0, IRQ0  
OR r0, MASK  
LDX IRQ0, r0
```

**Caution:**

To avoid missing interrupts, use the following coding style to set bits in the Interrupt Request registers:

Good coding style that avoids lost interrupt requests:

```
ORX IRQ0, MASK
```

Interrupt Control Register Definitions

For all interrupts other than the Watch-Dog Timer interrupt, the Primary Oscillator Fail Trap, and the Watchdog Oscillator Fail Trap, the interrupt control registers enable individual interrupts, set interrupt priorities, and indicate interrupt requests.

Interrupt Request 0 Register

The Interrupt Request 0 (IRQ0) register (Table 31) stores the interrupt requests for both vectored and polled interrupts. When a request is presented to the interrupt controller, the corresponding bit in the IRQ0 register becomes 1. If interrupts are globally enabled (vectored interrupts), the interrupt controller passes an interrupt request to the eZ8 CPU. If interrupts are globally disabled (polled interrupts), the eZ8 CPU can read the Interrupt Request 0 register to determine if any interrupt requests are pending.



Table 31. Interrupt Request 0 Register (IRQ0)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------------|----------|----------|----------|----------|----------|----------|----------|----------|
| FIELD | Reserved | T1I | T0I | U0RXI | U0TXI | Reserved | Reserved | ADCI |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W |
| ADDR | FC0H | | | | | | | |

Reserved—Must be 0.

T1I—Timer 1 Interrupt Request

0 = No interrupt request is pending for Timer 1.

1 = An interrupt request from Timer 1 is awaiting service.

T0I—Timer 0 Interrupt Request

0 = No interrupt request is pending for Timer 0.

1 = An interrupt request from Timer 0 is awaiting service.

U0RXI—UART 0 Receiver Interrupt Request

0 = No interrupt request is pending for the UART 0 receiver.

1 = An interrupt request from the UART 0 receiver is awaiting service.

U0TXI—UART 0 Transmitter Interrupt Request

0 = No interrupt request is pending for the UART 0 transmitter.

1 = An interrupt request from the UART 0 transmitter is awaiting service.

ADCI—ADC Interrupt Request

0 = No interrupt request is pending for the Analog-to-Digital Converter.

1 = An interrupt request from the Analog-to-Digital Converter is awaiting service.

Interrupt Request 1 Register

The Interrupt Request 1 (IRQ1) register (Table 32) stores interrupt requests for both vectored and polled interrupts. When a request is presented to the interrupt controller, the corresponding bit in the IRQ1 register becomes 1. If interrupts are globally enabled (vectored interrupts), the interrupt controller passes an interrupt request to the eZ8 CPU. If interrupts are globally disabled (polled interrupts), the eZ8 CPU can read the Interrupt Request 1 register to determine if any interrupt requests are pending.



Table 32. Interrupt Request 1 Register (IRQ1)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------|------|------|------|------|------|------|------|
| FIELD | PA7I | PA6I | PA5I | PA4I | PA3I | PA2I | PA1I | PA0I |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | FC3H | | | | | | | |

PA_xI—Port A Pin *x* Interrupt Request

0 = No interrupt request is pending for GPIO Port A pin *x*.

1 = An interrupt request from GPIO Port A pin *x* is awaiting service.

where *x* indicates the specific GPIO Port pin number (0–7).

Interrupt Request 2 Register

The Interrupt Request 2 (IRQ2) register (Table 33) stores interrupt requests for both vectored and polled interrupts. When a request is presented to the interrupt controller, the corresponding bit in the IRQ2 register becomes 1. If interrupts are globally enabled (vectored interrupts), the interrupt controller passes an interrupt request to the eZ8 CPU. If interrupts are globally disabled (polled interrupts), the eZ8 CPU can read the Interrupt Request 2 register to determine if any interrupt requests are pending.

Table 33. Interrupt Request 2 Register (IRQ2)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----------|-----|-----|-----|------|------|------|------|
| FIELD | Reserved | | | | PC3I | PC2I | PC1I | PC0I |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | FC6H | | | | | | | |

Reserved—Must be 0.

PC_xI—Port C Pin *x* Interrupt Request

0 = No interrupt request is pending for GPIO Port C pin *x*.

1 = An interrupt request from GPIO Port C pin *x* is awaiting service.

where *x* indicates the specific GPIO Port C pin number (0–3).

IRQ0 Enable High and Low Bit Registers

Table 34 describes the priority control for IRQ0. The IRQ0 Enable High and Low Bit registers (Tables 35 and 36) form a priority encoded enabling for interrupts in the Interrupt Request 0 register. Priority is generated by setting bits in each register.



Table 34. IRQ0 Enable and Priority Encoding

| IRQ0ENH[x] | IRQ0ENL[x] | Priority | Description |
|------------|------------|----------|-------------|
| 0 | 0 | Disabled | Disabled |
| 0 | 1 | Level 1 | Low |
| 1 | 0 | Level 2 | Nominal |
| 1 | 1 | Level 3 | High |

where x indicates the register bits from 0–7.

Table 35. IRQ0 Enable High Bit Register (IRQ0ENH)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----------|-------|-------|--------|--------|----------|----------|--------|
| FIELD | Reserved | T1ENH | T0ENH | U0RENH | U0TENH | Reserved | Reserved | ADCENH |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | FC1H | | | | | | | |

Reserved—Must be 0.

T1ENH—Timer 1 Interrupt Request Enable High Bit

T0ENH—Timer 0 Interrupt Request Enable High Bit

U0RENH—UART 0 Receive Interrupt Request Enable High Bit

U0TENH—UART 0 Transmit Interrupt Request Enable High Bit

ADCENH—ADC Interrupt Request Enable High Bit

Table 36. IRQ0 Enable Low Bit Register (IRQ0ENL)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----------|-------|-------|--------|--------|----------|----------|--------|
| FIELD | Reserved | T1ENL | T0ENL | U0RENL | U0TENL | Reserved | Reserved | ADCENL |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R | R/W | R/W | R/W | R/W | R | R | R/W |
| ADDR | FC2H | | | | | | | |

Reserved—Must be 0.

T1ENL—Timer 1 Interrupt Request Enable Low Bit

T0ENL—Timer 0 Interrupt Request Enable Low Bit

U0RENL—UART 0 Receive Interrupt Request Enable Low Bit

U0TENL—UART 0 Transmit Interrupt Request Enable Low Bit

ADCENL—ADC Interrupt Request Enable Low Bit



IRQ1 Enable High and Low Bit Registers

Table 37 describes the priority control for IRQ1. The IRQ1 Enable High and Low Bit registers (Tables 38 and 39) form a priority encoded enabling for interrupts in the Interrupt Request 1 register. Priority is generated by setting bits in each register.

Table 37. IRQ1 Enable and Priority Encoding

| IRQ1ENH[x] | IRQ1ENL[x] | Priority | Description |
|------------|------------|----------|-------------|
| 0 | 0 | Disabled | Disabled |
| 0 | 1 | Level 1 | Low |
| 1 | 0 | Level 2 | Nominal |
| 1 | 1 | Level 3 | High |

where x indicates the register bits from 0–7.

Table 38. IRQ1 Enable High Bit Register (IRQ1ENH)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|--------|--------|--------|--------|---------|---------|--------|--------|
| FIELD | PA7ENH | PA6ENH | PA5ENH | PA4ENH | PAD3ENH | PAD2ENH | PA1ENH | PA0ENH |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | FC4H | | | | | | | |

PA_xENH—Port A Bit[x] Interrupt Request Enable High Bit

Refer to the Interrupt Port Select register for selection of either Port A or Port D as the interrupt source.

Table 39. IRQ1 Enable Low Bit Register (IRQ1ENL)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|--------|--------|--------|--------|--------|--------|--------|--------|
| FIELD | PA7ENL | PA6ENL | PA5ENL | PA4ENL | PA3ENL | PA2ENL | PA1ENL | PA0ENL |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | FC5H | | | | | | | |

PA_xENL—Port A Bit[x] Interrupt Request Enable Low Bit



IRQ2 Enable High and Low Bit Registers

Table 40 describes the priority control for IRQ2. The IRQ2 Enable High and Low Bit registers (Tables 41 and 42) form a priority encoded enabling for interrupts in the Interrupt Request 2 register. Priority is generated by setting bits in each register.

Table 40. IRQ2 Enable and Priority Encoding

| IRQ2ENH[x] | IRQ2ENL[x] | Priority | Description |
|------------|------------|----------|-------------|
| 0 | 0 | Disabled | Disabled |
| 0 | 1 | Level 1 | Low |
| 1 | 0 | Level 2 | Nominal |
| 1 | 1 | Level 3 | High |

where x indicates the register bits from 0–7.

Table 41. IRQ2 Enable High Bit Register (IRQ2ENH)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----------|-----|-----|-----|-------|-------|-------|-------|
| FIELD | Reserved | | | | C3ENH | C2ENH | C1ENH | C0ENH |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | FC7H | | | | | | | |

Reserved—Must be 0.

C3ENH—Port C3 Interrupt Request Enable High Bit

C2ENH—Port C2 Interrupt Request Enable High Bit

C1ENH—Port C1 Interrupt Request Enable High Bit

C0ENH—Port C0 Interrupt Request Enable High Bit

Table 42. IRQ2 Enable Low Bit Register (IRQ2ENL)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----------|-----|-----|-----|-------|-------|-------|-------|
| FIELD | Reserved | | | | C3ENL | C2ENL | C1ENL | C0ENL |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | FC8H | | | | | | | |

Reserved—Must be 0.

C3ENL—Port C3 Interrupt Request Enable Low Bit

C2ENL—Port C2 Interrupt Request Enable Low Bit



C1ENL—Port C1 Interrupt Request Enable Low Bit
C0ENL—Port C0 Interrupt Request Enable Low Bit

Interrupt Edge Select Register

The Interrupt Edge Select (IRQES) register (Table 43) determines whether an interrupt is generated for the rising edge or falling edge on the selected GPIO Port A or Port D input pin.

Table 43. Interrupt Edge Select Register (IRQES)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------|------|------|------|------|------|------|------|
| FIELD | IES7 | IES6 | IES5 | IES4 | IES3 | IES2 | IES1 | IES0 |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | FCDH | | | | | | | |

IES_x—Interrupt Edge Select *x*

0 = An interrupt request is generated on the falling edge of the PA_x input or PD_x.

1 = An interrupt request is generated on the rising edge of the PA_x input PD_x.
where *x* indicates the specific GPIO Port pin number (0 through 7).

PA6CS—PA6/Comparator Selection

0 = PA6 is used for the interrupt for PA6CS interrupt request.

1 = The Comparator is used for the interrupt for PA6CS interrupt request.

Reserved—Must be 0.

Interrupt Control Register

The Interrupt Control (IRQCTL) register (Table 44) contains the master enable bit for all interrupts.

Table 44. Interrupt Control Register (IRQCTL)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------|----------|---|---|---|---|---|---|
| FIELD | IRQE | Reserved | | | | | | |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R | R | R | R | R | R | R |
| ADDR | FCFH | | | | | | | |

IRQE—Interrupt Request Enable

This bit is set to 1 by executing an EI (Enable Interrupts) or IRET (Interrupt Return)



instruction, or by a direct register write of a 1 to this bit. It is reset to 0 by executing a DI instruction, eZ8 CPU acknowledgement of an interrupt request, Reset or by a direct register write of a 0 to this bit.

0 = Interrupts are disabled.

1 = Interrupts are enabled.

Reserved—Must be 0.

Timers

Overview

These Z8 Encore!® 4K Series products contain up to two 16-bit reloadable timers that can be used for timing, event counting, or generation of pulse-width modulated (PWM) signals. The timers' features include:

- 16-bit reload counter
- Programmable prescaler with prescale values from 1 to 128
- PWM output generation
- Capture and compare capability
- External input pin for timer input, clock gating, or capture signal. External input pin signal frequency is limited to a maximum of one-fourth the system clock frequency.
- Timer output pin
- Timer interrupt

In addition to the timers described in this chapter, the Baud Rate Generator of the UART (if unused) may also provide basic timing functionality. Refer to chapter “UART” on **page 84** for information about using the Baud Rate Generator as an additional timer.

Architecture

Figure 9 illustrates the architecture of the timers.

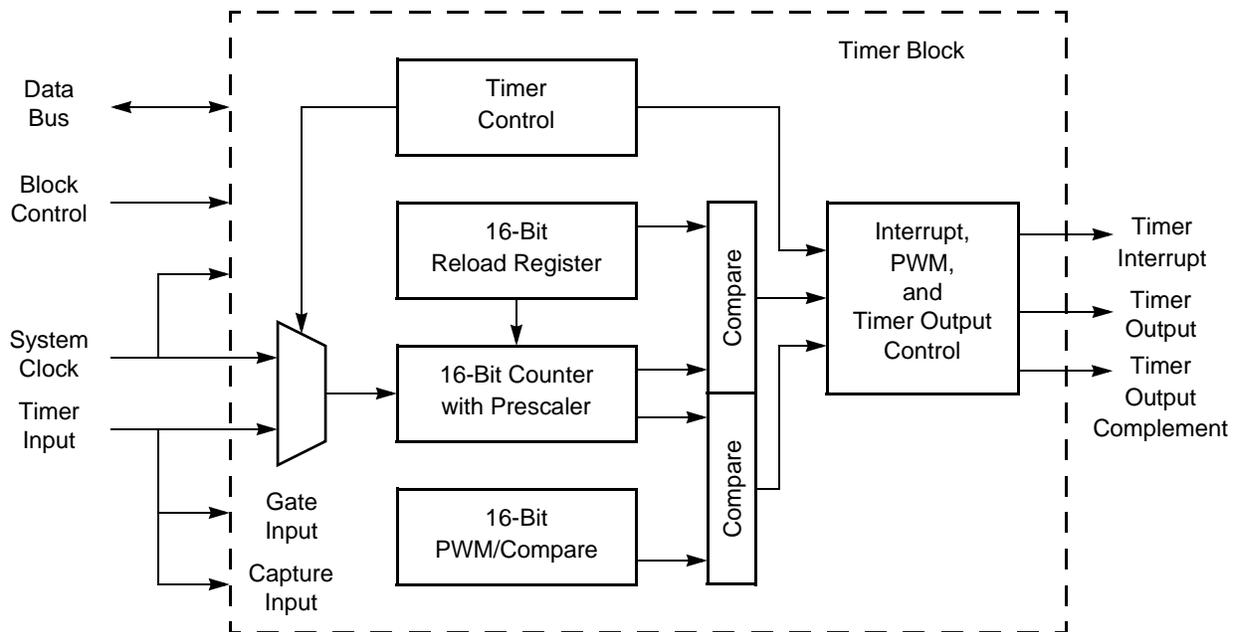


Figure 9. Timer Block Diagram

Operation

The timers are 16-bit up-counters. Minimum time-out delay is set by loading the value 0001H into the Timer Reload High and Low Byte registers and setting the prescale value to 1. Maximum time-out delay is set by loading the value 0000H into the Timer Reload High and Low Byte registers and setting the prescale value to 128. If the Timer reaches FFFFH, the timer rolls over to 0000H and continues counting.

Timer Operating Modes

The timers can be configured to operate in the following modes:

ONE-SHOT Mode

In ONE-SHOT mode, the timer counts up to the 16-bit Reload value stored in the Timer Reload High and Low Byte registers. The timer input is the system clock. Upon reaching the Reload value, the timer generates an interrupt and the count value in the Timer High and Low Byte registers is reset to 0001H. The timer is automatically disabled and stops counting.

Also, if the Timer Output alternate function is enabled, the Timer Output pin changes state for one system clock cycle (from Low to High or from High to Low) upon timer Reload. If



it is appropriate to have the Timer Output make a state change at a One-Shot time-out (rather than a single cycle pulse), first set the TPOL bit in the Timer Control Register to the start value before enabling ONE-SHOT mode. After starting the timer, set TPOL to the opposite bit value.

The steps for configuring a timer for ONE-SHOT mode and initiating the count are as follows:

1. Write to the Timer Control register to:
 - Disable the timer
 - Configure the timer for ONE-SHOT mode.
 - Set the prescale value.
 - Set the initial output level (High or Low) if using the Timer Output alternate function.
2. Write to the Timer High and Low Byte registers to set the starting count value.
3. Write to the Timer Reload High and Low Byte registers to set the Reload value.
4. If appropriate, enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers.
5. If using the Timer Output function, configure the associated GPIO port pin for the Timer Output alternate function.
6. Write to the Timer Control register to enable the timer and initiate counting.

In ONE-SHOT mode, the system clock always provides the timer input. The timer period is given by the following equation:

$$\text{One-Shot Mode Time-Out Period (s)} = \frac{(\text{Reload Value} - \text{Start Value}) \times \text{Prescale}}{\text{System Clock Frequency (Hz)}}$$

CONTINUOUS Mode

In CONTINUOUS mode, the timer counts up to the 16-bit Reload value stored in the Timer Reload High and Low Byte registers. The timer input is the system clock. Upon reaching the Reload value, the timer generates an interrupt, the count value in the Timer High and Low Byte registers is reset to 0001H and counting resumes. Also, if the Timer Output alternate function is enabled, the Timer Output pin changes state (from Low to High or from High to Low) at timer Reload.

The steps for configuring a timer for CONTINUOUS mode and initiating the count are as follows:

1. Write to the Timer Control register to:
 - Disable the timer
 - Configure the timer for CONTINUOUS mode.
 - Set the prescale value.

- If using the Timer Output alternate function, set the initial output level (High or Low).
2. Write to the Timer High and Low Byte registers to set the starting count value (usually 0001H). This action only affects the first pass in CONTINUOUS mode. After the first timer Reload in CONTINUOUS mode, counting always begins at the reset value of 0001H.
 3. Write to the Timer Reload High and Low Byte registers to set the Reload value.
 4. Enable the timer interrupt (if appropriate) and set the timer interrupt priority by writing to the relevant interrupt registers.
 5. Configure the associated GPIO port pin (if using the Timer Output function) for the Timer Output alternate function.
 6. Write to the Timer Control register to enable the timer and initiate counting.

In CONTINUOUS mode, the system clock always provides the timer input. The timer period is given by the following equation:

$$\text{Continuous Mode Time-Out Period (s)} = \frac{\text{Reload Value} \times \text{Prescale}}{\text{System Clock Frequency (Hz)}}$$

If an initial starting value other than 0001H is loaded into the Timer High and Low Byte registers, use the ONE-SHOT mode equation to determine the first time-out period.

COUNTER Mode

In COUNTER mode, the timer counts input transitions from a GPIO port pin. The timer input is taken from the GPIO Port pin Timer Input alternate function. The TPOL bit in the Timer Control Register selects whether the count occurs on the rising edge or the falling edge of the Timer Input signal. In COUNTER mode, the prescaler is disabled.



Caution: The input frequency of the Timer Input signal must not exceed one-fourth the system clock frequency.

Upon reaching the Reload value stored in the Timer Reload High and Low Byte registers, the timer generates an interrupt, the count value in the Timer High and Low Byte registers is reset to 0001H and counting resumes. Also, if the Timer Output alternate function is enabled, the Timer Output pin changes state (from Low to High or from High to Low) at timer Reload.

The steps for configuring a timer for COUNTER mode and initiating the count are as follows:

1. Write to the Timer Control register to:
 - Disable the timer
 - Configure the timer for COUNTER mode

- Select either the rising edge or falling edge of the Timer Input signal for the count. This selection also sets the initial logic level (High or Low) for the Timer Output alternate function. However, the Timer Output function is not required to be enabled.
2. Write to the Timer High and Low Byte registers to set the starting count value. This only affects the first pass in COUNTER mode. After the first timer Reload in COUNTER mode, counting always begins at the reset value of 0001H. In COUNTER mode the Timer High and Low Byte registers must be written with the value 0001H.
 3. Write to the Timer Reload High and Low Byte registers to set the Reload value.
 4. If appropriate, enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers.
 5. Configure the associated GPIO port pin for the Timer Input alternate function.
 6. If using the Timer Output function, configure the associated GPIO port pin for the Timer Output alternate function.
 7. Write to the Timer Control register to enable the timer.

In COUNTER mode, the number of Timer Input transitions since the timer start is given by the following equation:

$$\text{Counter Mode Timer Input Transitions} = \text{Current Count Value} - \text{Start Value}$$

COMPARATOR COUNTER Mode

In COMPARATOR COUNTER mode, the timer counts input transitions from the analog comparator output. The TPOL bit in the Timer Control Register selects whether the count occurs on the rising edge or the falling edge of the comparator output signal. In COMPARATOR COUNTER mode, the prescaler is disabled.



Caution: The frequency of the comparator output signal must not exceed one-fourth the system clock frequency.

After reaching the Reload value stored in the Timer Reload High and Low Byte registers, the timer generates an interrupt, the count value in the Timer High and Low Byte registers is reset to 0001H and counting resumes. Also, if the Timer Output alternate function is enabled, the Timer Output pin changes state (from Low to High or from High to Low) at timer Reload.

The steps for configuring a timer for COMPARATOR COUNTER mode and initiating the count are as follows:

1. Write to the Timer Control register to:
 - Disable the timer
 - Configure the timer for COMPARATOR COUNTER mode



- Select either the rising edge or falling edge of the comparator output signal for the count. This also sets the initial logic level (High or Low) for the Timer Output alternate function. However, the Timer Output function is not required to be enabled.
2. Write to the Timer High and Low Byte registers to set the starting count value. This action only affects the first pass in COMPARATOR COUNTER mode. After the first timer Reload in COMPARATOR COUNTER mode, counting always begins at the reset value of 0001H. Generally, in COMPARATOR COUNTER mode the Timer High and Low Byte registers must be written with the value 0001H.
 3. Write to the Timer Reload High and Low Byte registers to set the Reload value.
 4. If appropriate, enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers.
 5. If using the Timer Output function, configure the associated GPIO port pin for the Timer Output alternate function.
 6. Write to the Timer Control register to enable the timer.

In COMPARATOR COUNTER mode, the number of comparator output transitions since the timer start is given by the following equation:

$$\text{Comparator Output Transitions} = \text{Current Count Value} - \text{Start Value}$$

PWM SINGLE OUTPUT Mode

In PWM SINGLE OUTPUT mode, the timer outputs a Pulse-Width Modulator (PWM) output signal through a GPIO Port pin. The timer input is the system clock. The timer first counts up to the 16-bit PWM match value stored in the Timer PWM High and Low Byte registers. When the timer count value matches the PWM value, the Timer Output toggles. The timer continues counting until it reaches the Reload value stored in the Timer Reload High and Low Byte registers. Upon reaching the Reload value, the timer generates an interrupt, the count value in the Timer High and Low Byte registers is reset to 0001H and counting resumes.

If the TPOL bit in the Timer Control register is set to 1, the Timer Output signal begins as a High (1) and transitions to a Low (0) when the timer value matches the PWM value. The Timer Output signal returns to a High (1) after the timer reaches the Reload value and is reset to 0001H.

If the TPOL bit in the Timer Control register is set to 0, the Timer Output signal begins as a Low (0) and transitions to a High (1) when the timer value matches the PWM value. The Timer Output signal returns to a Low (0) after the timer reaches the Reload value and is reset to 0001H.

The steps for configuring a timer for PWM Single Output mode and initiating the PWM operation are as follows:



1. Write to the Timer Control register to:
 - Disable the timer
 - Configure the timer for PWM mode.
 - Set the prescale value.
 - Set the initial logic level (High or Low) and PWM High/Low transition for the Timer Output alternate function.
2. Write to the Timer High and Low Byte registers to set the starting count value (typically 0001H). This only affects the first pass in PWM mode. After the first timer reset in PWM mode, counting always begins at the reset value of 0001H.
3. Write to the PWM High and Low Byte registers to set the PWM value.
4. Write to the Timer Reload High and Low Byte registers to set the Reload value (PWM period). The Reload value must be greater than the PWM value.
5. If appropriate, enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers.
6. Configure the associated GPIO port pin for the Timer Output alternate function.
7. Write to the Timer Control register to enable the timer and initiate counting.

The PWM period is represented by the following equation:

$$\text{PWM Period (s)} = \frac{\text{Reload Value} \times \text{Prescale}}{\text{System Clock Frequency (Hz)}}$$

If an initial starting value other than 0001H is loaded into the Timer High and Low Byte registers, use the ONE-SHOT mode equation to determine the first PWM time-out period.

If TPOL is set to 0, the ratio of the PWM output High time to the total period is represented by:

$$\text{PWM Output High Time Ratio (\%)} = \frac{\text{Reload Value} - \text{PWM Value}}{\text{Reload Value}} \times 100$$

If TPOL is set to 1, the ratio of the PWM output High time to the total period is represented by:

$$\text{PWM Output High Time Ratio (\%)} = \frac{\text{PWM Value}}{\text{Reload Value}} \times 100$$

PWM Dual Output Mode

In PWM DUAL OUTPUT mode, the timer outputs a Pulse-Width Modulated (PWM) output signal pair (basic PWM signal and its complement) through two GPIO Port pins. The timer input is the system clock. The timer first counts up to the 16-bit PWM match value



stored in the Timer PWM High and Low Byte registers. When the timer count value matches the PWM value, the Timer Output toggles. The timer continues counting until it reaches the Reload value stored in the Timer Reload High and Low Byte registers. Upon reaching the Reload value, the timer generates an interrupt, the count value in the Timer High and Low Byte registers is reset to 0001H and counting resumes.

If the TPOL bit in the Timer Control register is set to 1, the Timer Output signal begins as a High (1) and transitions to a Low (0) when the timer value matches the PWM value. The Timer Output signal returns to a High (1) after the timer reaches the Reload value and is reset to 0001H.

If the TPOL bit in the Timer Control register is set to 0, the Timer Output signal begins as a Low (0) and transitions to a High (1) when the timer value matches the PWM value. The Timer Output signal returns to a Low (0) after the timer reaches the Reload value and is reset to 0001H.

The timer also generates a second PWM output signal Timer Output Complement. The Timer Output Complement is the complement of the Timer Output PWM signal. A programmable deadband delay can be configured to time delay (0 to 128 system clock cycles) PWM output transitions on these two pins from a low to a high (inactive to active). This ensures a time gap between the deassertion of one PWM output to the assertion of its complement.

The steps for configuring a timer for PWM Dual Output mode and initiating the PWM operation are as follows:

1. Write to the Timer Control register to:
 - Disable the timer
 - Configure the timer for PWM Dual Output mode. Setting the mode also involves writing to TMODEHI bit in TxCTL1 register.
 - Set the prescale value.
 - Set the initial logic level (High or Low) and PWM High/Low transition for the Timer Output alternate function.
2. Write to the Timer High and Low Byte registers to set the starting count value (typically 0001H). This only affects the first pass in PWM mode. After the first timer reset in PWM mode, counting always begins at the reset value of 0001H.
3. Write to the PWM High and Low Byte registers to set the PWM value.
4. Write to the PWM Control register to set the PWM dead band delay value. The deadband delay must be less than the duration of the positive phase of the PWM signal (as defined by the PWM high and low byte registers). It must also be less than the duration of the negative phase of the PWM signal (as defined by the difference between the PWM registers and the Timer Reload registers).
5. Write to the Timer Reload High and Low Byte registers to set the Reload value (PWM period). The Reload value must be greater than the PWM value.



6. If appropriate, enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers.
7. Configure the associated GPIO port pin for the Timer Output and Timer Output Complement alternate functions. The Timer Output Complement function is shared with the Timer Input function for both timers. Setting the timer mode to Dual PWM automatically switches the function from Timer In to Timer Out Complement.
8. Write to the Timer Control register to enable the timer and initiate counting.

The PWM period is represented by the following equation:

$$\text{PWM Period (s)} = \frac{\text{Reload Value} \times \text{Prescale}}{\text{System Clock Frequency (Hz)}}$$

If an initial starting value other than 0001H is loaded into the Timer High and Low Byte registers, the ONE-SHOT mode equation determines the first PWM time-out period.

If TPOL is set to 0, the ratio of the PWM output High time to the total period is represented by:

$$\text{PWM Output High Time Ratio (\%)} = \frac{\text{Reload Value} - \text{PWM Value}}{\text{Reload Value}} \times 100$$

If TPOL is set to 1, the ratio of the PWM output High time to the total period is represented by:

$$\text{PWM Output High Time Ratio (\%)} = \frac{\text{PWM Value}}{\text{Reload Value}} \times 100$$

CAPTURE Mode

In CAPTURE mode, the current timer count value is recorded when the appropriate external Timer Input transition occurs. The Capture count value is written to the Timer PWM High and Low Byte Registers. The timer input is the system clock. The TPOL bit in the Timer Control register determines if the Capture occurs on a rising edge or a falling edge of the Timer Input signal. When the Capture event occurs, an interrupt is generated and the timer continues counting. The INPCAP bit in TxCTL1 register is set to indicate the timer interrupt is because of an input capture event.

The timer continues counting up to the 16-bit Reload value stored in the Timer Reload High and Low Byte registers. Upon reaching the Reload value, the timer generates an interrupt and continues counting. The INPCAP bit in TxCTL1 register clears indicating the timer interrupt is not because of an input capture event.

The steps for configuring a timer for CAPTURE mode and initiating the count are as follows:

1. Write to the Timer Control register to:
 - Disable the timer
 - Configure the timer for CAPTURE mode.
 - Set the prescale value.
 - Set the Capture edge (rising or falling) for the Timer Input.
2. Write to the Timer High and Low Byte registers to set the starting count value (typically 0001H).
3. Write to the Timer Reload High and Low Byte registers to set the Reload value.
4. Clear the Timer PWM High and Low Byte registers to 0000H. Clearing these registers allows user software to determine if interrupts were generated by either a capture event or a reload. If the PWM High and Low Byte registers still contain 0000H after the interrupt, the interrupt was generated by a Reload.
5. Enable the timer interrupt, if appropriate, and set the timer interrupt priority by writing to the relevant interrupt registers. By default, the timer interrupt is generated for both input capture and reload events. If appropriate, configure the timer interrupt to be generated only at the input capture event or the reload event by setting TICONFIG field of the TxCTL1 register.
6. Configure the associated GPIO port pin for the Timer Input alternate function.
7. Write to the Timer Control register to enable the timer and initiate counting.

In CAPTURE mode, the elapsed time from timer start to Capture event can be calculated using the following equation:

$$\text{Capture Elapsed Time (s)} = \frac{(\text{Capture Value} - \text{Start Value}) \times \text{Prescale}}{\text{System Clock Frequency (Hz)}}$$

CAPTURE RESTART Mode

In CAPTURE RESTART mode, the current timer count value is recorded when the acceptable external Timer Input transition occurs. The Capture count value is written to the Timer PWM High and Low Byte Registers. The timer input is the system clock. The TPOL bit in the Timer Control register determines if the Capture occurs on a rising edge or a falling edge of the Timer Input signal. When the Capture event occurs, an interrupt is generated and the count value in the Timer High and Low Byte registers is reset to 0001H and counting resumes. The INPCAP bit in TxCTL1 register is set to indicate the timer interrupt is because of an input capture event.

If no Capture event occurs, the timer counts up to the 16-bit Compare value stored in the Timer Reload High and Low Byte registers. Upon reaching the Reload value, the timer generates an interrupt, the count value in the Timer High and Low Byte registers is reset to 0001H and counting resumes. The INPCAP bit in TxCTL1 register is cleared to indicate the timer interrupt is not caused by an input capture event.



The steps for configuring a timer for CAPTURE RESTART mode and initiating the count are as follows:

1. Write to the Timer Control register to:
 - Disable the timer
 - Configure the timer for CAPTURE RESTART mode. Setting the mode also involves writing to TMODEHI bit in TxCTL1 register.
 - Set the prescale value.
 - Set the Capture edge (rising or falling) for the Timer Input.
2. Write to the Timer High and Low Byte registers to set the starting count value (typically 0001H).
3. Write to the Timer Reload High and Low Byte registers to set the Reload value.
4. Clear the Timer PWM High and Low Byte registers to 0000H. This allows user software to determine if interrupts were generated by either a capture event or a reload. If the PWM High and Low Byte registers still contain 0000H after the interrupt, the interrupt was generated by a Reload.
5. Enable the timer interrupt, if appropriate, and set the timer interrupt priority by writing to the relevant interrupt registers. By default, the timer interrupt is generated for both input capture and reload events. If appropriate, configure the timer interrupt to be generated only at the input capture event or the reload event by setting TICONFIG field of the TxCTL1 register.
6. Configure the associated GPIO port pin for the Timer Input alternate function.
7. Write to the Timer Control register to enable the timer and initiate counting.

In CAPTURE mode, the elapsed time from timer start to Capture event can be calculated using the following equation:

$$\text{Capture Elapsed Time (s)} = \frac{(\text{Capture Value} - \text{Start Value}) \times \text{Prescale}}{\text{System Clock Frequency (Hz)}}$$

COMPARE Mode

In COMPARE mode, the timer counts up to the 16-bit maximum Compare value stored in the Timer Reload High and Low Byte registers. The timer input is the system clock. Upon reaching the Compare value, the timer generates an interrupt and counting continues (the timer value is not reset to 0001H). Also, if the Timer Output alternate function is enabled, the Timer Output pin changes state (from Low to High or from High to Low) upon Compare.

If the Timer reaches FFFFH, the timer rolls over to 0000H and continue counting.

The steps for configuring a timer for COMPARE mode and initiating the count are as follows:



1. Write to the Timer Control register to:
 - Disable the timer
 - Configure the timer for Compare mode.
 - Set the prescale value.
 - Set the initial logic level (High or Low) for the Timer Output alternate function, if appropriate.
2. Write to the Timer High and Low Byte registers to set the starting count value.
3. Write to the Timer Reload High and Low Byte registers to set the Compare value.
4. Enable the timer interrupt, if appropriate, and set the timer interrupt priority by writing to the relevant interrupt registers.
5. If using the Timer Output function, configure the associated GPIO port pin for the Timer Output alternate function.
6. Write to the Timer Control register to enable the timer and initiate counting.

In Compare mode, the system clock always provides the timer input. The Compare time can be calculated by the following equation:

$$\text{Compare Mode Time (s)} = \frac{(\text{Compare Value} - \text{Start Value}) \times \text{Prescale}}{\text{System Clock Frequency (Hz)}}$$

GATED Mode

In GATED mode, the timer counts only when the Timer Input signal is in its active state (asserted), as determined by the TPOL bit in the Timer Control register. When the Timer Input signal is asserted, counting begins. A timer interrupt is generated when the Timer Input signal is deasserted or a timer reload occurs. To determine if a Timer Input signal deassertion generated the interrupt, read the associated GPIO input value and compare to the value stored in the TPOL bit.

The timer counts up to the 16-bit Reload value stored in the Timer Reload High and Low Byte registers. The timer input is the system clock. When reaching the Reload value, the timer generates an interrupt, the count value in the Timer High and Low Byte registers is reset to 0001H and counting resumes (assuming the Timer Input signal remains asserted). Also, if the Timer Output alternate function is enabled, the Timer Output pin changes state (from Low to High or from High to Low) at timer reset.

The steps for configuring a timer for GATED mode and initiating the count are as follows:

1. Write to the Timer Control register to:
 - Disable the timer
 - Configure the timer for Gated mode.
 - Set the prescale value.

2. Write to the Timer High and Low Byte registers to set the starting count value. Writing these registers only affects the first pass in GATED mode. After the first timer reset in GATED mode, counting always begins at the reset value of 0001H.
3. Write to the Timer Reload High and Low Byte registers to set the Reload value.
4. Enable the timer interrupt, if appropriate, and set the timer interrupt priority by writing to the relevant interrupt registers. By default, the timer interrupt is generated for both input deassertion and reload events. If appropriate, configure the timer interrupt to be generated only at the input deassertion event or the reload event by setting TICONFIG field of the TxCTL1 register.
5. Configure the associated GPIO port pin for the Timer Input alternate function.
6. Write to the Timer Control register to enable the timer.
7. Assert the Timer Input signal to initiate the counting.

CAPTURE/COMPARE Mode

In CAPTURE/COMPARE mode, the timer begins counting on the first external Timer Input transition. The acceptable transition (rising edge or falling edge) is set by the TPOL bit in the Timer Control Register. The timer input is the system clock.

Every subsequent acceptable transition (after the first) of the Timer Input signal captures the current count value. The Capture value is written to the Timer PWM High and Low Byte Registers. When the Capture event occurs, an interrupt is generated, the count value in the Timer High and Low Byte registers is reset to 0001H, and counting resumes. The INPCAP bit in TxCTL1 register is set to indicate the timer interrupt is caused by an input capture event.

If no Capture event occurs, the timer counts up to the 16-bit Compare value stored in the Timer Reload High and Low Byte registers. Upon reaching the Compare value, the timer generates an interrupt, the count value in the Timer High and Low Byte registers is reset to 0001H and counting resumes. The INPCAP bit in TxCTL1 register is cleared to indicate the timer interrupt is not because of an input capture event.

The steps for configuring a timer for CAPTURE/COMPARE mode and initiating the count are as follows:

1. Write to the Timer Control register to:
 - Disable the timer
 - Configure the timer for CAPTURE/COMPARE mode.
 - Set the prescale value.
 - Set the Capture edge (rising or falling) for the Timer Input.
2. Write to the Timer High and Low Byte registers to set the starting count value (typically 0001H).
3. Write to the Timer Reload High and Low Byte registers to set the Compare value.

4. Enable the timer interrupt, if appropriate, and set the timer interrupt priority by writing to the relevant interrupt registers. By default, the timer interrupt are generated for both input capture and reload events. If appropriate, configure the timer interrupt to be generated only at the input capture event or the reload event by setting TICONFIG field of the TxCTL1 register.
5. Configure the associated GPIO port pin for the Timer Input alternate function.
6. Write to the Timer Control register to enable the timer.
7. Counting begins on the first appropriate transition of the Timer Input signal. No interrupt is generated by this first edge.

In CAPTURE/COMPARE mode, the elapsed time from timer start to Capture event can be calculated using the following equation:

$$\text{Capture Elapsed Time (s)} = \frac{(\text{Capture Value} - \text{Start Value}) \times \text{Prescale}}{\text{System Clock Frequency (Hz)}}$$

Reading the Timer Count Values

The current count value in the timers can be read while counting (enabled). This capability has no effect on timer operation. When the timer is enabled and the Timer High Byte register is read, the contents of the Timer Low Byte register are placed in a holding register. A subsequent read from the Timer Low Byte register returns the value in the holding register. This operation allows accurate reads of the full 16-bit timer count value while enabled. When the timers are not enabled, a read from the Timer Low Byte register returns the actual value in the counter.

Timer Pin Signal Operation

Timer Output is a GPIO Port pin alternate function. The Timer Output is toggled every time the counter is reloaded.

The timer input can be used as a selectable counting source. It shares the same pin as the complementary timer output. When selected by the GPIO Alternate Function Registers, this pin functions as a timer input in all modes except for the DUAL PWM OUTPUT mode. For this mode, there is no timer input available.



Timer Control Register Definitions

Timer 0–1 High and Low Byte Registers

The Timer 0–1 High and Low Byte (TxH and TxL) registers (Tables 45 and 39) contain the current 16-bit timer count value. When the timer is enabled, a read from TxH causes the value in TxL to be stored in a temporary holding register. A read from TxL always returns this temporary register when the timers are enabled. When the timer is disabled, reads from the TxL reads the register directly.

Writing to the Timer High and Low Byte registers while the timer is enabled is not recommended. There are no temporary holding registers available for write operations, so simultaneous 16-bit writes are not possible. If either the Timer High or Low Byte registers are written during counting, the 8-bit written value is placed in the counter (High or Low Byte) at the next clock edge. The counter continues counting from the new value.

Table 45. Timer 0–1 High Byte Register (TxH)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------------|-----|-----|-----|-----|-----|-----|-----|
| FIELD | TH | | | | | | | |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | F00H, F08H | | | | | | | |

Table 46. Timer 0–1 Low Byte Register (TxL)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------------|-----|-----|-----|-----|-----|-----|-----|
| FIELD | TL | | | | | | | |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | F01H, F09H | | | | | | | |

TH and TL—Timer High and Low Bytes

These 2 bytes, {TH[7:0], TL[7:0]}, contain the current 16-bit timer count value.

Timer Reload High and Low Byte Registers

The Timer 0–1 Reload High and Low Byte (TxRH and TxRL) registers (Tables 47 and 41) store a 16-bit reload value, {TRH[7:0], TRL[7:0]}. Values written to the Timer Reload High Byte register are stored in a temporary holding register. When a write to the Timer Reload Low Byte register occurs, the temporary holding register value is written to the Timer High Byte register. This operation allows simultaneous updates of the 16-bit Timer



Reload value.
In COMPARE mode, the Timer Reload High and Low Byte registers store the 16-bit Compare value.

Table 47. Timer 0–1 Reload High Byte Register (TxRH)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------------|-----|-----|-----|-----|-----|-----|-----|
| FIELD | TRH | | | | | | | |
| RESET | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | F02H, F0AH | | | | | | | |

Table 48. Timer 0–1 Reload Low Byte Register (TxRL)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------------|-----|-----|-----|-----|-----|-----|-----|
| FIELD | TRL | | | | | | | |
| RESET | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | F03H, F0BH | | | | | | | |

TRH and TRL—Timer Reload Register High and Low
These two bytes form the 16-bit Reload value, {TRH[7:0], TRL[7:0]}. This value sets the maximum count value which initiates a timer reload to 0001H. In Compare mode, these two bytes form the 16-bit Compare value.

Timer 0-1 PWM High and Low Byte Registers

The Timer 0-1 PWM High and Low Byte (TxPWMH and TxPWML) registers (Tables 49 and 43) control Pulse-Width Modulator (PWM) operations. These registers also store the Capture values for the CAPTURE and CAPTURE/COMPARE modes.

Table 49. Timer 0–1 PWM High Byte Register (TxPWMH)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------------|-----|-----|-----|-----|-----|-----|-----|
| FIELD | PWMH | | | | | | | |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | F04H, F0CH | | | | | | | |



Table 50. Timer 0–1 PWM Low Byte Register (TxPWML)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------------|-----|-----|-----|-----|-----|-----|-----|
| FIELD | PWML | | | | | | | |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | F05H, F0DH | | | | | | | |

PWMH and PWML—Pulse-Width Modulator High and Low Bytes

These two bytes, {PWMH[7:0], PWML[7:0]}, form a 16-bit value that is compared to the current 16-bit timer count. When a match occurs, the PWM output changes state. The PWM output value is set by the TPOL bit in the Timer Control Register (TxCTL1) register.

The TxPWMH and TxPWML registers also store the 16-bit captured timer value when operating in Capture or Capture/Compare modes.

Timer 0–1 Control Registers

Time 0–1 Control Register 0

The Timer Control Register 0 (TxCTL0) and Timer Control Register 1 (TxCTL1) determine the timer operating mode. It also includes a programmable PWM deadband delay, two bits to configure timer interrupt definition, and a status bit to identify if the most recent timer interrupt is caused by an input capture event.

Table 51. Timer 0–1 Control Register 0 (TxCTL0)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------------|----------|-----|----------|------|-----|-----|--------|
| FIELD | TMODEHI | TICONFIG | | Reserved | PWMD | | | INPCAP |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | F06H, F0EH | | | | | | | |

TMODEHI—Timer Mode High Bit

This bit along with the TMODE field in TxCTL1 register determines the operating mode of the timer. This is the most significant bit of the Timer mode selection value. See the TxCTL1 register description on the next page for additional details.

TICONFIG—Timer Interrupt Configuration

This field configures timer interrupt definition.



- 0x = Timer Interrupt occurs on all defined Reload, Compare and Input Events
- 10 = Timer Interrupt only on defined Input Capture/Deassertion Events
- 11 = Timer Interrupt only on defined Reload/Compare Events

Reserved—Must be 0.

PWMD—PWM Delay value

This field is a programmable delay to control the number of system clock cycles delay before the Timer Output and the Timer Output Complement are forced to their active state.

- 000 = No delay
- 001 = 2 cycles delay
- 010 = 4 cycles delay
- 011 = 8 cycles delay
- 100 = 16 cycles delay
- 101 = 32 cycles delay
- 110 = 64 cycles delay
- 111 = 128 cycles delay

INPCAP—Input Capture Event

This bit indicates if the most recent timer interrupt is caused by a Timer Input Capture Event.

- 0 = Previous timer interrupt is not a result of Timer Input Capture Event
- 1 = Previous timer interrupt is a result of Timer Input Capture Event

Timer 0–1 Control Register 1

The Timer 0–1 Control (TxCTL1) registers enable/disable the timers, set the prescaler value, and determine the timer operating mode.

Table 52. Timer 0–1 Control Register 1 (TxCTL1)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------------|------|------|-----|-----|-------|-----|-----|
| FIELD | TEN | TPOL | PRES | | | TMODE | | |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | F07H, F0FH | | | | | | | |

TEN—Timer Enable

0 = Timer is disabled.

1 = Timer enabled to count.

TPOL—Timer Input/Output Polarity

Operation of this bit is a function of the current operating mode of the timer.

ONE-SHOT mode

When the timer is disabled, the Timer Output signal is set to the value of this bit.

When the timer is enabled, the Timer Output signal is complemented upon timer Reload.

CONTINUOUS mode

When the timer is disabled, the Timer Output signal is set to the value of this bit. When the timer is enabled, the Timer Output signal is complemented upon timer Reload.

COUNTER mode

When the timer is disabled, the Timer Output signal is set to the value of this bit. When the timer is enabled, the Timer Output signal is complemented upon timer Reload.

PWM SINGLE OUTPUT mode

0 = Timer Output is forced Low (0) when the timer is disabled. When enabled, the Timer Output is forced High (1) upon PWM count match and forced Low (0) upon Reload.

1 = Timer Output is forced High (1) when the timer is disabled. When enabled, the Timer Output is forced Low (0) upon PWM count match and forced High (1) upon Reload.

CAPTURE mode

0 = Count is captured on the rising edge of the Timer Input signal.
1 = Count is captured on the falling edge of the Timer Input signal.

COMPARE mode

When the timer is disabled, the Timer Output signal is set to the value of this bit. When the timer is enabled, the Timer Output signal is complemented upon timer Reload.

GATED mode

0 = Timer counts when the Timer Input signal is High (1) and interrupts are generated on the falling edge of the Timer Input.

1 = Timer counts when the Timer Input signal is Low (0) and interrupts are generated on the rising edge of the Timer Input.

CAPTURE/COMPARE mode

0 = Counting is started on the first rising edge of the Timer Input signal. The current count is captured on subsequent rising edges of the Timer Input signal.

1 = Counting is started on the first falling edge of the Timer Input signal. The current count is captured on subsequent falling edges of the Timer Input signal.

PWM DUAL OUTPUT mode

0 = Timer Output is forced Low (0) and Timer Output Complement is forced High (1) when the timer is disabled. When enabled, the Timer Output is forced High (1) upon PWM count match and forced Low (0) upon Reload. When enabled, the Timer Output Complement is forced Low (0) upon PWM count match and forced High (1) upon Reload. The PWMD field in TxCTL0 register is a programmable delay to control the



number of cycles time delay before the Timer Output and the Timer Output Complement is forced to High (1).

1 = Timer Output is forced High (1) and Timer Output Complement is forced Low (0) when the timer is disabled. When enabled, the Timer Output is forced Low (0) upon PWM count match and forced High (1) upon Reload. When enabled, the Timer Output Complement is forced High (1) upon PWM count match and forced Low (0) upon Reload. The PWMD field in TxCTL0 register is a programmable delay to control the number of cycles time delay before the Timer Output and the Timer Output Complement is forced to Low (0).

CAPTURE RESTART mode

0 = Count is captured on the rising edge of the Timer Input signal.
1 = Count is captured on the falling edge of the Timer Input signal.

COMPARATOR COUNTER mode

When the timer is disabled, the Timer Output signal is set to the value of this bit. When the timer is enabled, the Timer Output signal is complemented upon timer Reload.

PRES—Prescale value.

The timer input clock is divided by 2^{PRES} , where PRES can be set from 0 to 7. The prescaler is reset each time the Timer is disabled. This reset ensures proper clock division each time the Timer is restarted.

- 000 = Divide by 1
- 001 = Divide by 2
- 010 = Divide by 4
- 011 = Divide by 8
- 100 = Divide by 16
- 101 = Divide by 32
- 110 = Divide by 64
- 111 = Divide by 128

TMODE—Timer mode

This field along with the TMODEHI bit in TxCTL0 register determines the operating mode of the timer. TMODEHI is the most significant bit of the Timer mode selection value.

- 0000 = One-Shot mode
- 0001 = Continuous mode
- 0010 = Counter mode
- 0011 = PWM Single Output mode
- 0100 = Capture mode
- 0101 = Compare mode
- 0110 = Gated mode
- 0111 = Capture/Compare mode
- 1000 = PWM Dual Output mode



1001 = Capture Restart mode
1010 = Comparator Counter Mode

Watch-Dog Timer

Overview

The Watch-Dog Timer (WDT) protects against corrupt or unreliable software, power faults, and other system-level problems which may place the 4K Series devices into unsuitable operating states. The Watch-Dog Timer includes the following features:

- On-chip RC oscillator
- A selectable time-out response: reset or interrupt
- 24-bit programmable time-out value

Operation

The Watch-Dog Timer (WDT) is a retriggerable one-shot timer that resets or interrupts the 4K Series devices when the WDT reaches its terminal count. The Watch-Dog Timer uses a dedicated on-chip RC oscillator as its clock source. The Watch-Dog Timer operates in only two modes: ON and OFF. Once enabled, it always counts and must be refreshed to prevent a time-out. Perform an enable by executing the WDT instruction or by setting the WDT_AO Flash Option Bit. The WDT_AO bit forces the Watch-Dog Timer to operate immediately upon reset, even if a WDT instruction has not been executed.

The Watch-Dog Timer is a 24-bit reloadable downcounter that uses three 8-bit registers in the eZ8 CPU register space to set the reload value. The nominal WDT time-out period is described by the following equation:

$$\text{WDT Time-out Period (ms)} = \frac{\text{WDT Reload Value}}{10}$$

where the WDT reload value is the decimal value of the 24-bit value given by {WDTU[7:0], WDTH[7:0], WDTL[7:0]} and the typical Watch-Dog Timer RC oscillator frequency is 10KHz. The Watch-Dog Timer cannot be refreshed after it reaches 000002H. The WDT Reload Value must not be set to values below 000004H. Table 53 provides information about approximate time-out delays for the minimum and maximum WDT reload values.



Table 53. Watch-Dog Timer Approximate Time-Out Delays

| WDT Reload Value (Hex) | WDT Reload Value (Decimal) | Approximate Time-Out Delay (with 10KHz typical WDT oscillator frequency) | |
|---------------------------|-------------------------------|---|------------------------|
| | | Typical | Description |
| 000004 | 4 | 400 μ s | Minimum time-out delay |
| FFFFFF | 16,777,215 | 28 minutes | Maximum time-out delay |

Watch-Dog Timer Refresh

When first enabled, the Watch-Dog Timer is loaded with the value in the Watch-Dog Timer Reload registers. The Watch-Dog Timer counts down to 000000H unless a WDT instruction is executed by the eZ8 CPU. Execution of the WDT instruction causes the downcounter to be reloaded with the WDT Reload value stored in the Watch-Dog Timer Reload registers. Counting resumes following the reload operation.

When the 4K Series devices are operating in DEBUG Mode (using the On-Chip Debugger), the Watch-Dog Timer is continuously refreshed to prevent any Watch-Dog Timer time-outs.

Watch-Dog Timer Time-Out Response

The Watch-Dog Timer times out when the counter reaches 000000H. A time-out of the Watch-Dog Timer generates either an interrupt or a system reset. The WDT_RES Flash Option Bit determines the time-out response of the Watch-Dog Timer. Refer to the chapter “Flash Option Bits” on page 126 for information regarding programming of the WDT_RES Flash Option Bit.

WDT Interrupt in Normal Operation

If configured to generate an interrupt when a time-out occurs, the Watch-Dog Timer issues an interrupt request to the interrupt controller and sets the WDT status bit in the Watch-Dog Timer Control register. If interrupts are enabled, the eZ8 CPU responds to the interrupt request by fetching the Watch-Dog Timer interrupt vector and executing code from the vector address. After time-out and interrupt generation, the Watch-Dog Timer counter rolls over to its maximum value of FFFFFH and continues counting. The Watch-Dog Timer counter is not automatically returned to its Reload Value.

The Reset Status Register (page 26) must be read before clearing the WDT interrupt. This read clears the WDT timeout flag and prevents further WDT interrupts for immediately occurring.



WDT Interrupt in STOP Mode

If configured to generate an interrupt when a time-out occurs and the 4K Series devices are in STOP mode, the Watch-Dog Timer automatically initiates a STOP Mode Recovery and generates an interrupt request. Both the WDT status bit and the STOP bit in the Watch-Dog Timer Control register are set to 1 following a WDT time-out in STOP mode. Refer to the chapter “**Reset, STOP Mode Recovery and Low Voltage Detection**” on **page 20** for more information about STOP Mode Recovery.

If interrupts are enabled, following completion of the STOP Mode Recovery the eZ8 CPU responds to the interrupt request by fetching the Watch-Dog Timer interrupt vector and executing code from the vector address.

WDT Reset in NORMAL Operation

If configured to generate a Reset when a time-out occurs, the Watch-Dog Timer forces the device into the System Reset state. The WDT status bit in the Watch-Dog Timer Control register is set to 1. Refer to the chapter “**Reset, STOP Mode Recovery and Low Voltage Detection**” on **page 20** for more information about system reset.

WDT Reset in STOP Mode

If configured to generate a Reset when a time-out occurs and the device is in STOP mode, the Watch-Dog Timer initiates a STOP Mode Recovery. Both the WDT status bit and the STOP bit in the Watch-Dog Timer Control register are set to 1 following WDT time-out in STOP mode. Refer to the chapter “**Reset, STOP Mode Recovery and Low Voltage Detection**” on **page 20** for more information.

Watch-Dog Timer Reload Unlock Sequence

Writing the unlock sequence to the Watch-Dog Timer (WDTCTL) Control register address unlocks the three Watch-Dog Timer Reload Byte registers (WDTU, WDTH, and WDTL) to allow changes to the time-out period. These write operations to the WDTCTL register address produce no effect on the bits in the WDTCTL register. The locking mechanism prevents spurious writes to the Reload registers. The following sequence is required to unlock the Watch-Dog Timer Reload Byte registers (WDTU, WDTH, and WDTL) for write access.

1. Write 55H to the Watch-Dog Timer Control register (WDTCTL).
2. Write AAH to the Watch-Dog Timer Control register (WDTCTL).
3. Write the Watch-Dog Timer Reload Upper Byte register (WDTU).
4. Write the Watch-Dog Timer Reload High Byte register (WDTH).
5. Write the Watch-Dog Timer Reload Low Byte register (WDTL).



All three Watch-Dog Timer Reload registers must be written in the order just listed. There must be no other register writes between each of these operations. If a register write occurs, the lock state machine resets and no further writes can occur unless the sequence is restarted. The value in the Watch-Dog Timer Reload registers is loaded into the counter when the Watch-Dog Timer is first enabled and every time a WDT instruction is executed.

Watch-Dog Timer Control Register Definitions

Watch-Dog Timer Control Register

The Watch-Dog Timer Control (WDTCTL) register is a write-only control register. Writing the 55H, AAH unlock sequence to the WDTCTL register address unlocks the three Watch-Dog Timer Reload Byte registers (WDTU, WDTL, and WDTL) to allow changes to the time-out period. These write operations to the WDTCTL register address produce no effect on the bits in the WDTCTL register. The locking mechanism prevents spurious writes to the Reload registers.

This register address is shared with the read-only Reset Status Register.

Table 54. Watch-Dog Timer Control Register (WDTCTL)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---------|---|---|---|---|---|---|---|
| FIELD | WDTUNLK | | | | | | | |
| RESET | X | X | X | X | X | X | X | X |
| R/W | W | W | W | W | W | W | W | W |
| ADDR | FF0H | | | | | | | |

WDTUNLK—Watch-Dog Timer Unlock

The user software must write the correct unlocking sequence to this register before it is allowed to modify the contents of the watch-dog timer reload registers.

Watch-Dog Timer Reload Upper, High and Low Byte Registers

The Watch-Dog Timer Reload Upper, High and Low Byte (WDTU, WDTL, WDTL) registers (Tables 55 through 57) form the 24-bit reload value that is loaded into the Watch-Dog Timer when a WDT instruction executes. The 24-bit reload value is {WDTU[7:0], WDTL[7:0], WDTL[7:0]}. Writing to these registers sets the appropriate Reload Value. Reading from these registers returns the current Watch-Dog Timer count value.



Caution: The 24-bit WDT Reload Value must not be set to a value less than 000004H.

Table 55. Watch-Dog Timer Reload Upper Byte Register (WDTU)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|------|------|------|------|------|------|------|------|
| FIELD | WDTU | | | | | | | |
| RESET | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | R/W* |
| ADDR | FF1H | | | | | | | |
| R/W* - Read returns the current WDT count value. Write sets the appropriate Reload Value. | | | | | | | | |

WDTU—WDT Reload Upper Byte
Most significant byte (MSB), Bits[23:16], of the 24-bit WDT reload value.

Table 56. Watch-Dog Timer Reload High Byte Register (WDTH)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|------|------|------|------|------|------|------|------|
| FIELD | WDTH | | | | | | | |
| RESET | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | R/W* |
| ADDR | FF2H | | | | | | | |
| R/W* - Read returns the current WDT count value. Write sets the appropriate Reload Value. | | | | | | | | |

WDTH—WDT Reload High Byte
Middle byte, Bits[15:8], of the 24-bit WDT reload value.

Table 57. Watch-Dog Timer Reload Low Byte Register (WDTL)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|------|------|------|------|------|------|------|------|
| FIELD | WDTL | | | | | | | |
| RESET | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | R/W* |
| ADDR | FF3H | | | | | | | |
| R/W* - Read returns the current WDT count value. Write sets the appropriate Reload Value. | | | | | | | | |

WDTL—WDT Reload Low
Least significant byte (LSB), Bits[7:0], of the 24-bit WDT reload value.



UART

Overview

The Universal Asynchronous Receiver/Transmitter (UART) is a full-duplex communication channel capable of handling asynchronous data transfers. The UART uses a single 8-bit data mode with selectable parity. Features of the UART include:

- 8-bit asynchronous data transfer
- Selectable even- and odd-parity generation and checking
- Option of one or two STOP bits
- Separate transmit and receive interrupts
- Framing, parity, overrun and break detection
- Separate transmit and receive enables
- 16-bit Baud Rate Generator (BRG)
- Selectable MULTIPROCESSOR (9-bit) mode with three configurable interrupt schemes
- Baud Rate Generator timer mode
- Driver Enable output for external bus transceivers

Architecture

The UART consists of three primary functional blocks: transmitter, receiver, and baud rate generator. The UART's transmitter and receiver function independently, but employ the same baud rate and data format. Figure 10 illustrates the UART architecture.

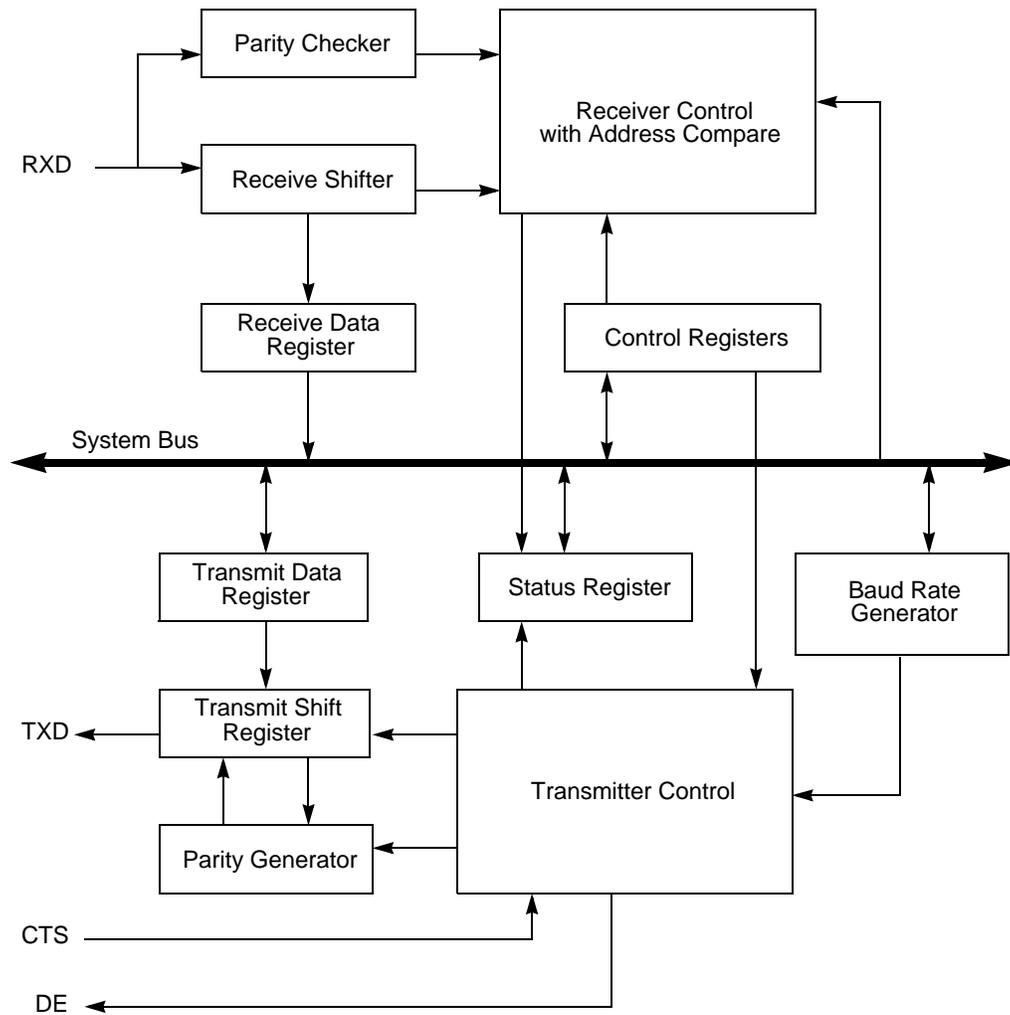


Figure 10. UART Block Diagram

Operation

Data Format

The UART always transmits and receives data in an 8-bit data format, least-significant bit first. An even or odd parity bit can be added to the data stream. Each character begins with an active Low Start bit and ends with either 1 or 2 active High Stop bits. Figures 11 and 12 illustrates the asynchronous data format employed by the UART without parity and with parity, respectively.

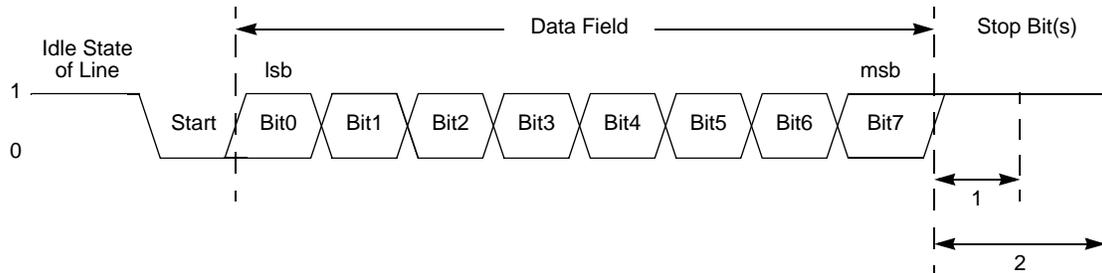


Figure 11. UART Asynchronous Data Format without Parity

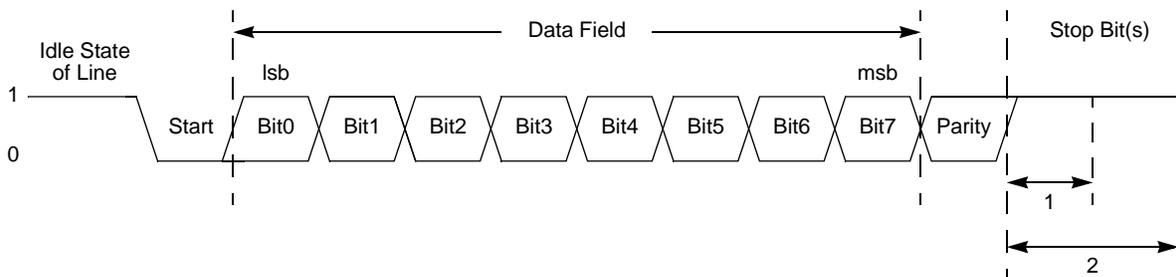


Figure 12. UART Asynchronous Data Format with Parity

Transmitting Data using the Polled Method

Follow these steps to transmit data using the polled method of operation:

1. Write to the UART Baud Rate High and Low Byte registers to set the required baud rate.
2. Enable the UART pin functions by configuring the associated GPIO Port pins for alternate function operation.
3. Write to the UART Control 1 register, if MULTIPROCESSOR mode is appropriate, to enable MULTIPROCESSOR (9-bit) mode functions.
4. Set the Multiprocessor Mode Select (MPEN) bit to enable MULTIPROCESSOR mode.
5. Write to the UART Control 0 register to:
 - Set the transmit enable bit (TEN) to enable the UART for data transmission
 - Set the parity enable bit (PEN), if parity is appropriate and MULTIPROCESSOR mode is not enabled, and select either even or odd parity (PSEL).
 - Set or clear the $\overline{\text{CTS}}$ bit to enable or disable control from the remote receiver using the $\overline{\text{CTS}}$ pin.



6. Check the TDRE bit in the UART Status 0 register to determine if the Transmit Data register is empty (indicated by a 1). If empty, continue to Step 6. If the Transmit Data register is full (indicated by a 0), continue to monitor the TDRE bit until the Transmit Data register becomes available to receive new data.
7. Write the UART Control 1 register to select the outgoing address bit.
8. Set the Multiprocessor Bit Transmitter (MPBT) if sending an address byte, clear it if sending a data byte.
9. Write the data byte to the UART Transmit Data register. The transmitter automatically transfers the data to the Transmit Shift register and transmits the data.
10. Make any changes to the Multiprocessor Bit Transmitter (MPBT) value, if appropriate and MULTIPROCESSOR mode is enabled,.
11. To transmit additional bytes, return to Step 5.

Transmitting Data using the Interrupt-Driven Method

The UART Transmitter interrupt indicates the availability of the Transmit Data register to accept new data for transmission. Follow these steps to configure the UART for interrupt-driven data transmission:

1. Write to the UART Baud Rate High and Low Byte registers to set the appropriate baud rate.
2. Enable the UART pin functions by configuring the associated GPIO Port pins for alternate function operation.
3. Execute a DI instruction to disable interrupts.
4. Write to the Interrupt control registers to enable the UART Transmitter interrupt and set the acceptable priority.
5. Write to the UART Control 1 register to enable MULTIPROCESSOR (9-bit) mode functions, if MULTIPROCESSOR mode is appropriate.
6. Set the MULTIPROCESSOR Mode Select (MPEN) to Enable MULTIPROCESSOR mode.
7. Write to the UART Control 0 register to:
 - Set the transmit enable bit (TEN) to enable the UART for data transmission
 - Enable parity, if appropriate and if MULTIPROCESSOR mode is not enabled, and select either even or odd parity.
 - Set or clear CTSE to enable or disable control from the remote receiver using the $\overline{\text{CTS}}$ pin.
8. Execute an EI instruction to enable interrupts.



The UART is now configured for interrupt-driven data transmission. Because the UART Transmit Data register is empty, an interrupt is generated immediately. When the UART Transmit interrupt is detected, the associated interrupt service routine (ISR) performs the following:

1. Write the UART Control 1 register to select the multiprocessor bit for the byte to be transmitted:
Set the Multiprocessor Bit Transmitter (MPBT) if sending an address byte, clear it if sending a data byte.
2. Write the data byte to the UART Transmit Data register. The transmitter automatically transfers the data to the Transmit Shift register and transmits the data.
3. Clear the UART Transmit interrupt bit in the applicable Interrupt Request register.
4. Execute the IRET instruction to return from the interrupt-service routine and wait for the Transmit Data register to again become empty.

Receiving Data using the Polled Method

Follow these steps to configure the UART for polled data reception:

5. Write to the UART Baud Rate High and Low Byte registers to set an acceptable baud rate for the incoming data stream.
6. Enable the UART pin functions by configuring the associated GPIO Port pins for alternate function operation.
7. Write to the UART Control 1 register to enable MULTIPROCESSOR mode functions, if appropriate.
8. Write to the UART Control 0 register to:
 - Set the receive enable bit (REN) to enable the UART for data reception
 - Enable parity, if appropriate and if Multiprocessor mode is not enabled, and select either even or odd parity.
9. Check the RDA bit in the UART Status 0 register to determine if the Receive Data register contains a valid data byte (indicated by a 1). If RDA is set to 1 to indicate available data, continue to Step 5. If the Receive Data register is empty (indicated by a 0), continue to monitor the RDA bit awaiting reception of the valid data.
10. Read data from the UART Receive Data register. If operating in MULTIPROCESSOR (9-bit) mode, further actions may be required depending on the MULTIPROCESSOR mode bits MPMD[1:0].
11. Return to Step 4 to receive additional data.



Receiving Data using the Interrupt-Driven Method

The UART Receiver interrupt indicates the availability of new data (as well as error conditions). Follow these steps to configure the UART receiver for interrupt-driven operation:

1. Write to the UART Baud Rate High and Low Byte registers to set the acceptable baud rate.
2. Enable the UART pin functions by configuring the associated GPIO Port pins for alternate function operation.
3. Execute a DI instruction to disable interrupts.
4. Write to the Interrupt control registers to enable the UART Receiver interrupt and set the acceptable priority.
5. Clear the UART Receiver interrupt in the applicable Interrupt Request register.
6. Write to the UART Control 1 Register to enable Multiprocessor (9-bit) mode functions, if appropriate.
 - Set the Multiprocessor Mode Select (MPEN) to Enable MULTIPROCESSOR mode.
 - Set the Multiprocessor Mode Bits, MPMD[1:0], to select the acceptable address matching scheme.
 - Configure the UART to interrupt on received data and errors or errors only (interrupt on errors only is unlikely to be useful for Z8 Encore!® devices without a DMA block)
7. Write the device address to the Address Compare Register (automatic MULTIPROCESSOR modes only).
8. Write to the UART Control 0 register to:
 - Set the receive enable bit (REN) to enable the UART for data reception
 - Enable parity, if appropriate and if multiprocessor mode is not enabled, and select either even or odd parity.
9. Execute an EI instruction to enable interrupts.

The UART is now configured for interrupt-driven data reception. When the UART Receiver interrupt is detected, the associated interrupt service routine (ISR) performs the following:

1. Checks the UART Status 0 register to determine the source of the interrupt - error, break, or received data.
2. Reads the data from the UART Receive Data register if the interrupt was because of data available. If operating in MULTIPROCESSOR (9-bit) mode, further actions may be required depending on the MULTIPROCESSOR mode bits MPMD[1:0].
3. Clears the UART Receiver interrupt in the applicable Interrupt Request register.



4. Executes the IRET instruction to return from the interrupt-service routine and await more data.

Clear To Send (CTS) Operation

The CTS pin, if enabled by the CTSE bit of the UART Control 0 register, performs flow control on the outgoing transmit datastream. The Clear To Send (CTS) input pin is sampled one system clock before beginning any new character transmission. To delay transmission of the next data character, an external receiver must deassert CTS at least one system clock cycle before a new data transmission begins. For multiple character transmissions, this action is typically performed during Stop Bit transmission. If CTS deasserts in the middle of a character transmission, the current character is sent completely.

MULTIPROCESSOR (9-bit) Mode

The UART has a MULTIPROCESSOR (9-bit) mode that uses an extra (9th) bit for selective communication when a number of processors share a common UART bus. In MULTIPROCESSOR mode (also referred to as 9-Bit mode), the multiprocessor bit (MP) is transmitted immediately following the 8-bits of data and immediately preceding the Stop bit(s) as illustrated in Figure 13. The character format is:

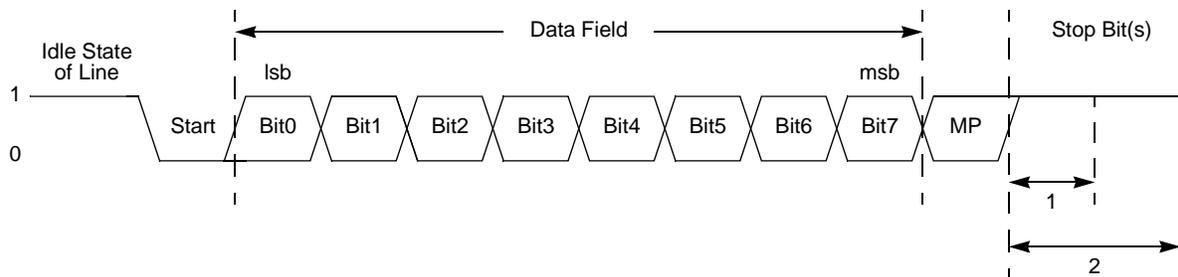


Figure 13. UART Asynchronous MULTIPROCESSOR Mode Data Format

In MULTIPROCESSOR (9-bit) mode, the Parity bit location (9th bit) becomes the Multiprocessor control bit. The UART Control 1 and Status 1 registers provide MULTIPROCESSOR (9-bit) mode control and status information. If an automatic address matching scheme is enabled, the UART Address Compare register holds the network address of the device.

MULTIPROCESSOR (9-bit) Mode Receive Interrupts

When MULTIPROCESSOR mode is enabled, the UART only processes frames addressed to it. The determination of whether a frame of data is addressed to the UART can be made in hardware, software or some combination of the two, depending on the multiprocessor configuration bits. In general, the address compare feature reduces the load on the CPU, because it does not require access to the UART when it receives data directed to other

devices on the multi-node network. The following three MULTIPROCESSOR modes are available in hardware:

- Interrupt on all address bytes
- Interrupt on matched address bytes and correctly framed data bytes
- Interrupt only on correctly framed data bytes

These modes are selected with MPMD [1:0] in the UART Control 1 Register. For all multiprocessor modes, bit MPEN of the UART Control 1 Register must be set to 1.

The first scheme is enabled by writing 01b to MPMD[1:0]. In this mode, all incoming address bytes cause an interrupt, while data bytes never cause an interrupt. The interrupt service routine must manually check the address byte that caused triggered the interrupt. If it matches the UART address, the software clears MPMD[0]. Each new incoming byte interrupts the CPU. The software is responsible for determining the end of the frame. It checks for the end-of-frame by reading the MPRX bit of the UART Status 1 Register for each incoming byte. If MPRX=1, a new frame has begun. If the address of this new frame is different from the UART's address, MPMD[0] must be set to 1 causing the UART interrupts to go inactive until the next address byte. If the new frame's address matches the UART's, the data in the new frame is processed as well.

The second scheme requires the following: set MPMD[1:0] to 10B and write the UART's address into the UART Address Compare Register. This mode introduces additional hardware control, interrupting only on frames that match the UART's address. When an incoming address byte does not match the UART's address, it is ignored. All successive data bytes in this frame are also ignored. When a matching address byte occurs, an interrupt is issued and further interrupts now occur on each successive data byte. When the first data byte in the frame is read, the NEWFRM bit of the UART Status 1 Register is asserted. All successive data bytes have NEWFRM=0. When the next address byte occurs, the hardware compares it to the UART's address. If there is a match, the interrupts continues and the NEWFRM bit is set for the first byte of the new frame. If there is no match, the UART ignores all incoming bytes until the next address match.

The third scheme is enabled by setting MPMD[1:0] to 11b and by writing the UART's address into the UART Address Compare Register. This mode is identical to the second scheme, except that there are no interrupts on address bytes. The first data byte of each frame remains accompanied by a NEWFRM assertion.

External Driver Enable

The UART provides a Driver Enable (DE) signal for off-chip bus transceivers. This feature reduces the software overhead associated with using a GPIO pin to control the transceiver when communicating on a multi-transceiver bus, such as RS-485.

Driver Enable is an active High signal that envelopes the entire transmitted data frame including parity and Stop bits as illustrated in Figure 14. The Driver Enable signal asserts when a byte is written to the UART Transmit Data register. The Driver Enable signal

asserts at least one UART bit period and no greater than two UART bit periods before the Start bit is transmitted. This allows a setup time to enable the transceiver. The Driver Enable signal deasserts one system clock period after the final Stop bit is transmitted. This one system clock delay allows both time for data to clear the transceiver before disabling it, as well as the ability to determine if another character follows the current character. In the event of back to back characters (new data must be written to the Transmit Data Register before the previous character is completely transmitted) the DE signal is not deasserted between characters. The Depol bit in the UART Control Register 1 sets the polarity of the Driver Enable signal.

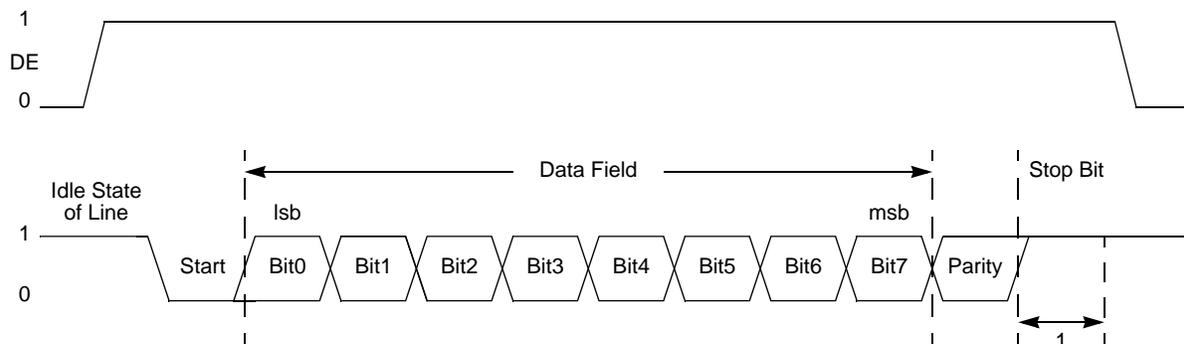


Figure 14. UART Driver Enable Signal Timing (shown with 1 Stop Bit and Parity)

The Driver Enable to Start bit setup time is calculated as follows:

$$\left(\frac{1}{\text{Baud Rate (Hz)}} \right) \leq \text{DE to Start Bit Setup Time (s)} \leq \left(\frac{2}{\text{Baud Rate (Hz)}} \right)$$

UART Interrupts

The UART features separate interrupts for the transmitter and the receiver. In addition, when the UART primary functionality is disabled, the Baud Rate Generator can also function as a basic timer with interrupt capability.

Transmitter Interrupts

The transmitter generates a single interrupt when the Transmit Data Register Empty bit (TDRE) is set to 1. This indicates that the transmitter is ready to accept new data for transmission. The TDRE interrupt occurs after the Transmit shift register has shifted the first bit of data out. The Transmit Data register can now be written with the next character to send. This action provides 7 bit periods of latency to load the Transmit Data register before the Transmit shift register completes shifting the current character. Writing to the UART Transmit Data register clears the TDRE bit to 0.

Receiver Interrupts

The receiver generates an interrupt when any of the following occurs:

- A data byte is received and is available in the UART Receive Data register. This interrupt can be disabled independently of the other receiver interrupt sources. The received data interrupt occurs after the receive character has been received and placed in the Receive Data register. To avoid an overrun error, software must respond to this received data available condition before the next character is completely received.

► **Note:** In MULTIPROCESSOR mode ($MPEN = 1$), the receive data interrupts are dependent on the multiprocessor configuration and the most recent address byte.

- A break is received
- An overrun is detected
- A data framing error is detected

UART Overrun Errors

When an overrun error condition occurs the UART prevents overwriting of the valid data currently in the Receive Data register. The Break Detect and Overrun status bits are not displayed until after the valid data has been read.

After the valid data has been read, the UART Status 0 register is updated to indicate the overrun condition (and Break Detect, if applicable). The RDA bit is set to 1 to indicate that the Receive Data register contains a data byte. However, because the overrun error occurred, this byte may not contain valid data and must be ignored. The BRKD bit indicates if the overrun was caused by a break condition on the line. After reading the status byte indicating an overrun error, the Receive Data register must be read again to clear the error bits in the UART Status 0 register. Updates to the Receive Data register occur only when the next data word is received.

UART Data and Error Handling Procedure

Figure 15 illustrates the recommended procedure for use in UART receiver interrupt service routines.

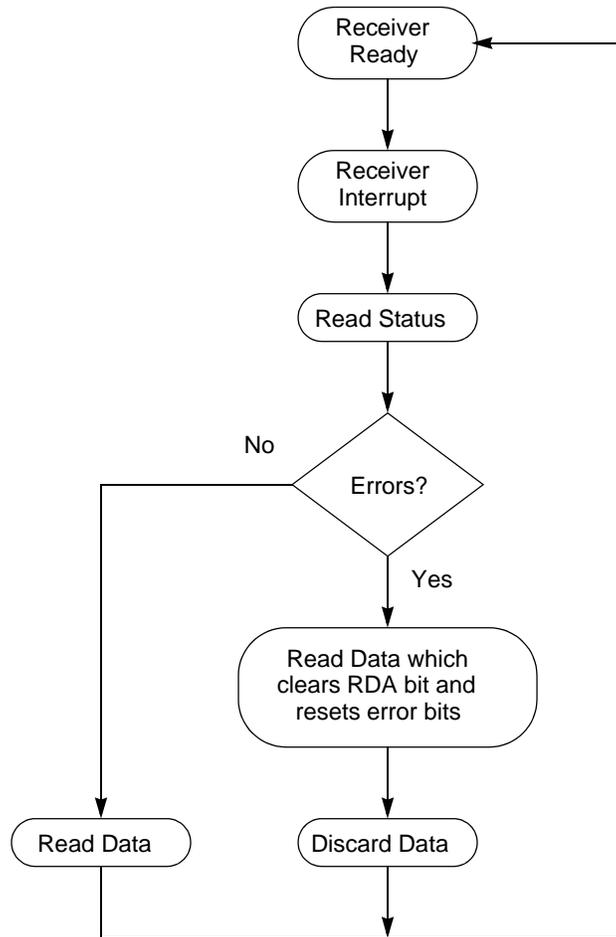


Figure 15. UART Receiver Interrupt Service Routine Flow

Baud Rate Generator Interrupts

If the Baud Rate Generator (BRG) interrupt enable is set, the UART Receiver interrupt asserts when the UART Baud Rate Generator reloads. This condition allows the Baud Rate Generator to function as an additional counter if the UART functionality is not employed.

UART Baud Rate Generator

The UART Baud Rate Generator creates a lower frequency baud rate clock for data transmission. The input to the Baud Rate Generator is the system clock. The UART Baud Rate High and Low Byte registers combine to create a 16-bit baud rate divisor value



(BRG[15:0]) that sets the data transmission rate (baud rate) of the UART. The UART data rate is calculated using the following equation:

$$\text{UART Data Rate (bits/s)} = \frac{\text{System Clock Frequency (Hz)}}{16 \times \text{UART Baud Rate Divisor Value}}$$

When the UART is disabled, the Baud Rate Generator functions as a basic 16-bit timer with interrupt on time-out. To configure the Baud Rate Generator as a timer with interrupt on time-out, complete the following procedure:

1. Disable the UART by clearing the REN and TEN bits in the UART Control 0 register to 0.
2. Load the acceptable 16-bit count value into the UART Baud Rate High and Low Byte registers.
3. Enable the Baud Rate Generator timer function and associated interrupt by setting the BIRQ bit in the UART Control 1 register to 1.

UART Control Register Definitions

The UART control registers support the UART and the associated Infrared Encoder/Decoders. For more information about the infrared operation, refer to the **Infrared Encoder/Decoder** chapter on page 103.

UART Transmit Data Register

Data bytes written to the UART Transmit Data register (Table 58) are shifted out on the TXD_x pin. The Write-only UART Transmit Data register shares a Register File address with the read-only UART Receive Data register.

Table 58. UART Transmit Data Register (U0TXD)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------|---|---|---|---|---|---|---|
| FIELD | TXD | | | | | | | |
| RESET | X | X | X | X | X | X | X | X |
| R/W | W | W | W | W | W | W | W | W |
| ADDR | F40H | | | | | | | |

TXD—Transmit Data

UART transmitter data byte to be shifted out through the TXD_x pin.



UART Receive Data Register

Data bytes received through the RXD_x pin are stored in the UART Receive Data register (Table 59). The read-only UART Receive Data register shares a Register File address with the Write-only UART Transmit Data register.

Table 59. UART Receive Data Register (U0RXD)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------|---|---|---|---|---|---|---|
| FIELD | RXD | | | | | | | |
| RESET | X | X | X | X | X | X | X | X |
| R/W | R | R | R | R | R | R | R | R |
| ADDR | F40H | | | | | | | |

RXD—Receive Data
UART receiver data byte from the RXD_x pin

UART Status 0 Register

The UART Status 0 and Status 1 registers (Table 60 and 54) identify the current UART operating configuration and status.

Table 60. UART Status 0 Register (U0STAT0)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------|----|----|----|------|------|-----|-----|
| FIELD | RDA | PE | OE | FE | BRKD | TDRE | TXE | CTS |
| RESET | 0 | 0 | 0 | 0 | 0 | 1 | 1 | X |
| R/W | R | R | R | R | R | R | R | R |
| ADDR | F41H | | | | | | | |

RDA—Receive Data Available

This bit indicates that the UART Receive Data register has received data. Reading the UART Receive Data register clears this bit.

0 = The UART Receive Data register is empty.

1 = There is a byte in the UART Receive Data register.

PE—Parity Error

This bit indicates that a parity error has occurred. Reading the UART Receive Data register clears this bit.

0 = No parity error has occurred.

1 = A parity error has occurred.

OE—Overrun Error

This bit indicates that an overrun error has occurred. An overrun occurs when new data is



received and the UART Receive Data register has not been read. If the RDA bit is reset to 0, reading the UART Receive Data register clears this bit.

0 = No overrun error occurred.

1 = An overrun error occurred.

FE—Framing Error

This bit indicates that a framing error (no Stop bit following data reception) was detected.

Reading the UART Receive Data register clears this bit.

0 = No framing error occurred.

1 = A framing error occurred.

BRKD—Break Detect

This bit indicates that a break occurred. If the data bits, parity/multiprocessor bit, and Stop bit(s) are all 0s this bit is set to 1. Reading the UART Receive Data register clears this bit.

0 = No break occurred.

1 = A break occurred.

TDRE—Transmitter Data Register Empty

This bit indicates that the UART Transmit Data register is empty and ready for additional data. Writing to the UART Transmit Data register resets this bit.

0 = Do not write to the UART Transmit Data register.

1 = The UART Transmit Data register is ready to receive an additional byte to be transmitted.

TXE—Transmitter Empty

This bit indicates that the transmit shift register is empty and character transmission is finished.

0 = Data is currently transmitting.

1 = Transmission is complete.

CTS— $\overline{\text{CTS}}$ signal

When this bit is read it returns the level of the $\overline{\text{CTS}}$ signal. This signal is active Low.

UART Status 1 Register

This register contains multiprocessor control and status bits.

Table 61. UART Status 1 Register (U0STAT1)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----------|---|---|---|-----|-----|--------|------|
| FIELD | Reserved | | | | | | NEWFRM | MPRX |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R | R | R | R | R/W | R/W | R | R |
| ADDR | F44H | | | | | | | |



Reserved—Must be 0.

NEWFRM—Status bit denoting the start of a new frame. Reading the UART Receive Data register resets this bit to 0.

0 = The current byte is not the first data byte of a new frame.

1 = The current byte is the first data byte of a new frame.

MPRX—Multiprocessor Receive

Returns the value of the most recent multiprocessor bit received. Reading from the UART Receive Data register resets this bit to 0.

UART Control 0 and Control 1 Registers

The UART Control 0 and Control 1 registers (Tables 62 and 63) configure the properties of the UART's transmit and receive operations. The UART Control registers must not be written while the UART is enabled.

Table 62. UART Control 0 Register (U0CTL0)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------------|----------|----------|----------|----------|----------|----------|----------|----------|
| FIELD | TEN | REN | CTSE | PEN | PSEL | SBRK | STOP | LBEN |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W |
| ADDR | F42H | | | | | | | |

TEN—Transmit Enable

This bit enables or disables the transmitter. The enable is also controlled by the $\overline{\text{CTS}}$ signal and the CTSE bit. If the $\overline{\text{CTS}}$ signal is low and the CTSE bit is 1, the transmitter is enabled.

0 = Transmitter disabled.

1 = Transmitter enabled.

REN—Receive Enable

This bit enables or disables the receiver.

0 = Receiver disabled.

1 = Receiver enabled.

CTSE—CTS Enable

0 = The $\overline{\text{CTS}}$ signal has no effect on the transmitter.

1 = The UART recognizes the $\overline{\text{CTS}}$ signal as an enable control from the transmitter.

PEN—Parity Enable

This bit enables or disables parity. Even or odd is determined by the PSEL bit.

0 = Parity is disabled.

1 = The transmitter sends data with an additional parity bit and the receiver receives an additional parity bit.



PSEL—Parity Select

0 = Even parity is transmitted and expected on all received data.

1 = Odd parity is transmitted and expected on all received data.

SBRK—Send Break

This bit pauses or breaks data transmission. Sending a break interrupts any transmission in progress, so ensure that the transmitter has finished sending data before setting this bit.

0 = No break is sent.

1 = Forces a break condition by setting the output of the transmitter to zero.

STOP—Stop Bit Select

0 = The transmitter sends one stop bit.

1 = The transmitter sends two stop bits.

LBEN—Loop Back Enable

0 = Normal operation.

1 = All transmitted data is looped back to the receiver.

Table 63. UART Control 1 Register (U0CTL1)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------------|----------|----------|----------|----------|----------|----------|----------|----------|
| FIELD | MPMD[1] | MPEN | MPMD[0] | MPBT | DEPOL | BRGCTL | RDAIRQ | IREN |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W |
| ADDR | F43H | | | | | | | |

MPMD[1:0]—MULTIPROCESSOR Mode

If MULTIPROCESSOR (9-bit) mode is enabled,

00 = The UART generates an interrupt request on all received bytes (data and address).

01 = The UART generates an interrupt request only on received address bytes.

10 = The UART generates an interrupt request when a received address byte matches the value stored in the Address Compare Register and on all successive data bytes until an address mismatch occurs.

11 = The UART generates an interrupt request on all received data bytes for which the most recent address byte matched the value in the Address Compare Register.

MPEN—MULTIPROCESSOR (9-bit) Enable

This bit is used to enable MULTIPROCESSOR (9-bit) mode.

0 = Disable MULTIPROCESSOR (9-bit) mode.

1 = Enable MULTIPROCESSOR (9-bit) mode.

MPBT—Multiprocessor Bit Transmit

This bit is applicable only when MULTIPROCESSOR (9-bit) mode is enabled. The 9th bit is used by the receiving device to determine if the data byte contains address or data information.



0 = Send a 0 in the multiprocessor bit location of the data stream (data byte).
1 = Send a 1 in the multiprocessor bit location of the data stream (address byte).

DEPOL—Driver Enable Polarity

0 = DE signal is Active High.
1 = DE signal is Active Low.

BRGCTL—Baud Rate Control

This bit causes an alternate UART behavior depending on the value of the REN bit in the UART Control 0 Register.

When the UART receiver is **not** enabled (REN=0), this bit determines whether the Baud Rate Generator issues interrupts.

0 = Reads from the Baud Rate High and Low Byte registers return the BRG Reload Value
1 = The Baud Rate Generator generates a receive interrupt when it counts down to 0. Reads from the Baud Rate High and Low Byte registers return the current BRG count value.

When the UART receiver is enabled (REN=1), this bit allows reads from the Baud Rate Registers to return the BRG count value instead of the Reload Value.

0 = Reads from the Baud Rate High and Low Byte registers return the BRG Reload Value.
1 = Reads from the Baud Rate High and Low Byte registers return the current BRG count value. Unlike the Timers, there is no mechanism to latch the Low Byte when the High Byte is read.

$\overline{\text{RDAIRQ}}$ —Receive Data Interrupt Enable

0 = Received data and receiver errors generates an interrupt request to the Interrupt Controller.
1 = Received data does not generate an interrupt request to the Interrupt Controller. Only receiver errors generate an interrupt request.

IREN—Infrared Encoder/Decoder Enable

0 = Infrared Encoder/Decoder is disabled. UART operates normally.
1 = Infrared Encoder/Decoder is enabled. The UART transmits and receives data through the Infrared Encoder/Decoder.

UART Address Compare Register

The UART Address Compare register stores the multi-node network address of the UART. When the MPMD[1] bit of UART Control Register 0 is set, all incoming address bytes are compared to the value stored in the Address Compare register. Receive interrupts and RDA assertions only occur in the event of a match.



Table 64. UART Address Compare Register (U0ADDR)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|-----------|-----|-----|-----|-----|-----|-----|-----|
| FIELD | COMP_ADDR | | | | | | | |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | F45H | | | | | | | |

COMP_ADDR—Compare Address
This 8-bit value is compared to incoming address bytes.

UART Baud Rate High and Low Byte Registers

The UART Baud Rate High and Low Byte registers (Tables 65 and 66) combine to create a 16-bit baud rate divisor value (BRG[15:0]) that sets the data transmission rate (baud rate) of the UART.

Table 65. UART Baud Rate High Byte Register (U0BRH)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------|-----|-----|-----|-----|-----|-----|-----|
| FIELD | BRH | | | | | | | |
| RESET | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | F46H | | | | | | | |

Table 66. UART Baud Rate Low Byte Register (U0BRL)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------|-----|-----|-----|-----|-----|-----|-----|
| FIELD | BRL | | | | | | | |
| RESET | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | F47H | | | | | | | |

The UART data rate is calculated using the following equation:

$$\text{UART Baud Rate (bits/s)} = \frac{\text{System Clock Frequency (Hz)}}{16 \times \text{UART Baud Rate Divisor Value}}$$

For a given UART data rate, calculate the integer baud rate divisor value using the following equation:



$$\text{UART Baud Rate Divisor Value (BRG)} = \text{Round}\left(\frac{\text{System Clock Frequency (Hz)}}{16 \times \text{UART Data Rate (bits/s)}}\right)$$

The baud rate error relative to the acceptable baud rate is calculated using the following equation:

$$\text{UART Baud Rate Error (\%)} = 100 \times \left(\frac{\text{Actual Data Rate} - \text{Desired Data Rate}}{\text{Desired Data Rate}} \right)$$

For reliable communication, the UART baud rate error must never exceed 5 percent. Table 67 provides information about data rate errors for 5.5296MHz system clock.

Table 67. UART Baud Rates

| 5.5296 MHz System Clock | | | |
|--------------------------------|------------------------------|--------------------------|------------------|
| Acceptable Rate (KHz) | BRG Divisor (Decimal) | Actual Rate (KHz) | Error (%) |
| 1250.0 | N/A | N/A | N/A |
| 625.0 | N/A | N/A | N/A |
| 250.0 | 1 | 345.6 | 38.24 |
| 115.2 | 3 | 115.2 | 0.00 |
| 57.6 | 6 | 57.6 | 0.00 |
| 38.4 | 9 | 38.4 | 0.00 |
| 19.2 | 18 | 19.2 | 0.00 |
| 9.60 | 36 | 9.60 | 0.00 |
| 4.80 | 72 | 4.80 | 0.00 |
| 2.40 | 144 | 2.40 | 0.00 |
| 1.20 | 288 | 1.20 | 0.00 |
| 0.60 | 576 | 0.60 | 0.00 |
| 0.30 | 1152 | 0.30 | 0.00 |

Infrared Encoder/Decoder

Overview

The 4K Series products contain a fully-functional, high-performance UART to Infrared Encoder/Decoder (Endec). The Infrared Endec is integrated with an on-chip UART to allow easy communication between the Z8 Encore!® and IrDA Physical Layer Specification, Version 1.3-compliant infrared transceivers. Infrared communication provides secure, reliable, low-cost, point-to-point communication between PCs, PDAs, cell phones, printers and other infrared enabled devices.

Architecture

Figure 16 illustrates the architecture of the Infrared Endec.

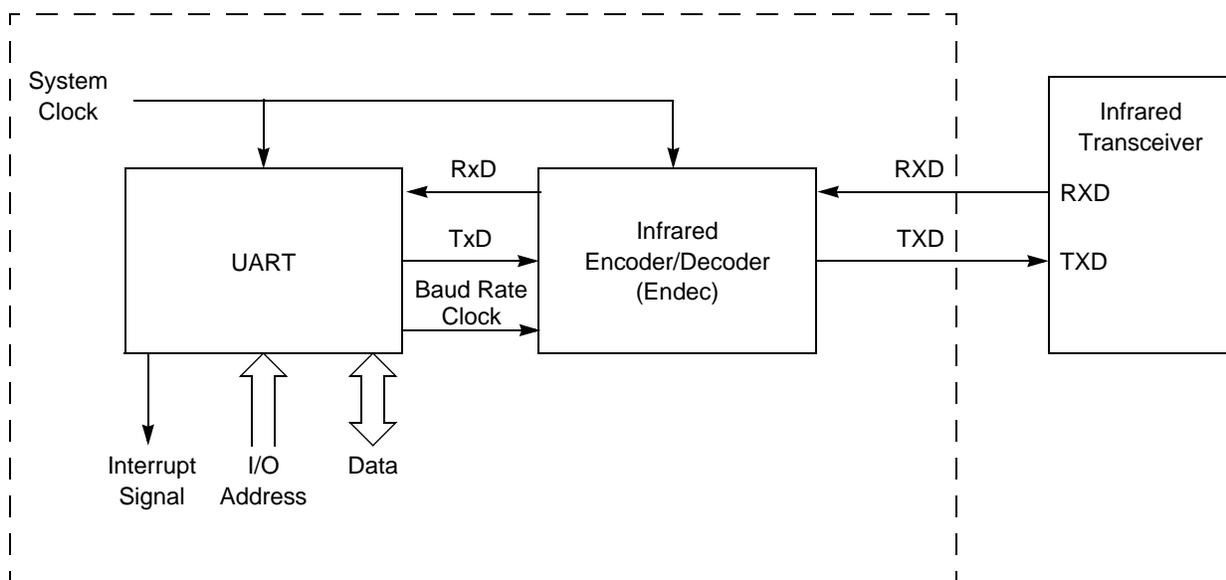


Figure 16. Infrared Data Communication System Block Diagram

Operation

When the Infrared Endec is enabled, the transmit data from the associated on-chip UART is encoded as digital signals in accordance with the IrDA standard and output to the infrared transceiver through the TXD pin. Likewise, data received from the infrared transceiver



is passed to the Infrared Endec through the RXD pin, decoded by the Infrared Endec, and passed to the UART. Communication is half-duplex, which means simultaneous data transmission and reception is not allowed.

The baud rate is set by the UART's Baud Rate Generator and supports IrDA standard baud rates from 9600 baud to 115.2 Kbaud. Higher baud rates are possible, but do not meet IrDA specifications. The UART must be enabled to use the Infrared Endec. The Infrared Endec data rate is calculated using the following equation:

$$\text{Infrared Data Rate (bits/s)} = \frac{\text{System Clock Frequency (Hz)}}{16 \times \text{UART Baud Rate Divisor Value}}$$

Transmitting IrDA Data

The data to be transmitted using the infrared transceiver is first sent to the UART. The UART's transmit signal (TXD) and baud rate clock are used by the IrDA to generate the modulation signal (IR_TXD) that drives the infrared transceiver. Each UART/Infrared data bit is 16 clocks wide. If the data to be transmitted is 1, the IR_TXD signal remains low for the full 16 clock period. If the data to be transmitted is 0, the transmitter first outputs a 7 clock low period, followed by a 3 clock high pulse. Finally, a 6 clock low pulse is output to complete the full 16 clock data period. Figure 17 illustrates IrDA data transmission. When the Infrared Endec is enabled, the UART's TXD signal is internal to the 4K Series products while the IR_TXD signal is output through the TXD pin.

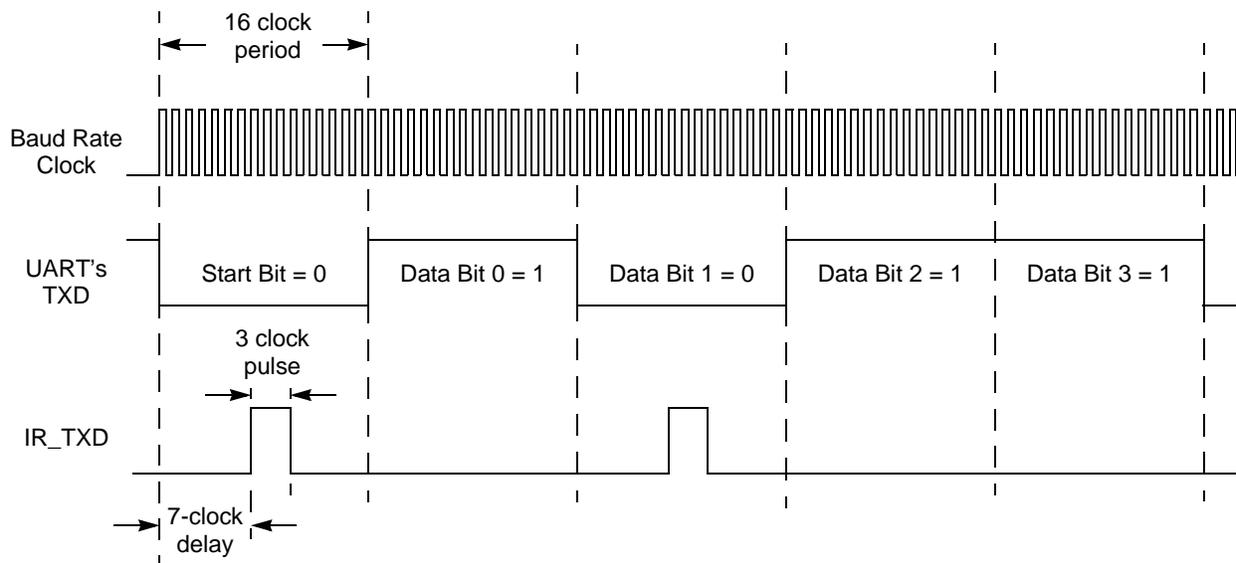


Figure 17. Infrared Data Transmission

Receiving IrDA Data

Data received from the infrared transceiver using the IR_RXD signal through the RXD pin is decoded by the Infrared Endec and passed to the UART. The UART's baud rate clock is used by the Infrared Endec to generate the demodulated signal (RXD) that drives the UART. Each UART/Infrared data bit is 16-clocks wide. Figure 18 illustrates data reception. When the Infrared Endec is enabled, the UART's RXD signal is internal to the 4K Series products while the IR_RXD signal is received through the RXD pin.

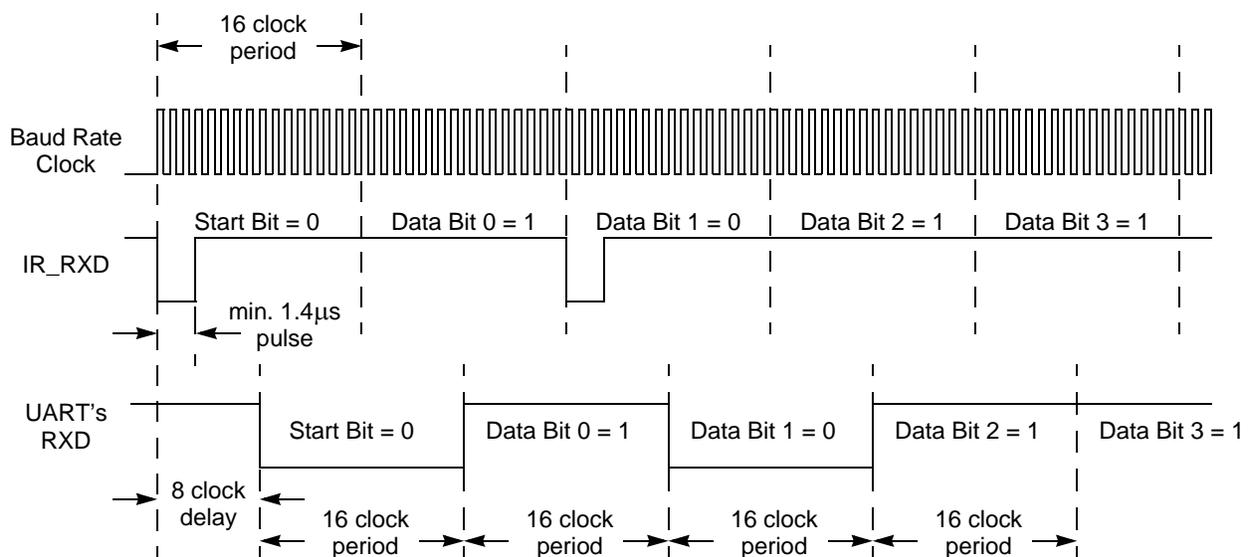


Figure 18. IrDA Data Reception

Infrared Data Reception



Caution: The system clock frequency must be at least 1.0MHz to ensure proper reception of the 1.4µs minimum width pulses allowed by the IrDA standard.

Endec Receiver Synchronization

The IrDA receiver uses a local baud rate clock counter (0 to 15 clock periods) to generate an input stream for the UART and to create a sampling window for detection of incoming pulses. The generated UART input (UART RXD) is delayed by 8 baud rate clock periods with respect to the incoming IrDA data stream. When a falling edge in the input data stream is detected, the Endec counter is reset. When the count reaches a value of 8, the UART RXD value is updated to reflect the value of the decoded data. When the count reaches 12 baud clock periods, the sampling window for the next incoming pulse opens. The window remains open until the count again reaches 8 (in other words, 24 baud clock periods since the previous pulse was detected), giving the Endec a sampling window of minus four baud rate clocks to plus eight baud rate clocks around the expected time of an



incoming pulse. If an incoming pulse is detected inside this window this process is repeated. If the incoming data is a logical 1 (no pulse), the Endec returns to the initial state and waits for the next falling edge. As each falling edge is detected, the Endec clock counter is reset, resynchronizing the Endec to the incoming signal, allowing the Endec to tolerate jitter and baud rate errors in the incoming datastream. Resynchronizing the Endec does not alter the operation of the UART, which ultimately receives the data. The UART is only synchronized to the incoming data stream when a Start bit is received.

Infrared Encoder/Decoder Control Register Definitions

All Infrared Endec configuration and status information is set by the UART control registers as defined beginning on page 84.



Caution: To prevent spurious signals during IrDA data transmission, set the IREN bit in the UART Control 1 register to 1 to enable the Infrared Encoder/Decoder **before** enabling the GPIO Port alternate function for the corresponding pin.

Analog-to-Digital Converter

Overview

The Analog-to-Digital Converter (ADC) converts an analog input signal to its digital representation. The features of this sigma-delta ADC include:

- 10-bit resolution
- 8 single-ended analog input sources are multiplexed with general-purpose I/O ports
- Interrupt upon conversion complete
- Internal voltage reference generator with three selectable levels

Architecture

Figure 19 illustrates the major functional blocks of the ADC. An analog multiplexer network selects the ADC input from the available analog pins, ANA0 through ANA7. The ADC requires an input reference voltage for the conversion. The voltage reference for the conversion can be input through the external VREF pin or generated internally by the voltage reference generator.

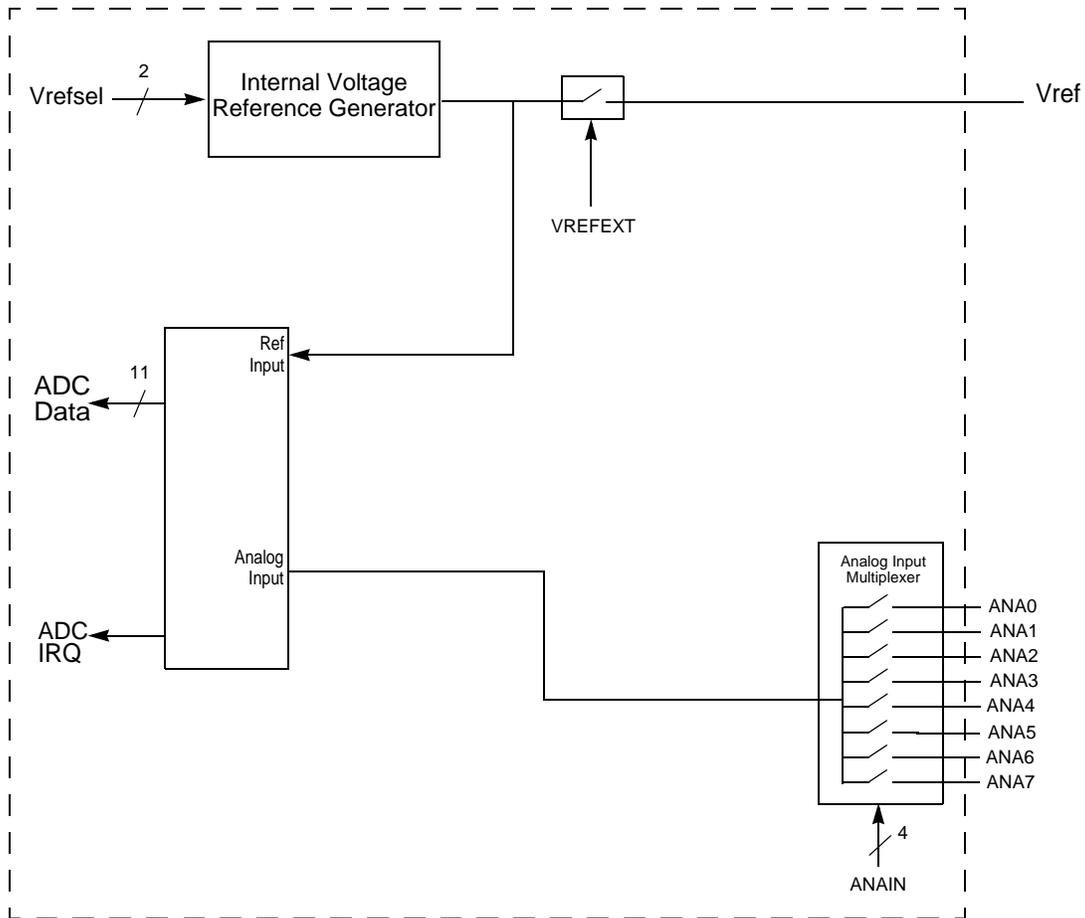


Figure 19. Analog-to-Digital Converter Block Diagram

Operation

Data Format

The output of the ADC is an 11-bit, signed, two's complement digital value. The output generally ranges from 0 to +1023, but offset errors can cause small negative values.

The ADC registers return 13 bits of data, but the two LSBs are intended for compensation use only. When the compensation routine is performed on the 13-bit raw ADC value, two bits of resolution are lost because of a rounding error. As a result, the final value is an 11-bit number.



Automatic Powerdown

If the ADC is idle (no conversions in progress) for 160 consecutive system clock cycles, portions of the ADC are automatically powered down. From this powerdown state, the ADC requires 40 system clock cycles to powerup. The ADC powers up when a conversion is requested by the ADC Control register.

Single-Shot Conversion

When configured for single-shot conversion, the ADC performs a single analog-to-digital conversion on the selected analog input channel. After completion of the conversion, the ADC shuts down. The steps for setting up the ADC and initiating a single-shot conversion are as follows:

1. Enable the acceptable analog inputs by configuring the general-purpose I/O pins for alternate function. This configuration disables the digital input and output drivers.
2. Write the **ADC Control/Status Register 1** to configure the ADC
 - Write the REFSELH bit of the pair {REFSELH, REFSELL} to select the internal voltage reference level or to disable the internal reference. The REFSELL bit is contained in the **ADC Control Register 0**.
3. Write to the **ADC Control Register 0** to configure the ADC and begin the conversion. The bit fields in the ADC Control register can be written simultaneously:
 - Write to the ANAIN[3:0] field to select from the available analog input sources (different input pins available depending on the device)
 - Clear CONT to 0 to select a single-shot conversion.
 - If the internal voltage reference must be output to a pin, set the REFEXT bit to 1. The internal voltage reference must be enabled in this case.
 - Write the REFSELL bit of the pair {REFSELH, REFSELL} to select the internal voltage reference level or to disable the internal reference. The REFSELH bit is contained in the **ADC Control/Status Register 1**.
 - Set CEN to 1 to start the conversion.
4. CEN remains 1 while the conversion is in progress. A single-shot conversion requires 5129 system clock cycles to complete. If a single-shot conversion is requested from an ADC powered-down state, the ADC uses 40 additional clock cycles to power-up before beginning the 5129 cycle conversion.
5. When the conversion is complete, the ADC control logic performs the following operations:
 - 11-bit two's-complement result written to {ADCD_H[7:0], ADCD_L[7:5]}.
 - CEN resets to 0 to indicate the conversion is complete.
 - An interrupt request is sent to the Interrupt Controller denoting conversion complete.

6. If the ADC remains idle for 160 consecutive system clock cycles, it is automatically powered-down.

Continuous Conversion

When configured for continuous conversion, the ADC continuously performs an analog-to-digital conversion on the selected analog input. Each new data value over-writes the previous value stored in the ADC Data registers. An interrupt is generated after each conversion.



Caution: In CONTINUOUS mode, ADC updates are limited by the input signal bandwidth of the ADC and the latency of the ADC and its digital filter. Step changes at the input are not detected at the next output from the ADC. The response of the ADC (in all modes) is limited by the input signal bandwidth and the latency.

Follow these steps for setting up the ADC and initiating continuous conversion:

1. Enable the acceptable analog input by configuring the general-purpose I/O pins for alternate function. This action disables the digital input and output driver.
2. Write the **ADC Control/Status Register 1** to configure the ADC
 - Write the REFSELH bit of the pair {REFSELH, REFSELL} to select the internal voltage reference level or to disable the internal reference. The REFSELL bit is contained in the **ADC Control Register 0**.
3. Write to the **ADC Control Register 0** to configure the ADC for continuous conversion. The bit fields in the ADC Control register may be written simultaneously:
 - Write to the ANAIN[3:0] field to select from the available analog input sources (different input pins available depending on the device)
 - Set CONT to 1 to select continuous conversion.
 - If the internal VREF must be output to a pin, set the REFEXT bit to 1. The internal voltage reference must be enabled in this case.
 - Write the REFSELL bit of the pair {REFSELH, REFSELL} to select the internal voltage reference level or to disable the internal reference. The REFSELH bit is contained in **ADC Control/Status Register 1**.
 - Set CEN to 1 to start the conversions.
4. When the first conversion in continuous operation is complete (after 5129 system clock cycles, plus the 40 cycles for power-up, if necessary), the ADC control logic performs the following operations:
 - CEN resets to 0 to indicate the first conversion is complete. CEN remains 0 for all subsequent conversions in continuous operation.
 - An interrupt request is sent to the Interrupt Controller to indicate the conversion is complete.



5. The ADC writes a new data result every 256 system clock cycles. For each completed conversion, the ADC control logic performs the following operations:
 - Writes the 11-bit two’s complement result to {ADCD_H[7:0], ADCD_L[7:5]}.
 - An interrupt request to the Interrupt Controller denoting conversion complete.
6. To disable continuous conversion, clear the CONT bit in the ADC Control Register to 0.

ADC Control Register Definitions

ADC Control Register 0

The ADC Control register selects the analog input channel and initiates the analog-to-digital conversion.

Table 68. ADC Control Register 0 (ADCCTL0)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------|---------|--------|------|------------|-----|-----|-----|
| FIELD | CEN | REFSELL | REFEXT | CONT | ANAIN[3:0] | | | |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | F70H | | | | | | | |

CEN—Conversion Enable

0 = Conversion is complete. Writing a 0 produces no effect. The ADC automatically clears this bit to 0 when a conversion is complete.

1 = Begin conversion. Writing a 1 to this bit starts a conversion. If a conversion is already in progress, the conversion restarts. This bit remains 1 until the conversion is complete.

REFSELL—Voltage Reference Level Select Low Bit; in conjunction with the High bit (REFSELH) in **ADC Control/Status Register 1**, this determines the level of the internal voltage reference; the following details the effects of {REFSELH, REFSELL}; note that this reference is independent of the Comparator reference

00= Internal Reference Disabled, reference comes from external pin

01= Internal Reference set to 1.0 V

10= Internal Reference set to 2.0 V (default)

11= Internal Reference set to 2.2 V

REFEXT - External Reference Select

0 = External reference buffer is disabled; Vref pin is available for GPIO functions

1 = The internal ADC reference is buffered and connected to the Vref pin

CONT

0 = Single-shot conversion. ADC data is output once at completion of the 5129 system



clock cycles

1 = Continuous conversion. ADC data updated every 256 system clock cycles

ANAIN[3:0]—Analog Input Select

These bits select the analog input for conversion. Not all Port pins in this list are available in all packages for the 4K Series. Refer to the chapter “**Pin Description**” on page 7 for information regarding the Port pins available with each package style. Do not enable unavailable analog inputs. Usage of these bits changes depending on the buffer mode selected in **ADC Control/Status Register 1**.

For the reserved values, all input switches are disabled to avoid leakage or other undesirable operation. ADC samples taken with reserved bit settings are undefined.

Single-Ended:

- 0000 = ANA0
- 0001 = ANA1
- 0010 = ANA2
- 0011 = ANA3
- 0100 = ANA4
- 0101 = ANA5
- 0110 = ANA6
- 0111 = ANA7
- 1000 = Reserved
- 1001 = Reserved
- 1010 = Reserved
- 1011 = Reserved
- 1100 = Reserved.
- 1101 = Reserved
- 1110 = Reserved
- 1111 = Reserved

ADC Control/Status Register 1

The second ADC Control register configures the input buffer stage, enables the threshold interrupts and contains the status of both threshold triggers.

Table 69. ADC Control/Status Register 1 (ADCCTL1)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------------|----------|----------|----------|----------|----------|----------|----------|----------|
| FIELD | REFSELH | Reserved | | | | | | |
| RESET | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W |
| ADDR | F71H | | | | | | | |



REFSELH—Voltage Reference Level Select High Bit; in conjunction with the Low bit (REFSELL) in **ADC Control Register 0**, this determines the level of the internal voltage reference; the following details the effects of {REFSELH, REFSELL}; this reference is independent of the Comparator reference

00= Internal Reference Disabled, reference comes from external pin

01= Internal Reference set to 1.0 V

10= Internal Reference set to 2.0 V (default)

11= Internal Reference set to 2.2 V

Reserved—Must be 0

ADC Data High Byte Register

The ADC Data High Byte register contains the upper eight bits of the ADC output. The output is an 11-bit two's complement value. During a single-shot conversion, this value is invalid. Access to the ADC Data High Byte register is read-only. Reading the ADC Data High Byte register latches data in the ADC Low Bits register.

Table 70. ADC Data High Byte Register (ADCD_H)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|-------|---|---|---|---|---|---|---|
| FIELD | ADCDH | | | | | | | |
| RESET | X | X | X | X | X | X | X | X |
| R/W | R | R | R | R | R | R | R | R |
| ADDR | F72H | | | | | | | |

ADCDH—ADC Data High Byte

This byte contains the upper eight bits of the ADC output. These bits are not valid during a single-shot conversion. During a continuous conversion, the most recent conversion output is held in this register. These bits are undefined after a Reset.

ADC Data Low Bits Register

The ADC Data Low Byte register contains the lower bits of the ADC output as well as an overflow status bit. The output is a 11-bit two's complement value. During a single-shot conversion, this value is invalid. Access to the ADC Data Low Byte register is read-only. Reading the ADC Data High Byte register latches data in the ADC Low Bits register.

Table 71. ADC Data Low Bits Register (ADCD_L)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|-------|---|---|----------|---|---|---|-----|
| FIELD | ADCDL | | | Reserved | | | | OVF |
| RESET | X | X | X | X | X | X | X | X |



Table 71. ADC Data Low Bits Register (ADCD_L)

| | | | | | | | | |
|-------------|------|---|---|---|---|---|---|---|
| R/W | R | R | R | R | R | R | R | R |
| ADDR | F73H | | | | | | | |

ADCDL—ADC Data Low Bits

These bits are the least significant three bits of the 11-bits of the differential ADC output. In single-ended mode, they are the least significant two bits of the 10-bit output. These bits are undefined after a Reset.

Reserved—Must be undefined.

OVF—Overflow Status

0= An overflow did not occur in the digital filter for the current sample.

1= An overflow did occur in the digital filter for the current sample.

Flash Memory

Overview

The products in the PartName features either 4KB (4096) or 2KB (2048 bytes) or 1KB (1024) of non-volatile Flash memory with read/write/erase capability. The Flash Memory can be programmed and erased in-circuit by either user code or through the On-Chip Debugger.

The Flash memory array is arranged in pages with 512 bytes per page. The 512-byte page is the minimum Flash block size that can be erased. Each page is divided into 8 rows of 64 bytes.

For program/data protection, the Flash memory is also divided into sectors. In the Part-Name, these sectors are 512 bytes in size; each sector maps to a page. Page and sector sizes are not equal for other members of the Z8 Encore!® family.

The first 2 bytes of the Flash Program memory are used as Flash Option Bits. Refer to the chapter “Flash Option Bits” on page 126 for more information about their operation.

Table 72 describes the Flash memory configuration for each device in the PartName. Figure 20 illustrates the Flash memory arrangement.

Table 72. 4K Series Flash Memory Configurations

| Part Number | Flash Size KB (Bytes) | Flash Pages | Program Memory Addresses | Flash Sector Size (bytes) |
|-------------|--------------------------|----------------|-----------------------------|------------------------------|
| Z8F04xA | 4 (4096) | 8 | 0000H–0FFFH | 512 |
| Z8F02xA | 2 (2048) | 4 | 0000H–07FFH | 512 |
| Z8F01xA | 1 (1024) | 2 | 0000H–03FFH | 512 |

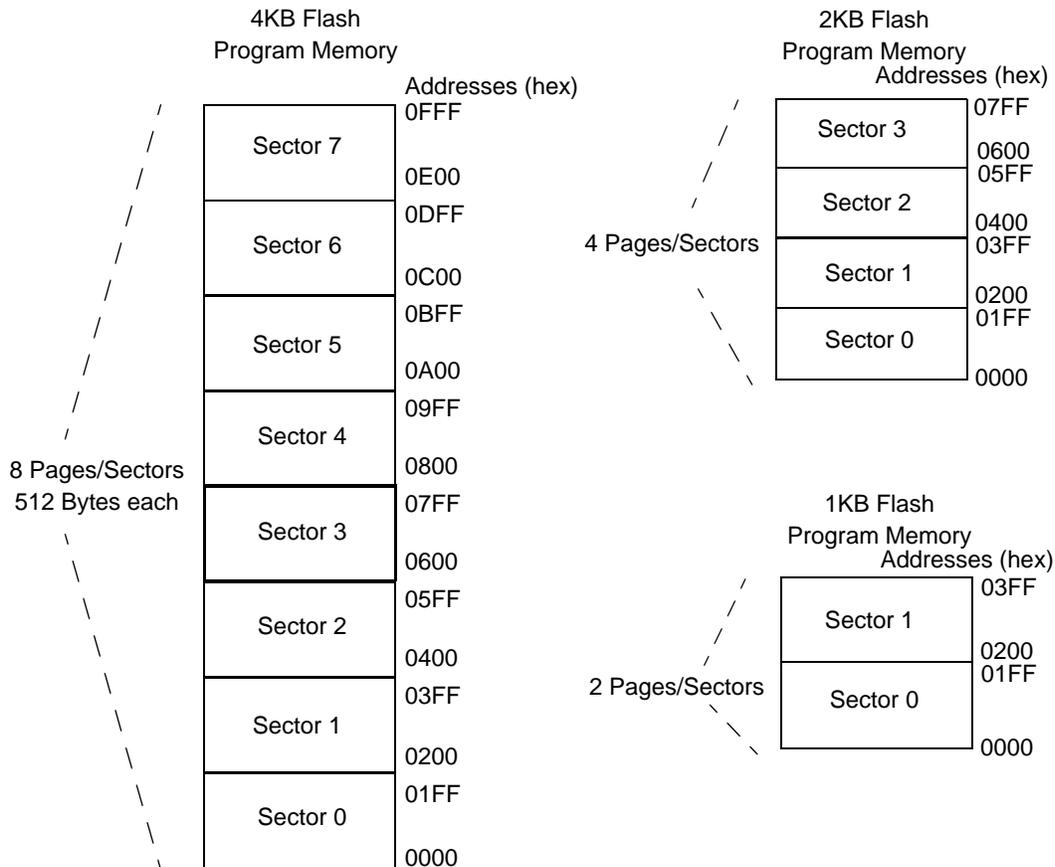


Figure 20. Flash Memory Arrangement

Flash Information Area

The Flash information area is separate from program memory and is mapped to the address range FE00H to FFFFH. Not all of these addresses are user accessible. Factory trim values for the Temperature Sensor and Internal Precision Oscillator are stored here. Factory calibration data for the ADC is also stored here.



Operation

The Flash Controller programs and erases Flash memory. The Flash Controller provides the proper Flash controls and timing for byte programming, Page Erase, and Mass Erase of Flash memory.

The Flash Controller contains several protection mechanisms to prevent accidental programming or erasure. These mechanism operate on the page, sector and full-memory levels.

The Flow Chart in Figure 21 illustrates basic Flash Controller operation. The following subsections provide details about the various operations (Lock, Unlock, Byte Programming, Page Protect, Page Unprotect, Page Select Page Erase, and Mass Erase) listed in Figure 21.

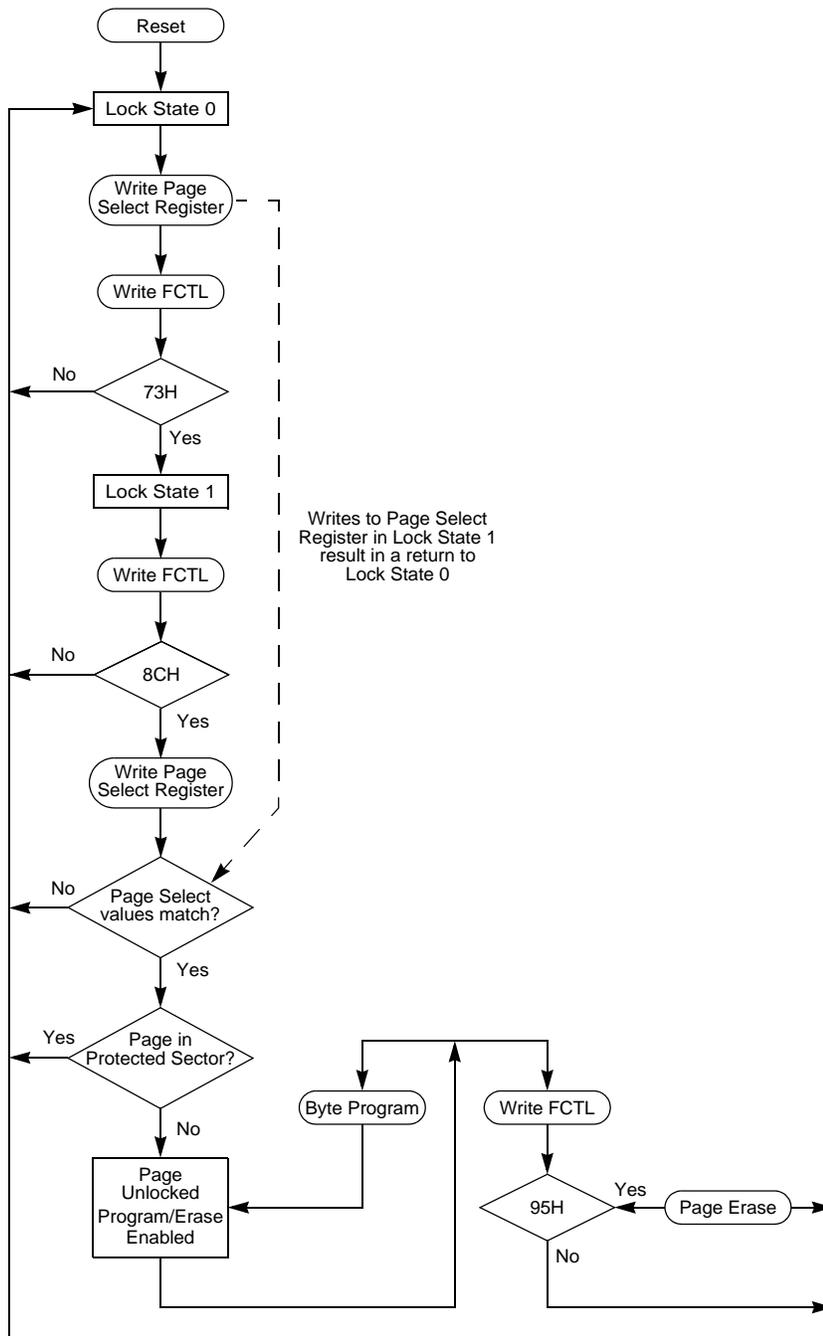


Figure 21. Flash Controller Operation Flow Chart

Flash Operation Timing Using the Flash Frequency Registers

Before performing either a program or erase operation on Flash memory, the user must first configure the Flash Frequency High and Low Byte registers. The Flash Frequency registers allow programming and erasing of the Flash with system clock frequencies ranging from 32KHz (32768Hz) through 5.53MHz.

The Flash Frequency High and Low Byte registers combine to form a 16-bit value, FFREQ, to control timing for Flash program and erase operations. The 16-bit binary Flash Frequency value must contain the system clock frequency (in KHz). This value is calculated using the following equation:

$$\text{FFREQ}[15:0] = \frac{\text{System Clock Frequency (Hz)}}{1000}$$



Caution: Flash programming and erasure are not supported for system clock frequencies below 32KHz (32768Hz) or above 5.53MHz. The Flash Frequency High and Low Byte registers must be loaded with the correct value to ensure operation of the 4K Series devices.

Flash Code Protection Against External Access

The user code contained within the Flash memory can be protected against external access with the On-Chip Debugger. Programming the FRP Flash Option Bit prevents reading of the user code with the On-Chip Debugger. Refer to the chapter “Flash Option Bits” on page 126 and the chapter “On-Chip Debugger” on page 131 for more information.

Flash Code Protection Against Accidental Program and Erasure

The 4K Series provides several levels of protection against accidental program and erasure of the Flash memory contents. This protection is provided by a combination of the Flash Option bits, the register locking mechanism, the page select redundancy and the sector level protection control of the Flash Controller.

Flash Code Protection Using the Flash Option Bits

The FHSWP and FWP Flash Option Bits combine to provide three levels of Flash Program Memory protection as listed in Table 73. Refer to the chapter “Flash Option Bits” on page 126 for more information.



Table 73. Flash Code Protection Using the Flash Option Bits

| FHSWP | FWP | Flash Code Protection Description |
|--------|-----|--|
| 0 | 0 | Programming and erasing disabled for all of Flash Program Memory. In user code programming, Page Erase, and Mass Erase are all disabled. Mass Erase is available through the On-Chip Debugger. |
| 0 or 1 | 1 | Programming, Page Erase, and Mass Erase are enabled for all of Flash Program Memory. |

Flash Code Protection Using the Flash Controller

At Reset, the Flash Controller locks to prevent accidental program or erasure of the Flash memory. To program or erase the Flash memory, first write the Page Select Register with the target page. Unlock the Flash Controller by making two consecutive writes to the Flash Control register with the values 73H and 8CH, sequentially. The Page Select Register must be rewritten with the same page previously stored there. If the two Page Select writes do not match, the controller reverts to a locked state. If the two writes match, the selected page becomes active. See Figure 21 for details.

After unlocking a specific page, the user can enable either Page Program or Erase. Writing the value 95H causes a Page Erase only if the active page resides in a sector that is not protected. Any other value written to the Flash Control register locks the Flash Controller. Mass Erase is not allowed in the user code but only in through the Debug Port.

After unlocking a specific page, the user can also write to any byte on that page. After a byte is written, the page remains unlocked, allowing for subsequent writes to other bytes on the same page. Further writes to the Flash Control Register cause the active page to revert to a locked state.

Sector Based Flash Protection

The final protection mechanism is implemented on a per-sector basis. The Flash memories of Z8 Encore!® devices are divided into maximum number of 8 sectors. A sector is 1/8 of the total size of the Flash memory, unless this value is smaller than the page size, in which case the sector and page sizes are equal. On the 4K Series devices, the sector size is 512 bytes, equal to the page size.

The Sector Protect Register controls the protection state of each Flash sector. This register is shared with the Page Select Register. It is accessed by unlocking the Flash controller and writing the command byte 5EH. The next write to the Page Select Register targets the Sector Protect Register.

The Sector Protect Register is initialized to 0 on reset, putting each sector into an unprotected state. When a bit in the Sector Protect Register is written to 1, the corresponding

sector can no longer be written or erased. After a bit of the Sector Protect Register has been set, it can not be cleared except by powering down the device.

Byte Programming

The Flash Memory is enabled for byte programming after unlocking the Flash Controller and successfully enabling either Mass Erase or Page Erase. When the Flash Controller is unlocked and Mass Erase is successfully enabled, all Program Memory locations are available for byte programming. In contrast, when the Flash Controller is unlocked and Page Erase is successfully enabled, only the locations of the selected page are available for byte programming. An erased Flash byte contains all 1's (FFH). The programming operation can only be used to change bits from 1 to 0. To change a Flash bit (or multiple bits) from 0 to 1 requires execution of either the Page Erase or Mass Erase commands.

Byte Programming can be accomplished using the On-Chip Debugger's Write Memory command or eZ8 CPU execution of the LDC or LDCI instructions. Refer to the *eZ8 CPU User Manual* (available for download at www.zilog.com) for a description of the LDC and LDCI instructions. While the Flash Controller programs the Flash memory, the eZ8 CPU idles but the system clock and on-chip peripherals continue to operate. To exit programming mode and lock the Flash, write any value to the Flash Control register, except the Mass Erase or Page Erase commands.



Caution: The byte at each address of the Flash memory cannot be programmed (any bits written to 0) more than twice before an erase cycle occurs.

Page Erase

The Flash memory can be erased one page (512 bytes) at a time. Page Erasing the Flash memory sets all bytes in that page to the value FFH. The Flash Page Select register identifies the page to be erased. Only a page residing in an unprotected sector can be erased. With the Flash Controller unlocked and the active page set, writing the value 95h to the Flash Control register initiates the Page Erase operation. While the Flash Controller executes the Page Erase operation, the eZ8 CPU idles but the system clock and on-chip peripherals continue to operate. The eZ8 CPU resumes operation after the Page Erase operation completes. If the Page Erase operation is performed using the On-Chip Debugger, poll the Flash Status register to determine when the Page Erase operation is complete. When the Page Erase is complete, the Flash Controller returns to its locked state.

Mass Erase

The Flash memory can also be Mass Erased using the Flash Controller, but only by using the On-Chip Debugger. Mass Erasing the Flash memory sets all bytes to the value FFH. With the Flash Controller unlocked and the Mass Erase successfully enabled, writing the value 63H to the Flash Control register initiates the Mass Erase operation. While the Flash



Controller executes the Mass Erase operation, the eZ8 CPU idles but the system clock and on-chip peripherals continue to operate. Using the On-Chip Debugger, poll the Flash Status register to determine when the Mass Erase operation is complete. When the Mass Erase is complete, the Flash Controller returns to its locked state.

Flash Controller Bypass

The Flash Controller can be bypassed and the control signals for the Flash memory brought out to the GPIO pins. Bypassing the Flash Controller allows faster Row Programming algorithms by controlling the Flash programming signals directly.

Row programming is recommended for gang programming applications and large volume customers who do not require in-circuit initial programming of the Flash memory. Mass Erase and Page Erase operations are also supported when the Flash Controller is bypassed.

Please refer to the document entitled *Third-Party Flash Programming Support for Z8 Encore!®* for more information about bypassing the Flash Controller. This document is available for download at www.zilog.com.

Flash Control Register Definitions

Flash Control Register

The Flash Controller must be unlocked using the Flash Control register before programming or erasing the Flash memory. Writing the sequence 73H 8CH, sequentially, to the Flash Control register unlocks the Flash Controller. When the Flash Controller is unlocked, the Flash memory can be enabled for Mass Erase or Page Erase by writing the appropriate enable command to the FCTL. Page Erase applies only to the active page selected in Flash Page Select register. Mass Erase is enabled only through the On-Chip Debugger. Writing an invalid value or an invalid sequence returns the Flash Controller to its locked state. The Write-only Flash Control Register shares its Register File address with the read-only Flash Status Register

Table 74. Flash Control Register (FCTL)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------|---|---|---|---|---|---|---|
| FIELD | FCMD | | | | | | | |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | W | W | W | W | W | W | W | W |
| ADDR | FF8H | | | | | | | |



FCMD—Flash Command

73H = First unlock command.

8CH = Second unlock command.

95H = Page Erase command (must be third command in sequence to initiate Page Erase).

63H = Mass Erase command (must be third command in sequence to initiate Mass Erase).

5EH = Enable Flash Sector Protect Register Access

Flash Status Register

The Flash Status register indicates the current state of the Flash Controller. This register can be read at any time. The read-only Flash Status Register shares its Register File address with the Write-only Flash Control Register.

Table 75. Flash Status Register (FSTAT)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----------|---|-------|---|---|---|---|---|
| FIELD | Reserved | | FSTAT | | | | | |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R | R | R | R | R | R | R | R |
| ADDR | FF8H | | | | | | | |

Reserved—Must be 0.

FSTAT—Flash Controller Status

000000 = Flash Controller locked.

000001 = First unlock command received (73H written).

000010 = Second unlock command received (8CH written).

000011 = Flash Controller unlocked.

000100 = Sector protect register selected.

001xxx = Program operation in progress.

010xxx = Page erase operation in progress.

100xxx = Mass erase operation in progress

Flash Page Select Register

The Flash Page Select register shares address space with the Flash Sector Protect Register. Unless the Flash controller is unlocked and written with 5EH, writes to this address target the Flash Page Select Register.

The register is used to select one of the 8 available Flash memory pages to be programmed or erased. Each Flash Page contains 512 bytes of Flash memory. During a Page Erase



operation, all Flash memory having addresses with the most significant 7-bits given by FPS[6:0] are chosen for program/erase operation.

Table 76. Flash Page Select Register (FPS)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----------|------|-----|-----|-----|-----|-----|-----|
| FIELD | Reserved | PAGE | | | | | | |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | FF9H | | | | | | | |

Reserved—Must be 0.

PAGE—Page Select

This 7-bit field identifies the Flash memory page for Page Erase and page unlocking. Program Memory Address[15:9] = PAGE[6:0]. For the Z8F04xx and Z8F02xx devices, the upper 4 bits must always be 0. For the Z8F01xx devices, the upper 5 bits must always be 0.

Flash Sector Protect Register

The Flash Sector Protect register is shared with the Flash Page Select Register. When the **Flash Control Register** is unlocked and written with 5EH, the next write to this address targets the Flash Sector Protect Register. In all other cases, it targets the Flash Page Select Register.

This register selects one of the 8 available Flash memory sectors to be protected. The reset state of each Sector Protect bit is an unprotected state. After a sector is protected by setting its corresponding register bit, it cannot be unprotected (the register bit cannot be cleared) without powering down the device.

Table 77. Flash Sector Protect Register (FPROT)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|--------|--------|--------|--------|--------|--------|--------|--------|
| FIELD | SPROT7 | SPROT6 | SPROT5 | SPROT4 | SPROT3 | SPROT2 | SPROT1 | SPROT0 |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | FF9H | | | | | | | |

SPROT7-SPROT0—Sector Protection

Each bit corresponds to a 512 byte Flash sector. For the Z8F04xx devices all bits are used. For the Z8F02xx devices, the upper 4 bits are unused. For the Z8F01xx devices, the upper 6 bits are unused.



Flash Frequency High and Low Byte Registers

The Flash Frequency High and Low Byte registers combine to form a 16-bit value, FFREQ, to control timing for Flash program and erase operations. The 16-bit binary Flash Frequency value must contain the system clock frequency (in KHz) and is calculated using the following equation:

$$\text{FFREQ}[15:0] = \{ \text{FFREQH}[7:0], \text{FFREQL}[7:0] \} = \frac{\text{System Clock Frequency}}{1000}$$



Caution: Flash programming and erasure is not supported for system clock frequencies below 20KHz or above 5.53MHz. The Flash Frequency High and Low Byte registers must be loaded with the correct value to ensure proper operation of the device.

Table 78. Flash Frequency High Byte Register (FFREQH)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|--------|-----|-----|-----|-----|-----|-----|-----|
| FIELD | FFREQH | | | | | | | |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | FFAH | | | | | | | |

FFREQH—Flash Frequency High Byte
High byte of the 16-bit Flash Frequency value.

Table 79. Flash Frequency Low Byte Register (FFREQL)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|--------|---|---|---|---|---|---|---|
| FIELD | FFREQL | | | | | | | |
| RESET | 0 | | | | | | | |
| R/W | R/W | | | | | | | |
| ADDR | FFBH | | | | | | | |

FFREQL—Flash Frequency Low Byte
Low byte of the 16-bit Flash Frequency value.

Flash Option Bits

Overview

Programmable Flash Option Bits allow user configuration of certain aspects of 4K Series operation. The feature configuration data is stored in the Flash Program Memory and read during Reset. The features available for control through the Flash Option Bits are:

- Watch-Dog Timer time-out response selection—interrupt or system reset
- Watch-Dog Timer enabled at Reset
- The ability to prevent unwanted read access to user code in Program Memory
- The ability to prevent accidental programming and erasure of all or a portion of the user code in Program Memory
- Voltage Brown-Out configuration—always enabled or disabled during STOP mode to reduce STOP mode power consumption
- Factory trimming information for the Internal Precision Oscillator and Temperature Sensor
- Factory calibration values for ADC compensation

Operation

Option Bit Configuration By Reset

Each time the Flash Option Bits are programmed or erased, the device must be Reset for the change to take effect. During any reset operation (System Reset or STOP Mode Recovery), the Flash Option Bits are automatically read from the Flash Program Memory and written to Option Configuration registers. The Option Configuration registers control operation of the devices within the 4K Series. Option Bit control is established before the device exits Reset and the eZ8 CPU begins code execution. The Option Configuration registers are not part of the Register File and are not accessible for read or write access.

Option Bit Types

User Option Bits

The user option bits are contained in the first two bytes of program memory. User access to these bits has been provided because these locations contain application-specific device



configurations. The information contained here is lost when page 0 of the program memory is erased.

Trim Option Bits

The trim option bits are contained in the information page of the Flash memory. These bits are factory programmed values required to optimize the operation of onboard analog circuitry and cannot be permanently altered by the user. Program memory may be erased without endangering these values. It is possible to alter working values of these bits by accessing the Trim Bit Address and Data Registers, but these working values are lost after a power loss.

There are 32 bytes of trim data. To modify one of these values the user code must first write a value between 00H and 1FH into the Trim Bit Address Register. The next write to the Trim Bit Data register changes the working value of the target trim data byte.

Reading the trim data requires the user code to write a value between 00H and 1FH into the Trim Bit Address Register. The next read from the Trim Bit Data register returns the working value of the target trim data byte.

Flash Option Bit Control Register Definitions

Trim Bit Address Register

This register contains the target address for an access to the trim option bits.

Table 80. Trim Bit Address Register (TRMADR)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|--|-----|-----|-----|-----|-----|-----|-----|
| FIELD | TRMADR - Trim Bit Address (00H to 1FH) | | | | | | | |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | FF6H | | | | | | | |

Trim Bit Data Register

This register contains the read or write data for access to the trim option bits.

Table 81. Trim Bit Data Register (TRMDR)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|-----------------------|---|---|---|---|---|---|---|
| FIELD | TRMDR - Trim Bit Data | | | | | | | |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |



Table 81. Trim Bit Data Register (TRMDR)

| | | | | | | | | |
|------|------|-----|-----|-----|-----|-----|-----|-----|
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | FF7H | | | | | | | |

Flash Option Bit Address Space

The first two bytes of Flash Program Memory at addresses 0000H and 0001H are reserved for the user-programmable Flash Option Bits.

Flash Program Memory Address 0000H

Table 82. Flash Option Bits at Program Memory Address 0000H

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----------------------|--------|----------|-----|--------|-----|----------|-----|
| FIELD | WDT_RES | WDT_AO | Reserved | | VBO_AO | FRP | Reserved | FWP |
| RESET | U | U | U | U | U | U | U | U |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | Program Memory 0000H | | | | | | | |

Note: U = Unchanged by Reset. R/W = Read/Write.

WDT_RES—Watch-Dog Timer Reset

0 = Watch-Dog Timer time-out generates an interrupt request. Interrupts must be globally enabled for the eZ8 CPU to acknowledge the interrupt request.

1 = Watch-Dog Timer time-out causes a system reset. This setting is the default for unprogrammed (erased) Flash.

WDT_AO—Watch-Dog Timer Always On

0 = Watch-Dog Timer is automatically enabled upon application of system power. Watch-Dog Timer can not be disabled.

1 = Watch-Dog Timer is enabled upon execution of the WDT instruction. Once enabled, the Watch-Dog Timer can only be disabled by a Reset or STOP Mode Recovery. This setting is the default for unprogrammed (erased) Flash.

VBO_AO—Voltage Brown-Out Protection Always On

0 = Voltage Brown-Out Protection is disabled in STOP mode to reduce total power consumption.

1 = Voltage Brown-Out Protection is always enabled including during STOP mode. This setting is the default for unprogrammed (erased) Flash.

FRP—Flash Read Protect

0 = User program code is inaccessible. Limited control features are available through the On-Chip Debugger.



1 = User program code is accessible. All On-Chip Debugger commands are enabled. This setting is the default for unprogrammed (erased) Flash.

Reserved—Must be 1.

FWP—Flash Write Protect

This Option Bit provides Flash Program Memory protection:

0 = Programming and erasure disabled for all of Flash Program Memory. Programming, Page Erase, and Mass Erase through User Code is disabled. Mass Erase is available using the On-Chip Debugger.

1 = Programming, Page Erase, and Mass Erase are enabled for all of Flash Program Memory.

Flash Program Memory Address 0001H

Table 83. Flash Options Bits at Program Memory Address 0001H

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|----------------------|-----|-----|-----|-----|-----|-----|-----|
| FIELD | Reserved | | | | | | | |
| RESET | U | U | U | U | U | U | U | U |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | Program Memory 0001H | | | | | | | |
| Note: U = Unchanged by Reset. R/W = Read/Write. | | | | | | | | |

Reserved—Must be 1.

Trim Bit Address Space

Trim Bit Address 0000H - Reserved

Trim Bit Address 0001H - Reserved

Trim Bit Address 0002H

Table 84. Trim Option Bits at 0002H (TIPO)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----------|---|---|---|---|---|---|---|
| FIELD | IPO_TRIM | | | | | | | |
| RESET | U | | | | | | | |



Table 84. Trim Option Bits at 0002H (TIPO)

| | |
|---|-------------------------------|
| R/W | R/W |
| ADDR | Information Page Memory 0022H |
| Note: U = Unchanged by Reset. R/W = Read/Write. | |

IPO_TRIM—Internal Precision Oscillator Trim Byte
Contains trimming bits for Internal Precision Oscillator.

Trim Bit Address 0003H - Reserved

Trim Bit Address 0004H - Reserved

On-Chip Debugger

Overview

The Z8 Encore!® 4K Series devices contain an integrated On-Chip Debugger (OCD) that provides advanced debugging features including:

- Reading and writing of the Register File
- Reading and writing of Program and Data Memory
- Setting of Breakpoints and Watchpoints
- Executing eZ8 CPU instructions

Architecture

The On-Chip Debugger consists of four primary functional blocks: transmitter, receiver, auto-baud detector/generator, and debug controller. Figure 22 illustrates the architecture of the On-Chip Debugger

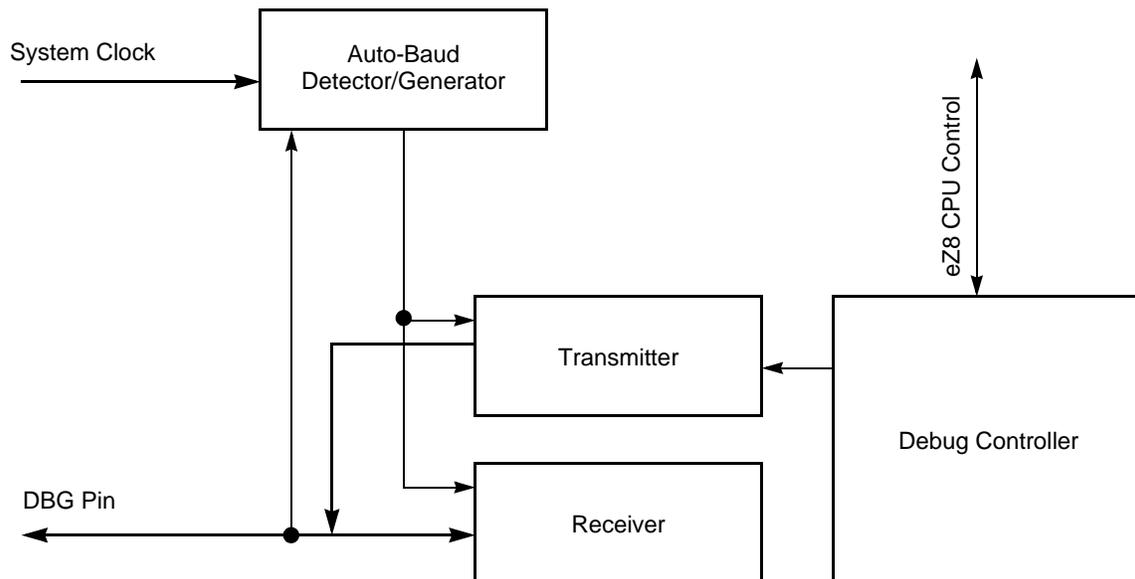


Figure 22. On-Chip Debugger Block Diagram

Operation

OCD Interface

The On-Chip Debugger uses the DBG pin for communication with an external host. This one-pin interface is a bi-directional open-drain interface that transmits and receives data. Data transmission is half-duplex, in that transmit and receive cannot occur simultaneously. The serial data on the DBG pin is sent using the standard asynchronous data format defined in RS-232. This pin creates an interface from the PartName products to the serial port of a host PC using minimal external hardware. Two different methods for connecting the DBG pin to an RS-232 interface are depicted in Figures 23 and 24. The recommended method is the buffered implementation depicted in Figure 24. The DBG pin must always be connected to V_{DD} through an external pull-up resistor.

**Caution:**

For operation of the On-Chip Debugger, all power pins (V_{DD} and AV_{DD}) must be supplied with power, and all ground pins (V_{SS} and AV_{SS}) must be properly grounded.

The DBG pin is open-drain and must always be connected to V_{DD} through an external pull-up resistor to insure proper operation.

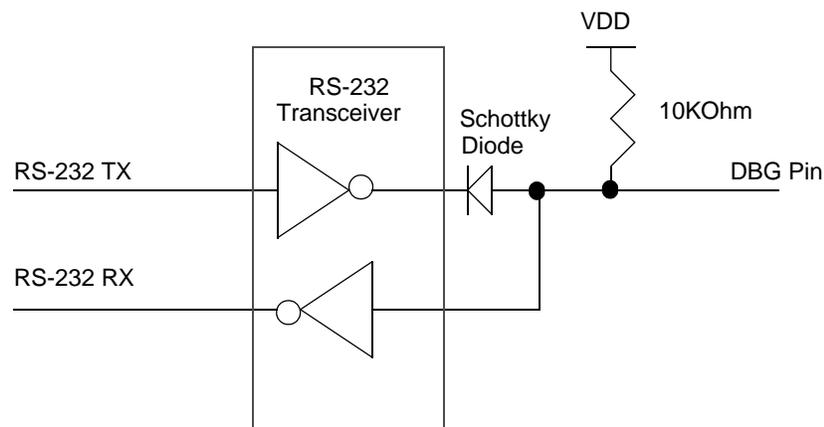


Figure 23. Interfacing the On-Chip Debugger's DBG Pin with an RS-232 Interface (1)

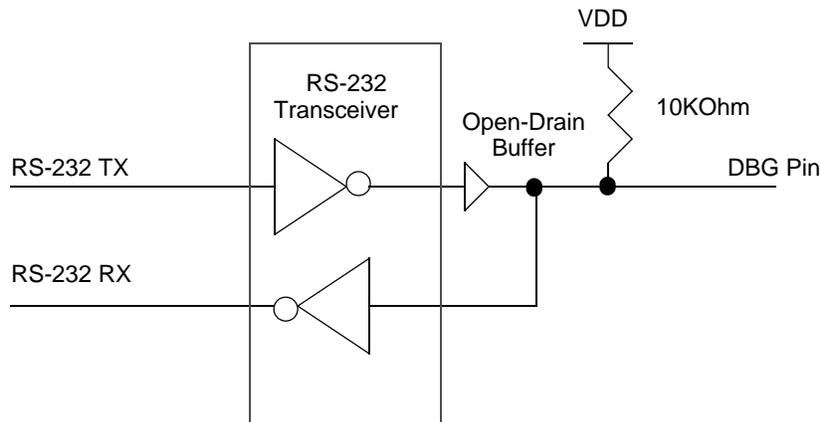


Figure 24. Interfacing the On-Chip Debugger's DBG Pin with an RS-232 Interface (2)

DEBUG Mode

The operating characteristics of the devices in DEBUG mode are:

- The eZ8 CPU fetch unit stops, idling the eZ8 CPU, unless directed by the OCD to execute specific instructions
- The system clock operates unless in STOP mode
- All enabled on-chip peripherals operate unless in STOP mode
- Automatically exits HALT mode
- Constantly refreshes the Watch-Dog Timer, if enabled

Entering DEBUG Mode

- The device enters DEBUG mode after the eZ8 CPU executes a BRK (Breakpoint) instruction.
- If the DBG pin is held Low during the most recent clock cycle of system reset, the part enters DEBUG mode upon exiting system reset.

Exiting DEBUG Mode

The device exits DEBUG mode following any of these operations:

- Clearing the DBGMODE bit in the OCD Control Register to 0.
- Power-on reset
- Voltage Brown-Out reset
- Watch-Dog Timer reset



- Asserting the $\overline{\text{RESET}}$ pin Low to initiate a Reset.
- Driving the DBG pin Low while the device is in STOP mode initiates a System Reset.

OCD Data Format

The OCD interface uses the asynchronous data format defined for RS-232. Each character is transmitted as 1 Start bit, 8 data bits (least-significant bit first), and 1.5 Stop bits (Figure 25)



Figure 25. OCD Data Format

OCD Auto-Baud Detector/Generator

To run over a range of baud rates (data bits per second) with various system clock frequencies, the On-Chip Debugger contains an Auto-Baud Detector/Generator. After a reset, the OCD is idle until it receives data. The OCD requires that the first character sent from the host is the character 80H. The character 80H has eight continuous bits Low (one Start bit plus 7 data bits), framed between High bits. The Auto-Baud Detector measures this period and sets the OCD Baud Rate Generator accordingly.

The Auto-Baud Detector/Generator is clocked by the system clock. The minimum baud rate is the system clock frequency divided by 512. For optimal operation with asynchronous datastreams, the maximum recommended baud rate is the system clock frequency divided by 8. The maximum possible baud rate for asynchronous datastreams is the system clock frequency divided by 4, but this theoretical maximum is possible only for low noise designs with clean signals. Table 85 lists minimum and recommended maximum baud rates for IPO frequencies.

Table 85. OCD Baud-Rate Limits

| System Clock Frequency (MHz) | Recommended Maximum Baud Rate (Kbps) | Recommended Standard PC Baud Rate (bps) | Minimum Baud Rate (Kbps) |
|------------------------------|--------------------------------------|---|--------------------------|
| 5.5296 | 1382.4 | 691,200 | 1.08 |
| 0.032768 (32KHz) | 4.096 | 2400 | 0.064 |

If the OCD receives a Serial Break (nine or more continuous bits Low) the Auto-Baud Detector/Generator resets. Reconfigure the Auto-Baud Detector/Generator by sending 80H.



OCD Serial Errors

The On-Chip Debugger can detect any of the following error conditions on the DBG pin:

- Serial Break (a minimum of nine continuous bits Low)
- Framing Error (received `STOP` bit is Low)
- Transmit Collision (OCD and host simultaneous transmission detected by the OCD)

When the OCD detects one of these errors, it aborts any command currently in progress, transmits a four character long Serial Break back to the host, and resets the Auto-Baud Detector/Generator. A Framing Error or Transmit Collision may be caused by the host sending a Serial Break to the OCD. Because of the open-drain nature of the interface, returning a Serial Break back to the host only extends the length of the Serial Break if the host releases the Serial Break early.

The host transmits a Serial Break on the DBG pin when first connecting to the 4K Series devices or when recovering from an error. A Serial Break from the host resets the Auto-Baud Generator/Detector but does not reset the OCD Control register. A Serial Break leaves the device in DEBUG mode if that is the current mode. The OCD is held in Reset until the end of the Serial Break when the DBG pin returns High. Because of the open-drain nature of the DBG pin, the host can send a Serial Break to the OCD even if the OCD is transmitting a character.

Breakpoints

Execution Breakpoints are generated using the BRK instruction (opcode `00H`). When the eZ8 CPU decodes a BRK instruction, it signals the On-Chip Debugger. If Breakpoints are enabled, the OCD enters DEBUG mode and idles the eZ8 CPU. If Breakpoints are not enabled, the OCD ignores the BRK signal and the BRK instruction operates as an NOP instruction.

Breakpoints in Flash Memory

The BRK instruction is opcode `00H`, which corresponds to the fully programmed state of a byte in Flash memory. To implement a Breakpoint, write `00H` to the required break address, overwriting the current instruction. To remove a Breakpoint, the corresponding page of Flash memory must be erased and reprogrammed with the original data.

Runtime Counter

The On-Chip Debugger contains a 16-bit Runtime Counter. It counts system clock cycles between Breakpoints. The counter starts counting when the On-Chip Debugger leaves DEBUG mode and stops counting when it enters DEBUG mode again or when it reaches the maximum count of `FFFFH`.



On-Chip Debugger Commands

The host communicates to the On-Chip Debugger by sending OCD commands using the DBG interface. During normal operation, only a subset of the OCD commands are available. In DEBUG mode, all OCD commands become available unless the user code and control registers are protected by programming the Flash Read Protect Option bit (FRP). The Flash Read Protect Option bit prevents the code in memory from being read out of the 4K Series products. When this option is enabled, several of the OCD commands are disabled. Table 86 on page 141 is a summary of the On-Chip Debugger commands. Each OCD command is described in further detail in the bulleted list following this table. Table 86 also indicates those commands that operate when the device is not in DEBUG mode (normal operation) and those commands that are disabled by programming the Flash Read Protect Option bit.

| Debug Command | Command Byte | Enabled when NOT in DEBUG mode? | Disabled by Flash Read Protect Option Bit |
|----------------------------|--------------|---------------------------------|--|
| Read OCD Revision | 00H | Yes | – |
| Reserved | 01H | – | – |
| Read OCD Status Register | 02H | Yes | – |
| Read Runtime Counter | 03H | – | – |
| Write OCD Control Register | 04H | Yes | Cannot clear DBGMODE bit |
| Read OCD Control Register | 05H | Yes | – |
| Write Program Counter | 06H | – | Disabled |
| Read Program Counter | 07H | – | Disabled |
| Write Register | 08H | – | Only writes of the Flash Memory Control registers are allowed. Additionally, only the Mass Erase command is allowed to be written to the Flash Control register. |
| Read Register | 09H | – | Disabled |
| Write Program Memory | 0AH | – | Disabled |
| Read Program Memory | 0BH | – | Disabled |
| Write Data Memory | 0CH | – | Yes |
| Read Data Memory | 0DH | – | – |
| Read Program Memory CRC | 0EH | – | – |
| Reserved | 0FH | – | – |
| Step Instruction | 10H | – | Disabled |



| Debug Command | Command Byte | Enabled when NOT in DEBUG mode? | Disabled by Flash Read Protect Option Bit |
|---------------------|--------------|---------------------------------|---|
| Stuff Instruction | 11H | – | Disabled |
| Execute Instruction | 12H | – | Disabled |
| Reserved | 13H–FFH | – | – |

In the following bulleted list of OCD Commands, data and commands sent from the host to the On-Chip Debugger are identified by 'DBG ← Command/Data'. Data sent from the On-Chip Debugger back to the host is identified by 'DBG → Data'

- **Read OCD Revision (00H)**—The Read OCD Revision command determines the version of the On-Chip Debugger. If OCD commands are added, removed, or changed, this revision number changes.

```
DBG ← 00H
DBG → OCDRev[15:8] (Major revision number)
DBG → OCDRev[7:0] (Minor revision number)
```

- **Read OCD Status Register (02H)**—The Read OCD Status Register command reads the OCDSTAT register.

```
DBG ← 02H
DBG → OCDSTAT[7:0]
```

- **Read Runtime Counter (03H)**—The Runtime Counter counts system clock cycles in between Breakpoints. The 16-bit Runtime Counter counts up from 0000H and stops at the maximum count of FFFFH. The Runtime Counter is overwritten during the Write Memory, Read Memory, Write Register, Read Register, Read Memory CRC, Step Instruction, Stuff Instruction, and Execute Instruction commands.

```
DBG ← 03H
DBG → RuntimeCounter[15:8]
DBG → RuntimeCounter[7:0]
```

- **Write OCD Control Register (04H)**—The Write OCD Control Register command writes the data that follows to the OCDCTL register. When the Flash Read Protect Option Bit is enabled, the DBGMODE bit (OCDCTL[7]) can only be set to 1, it cannot be cleared to 0 and the only method of returning the device to normal operating mode is to reset the device.

```
DBG ← 04H
DBG ← OCDCTL[7:0]
```

- **Read OCD Control Register (05H)**—The Read OCD Control Register command reads the value of the OCDCTL register.



```
DBG ← 05H
DBG → OCDCTL[7:0]
```

- **Write Program Counter (06H)**—The Write Program Counter command writes the data that follows to the eZ8 CPU's Program Counter (PC). If the device is not in DEBUG mode or if the Flash Read Protect Option bit is enabled, the Program Counter (PC) values are discarded.

```
DBG ← 06H
DBG ← ProgramCounter[15:8]
DBG ← ProgramCounter[7:0]
```

- **Read Program Counter (07H)**—The Read Program Counter command reads the value in the eZ8 CPU's Program Counter (PC). If the device is not in DEBUG mode or if the Flash Read Protect Option bit is enabled, this command returns FFFFH.

```
DBG ← 07H
DBG → ProgramCounter[15:8]
DBG → ProgramCounter[7:0]
```

- **Write Register (08H)**—The Write Register command writes data to the Register File. Data can be written 1–256 bytes at a time (256 bytes can be written by setting size to 0). If the device is not in DEBUG mode, the address and data values are discarded. If the Flash Read Protect Option bit is enabled, only writes to the Flash Control Registers are allowed and all other register write data values are discarded.

```
DBG ← 08H
DBG ← {4'h0, Register Address[11:8]}
DBG ← Register Address[7:0]
DBG ← Size[7:0]
DBG ← 1-256 data bytes
```

- **Read Register (09H)**—The Read Register command reads data from the Register File. Data can be read 1–256 bytes at a time (256 bytes can be read by setting size to 0). If the device is not in DEBUG mode or if the Flash Read Protect Option bit is enabled, this command returns FFH for all the data values.

```
DBG ← 09H
DBG ← {4'h0, Register Address[11:8]}
DBG ← Register Address[7:0]
DBG ← Size[7:0]
DBG → 1-256 data bytes
```

- **Write Program Memory (0AH)**—The Write Program Memory command writes data to Program Memory. This command is equivalent to the LDC and LDCI instructions. Data can be written 1–65536 bytes at a time (65536 bytes can be written by setting size to 0). The on-chip Flash Controller must be written to and unlocked for the programming operation to occur. If the Flash Controller is not unlocked, the data is discarded. If the device



is not in DEBUG mode or if the Flash Read Protect Option bit is enabled, the data is discarded.

```
DBG ← 0AH
DBG ← Program Memory Address[15:8]
DBG ← Program Memory Address[7:0]
DBG ← Size[15:8]
DBG ← Size[7:0]
DBG ← 1-65536 data bytes
```

- **Read Program Memory (0BH)**—The Read Program Memory command reads data from Program Memory. This command is equivalent to the LDC and LDCI instructions. Data can be read 1–65536 bytes at a time (65536 bytes can be read by setting size to 0). If the device is not in DEBUG mode or if the Flash Read Protect Option Bit is enabled, this command returns FFH for the data.

```
DBG ← 0BH
DBG ← Program Memory Address[15:8]
DBG ← Program Memory Address[7:0]
DBG ← Size[15:8]
DBG ← Size[7:0]
DBG → 1-65536 data bytes
```

- **Write Data Memory (0CH)**—The Write Data Memory command writes data to Data Memory. This command is equivalent to the LDE and LDEI instructions. Data can be written 1–65536 bytes at a time (65536 bytes can be written by setting size to 0). If the device is not in DEBUG mode or if the Flash Read Protect Option Bit is enabled, the data is discarded.

```
DBG ← 0CH
DBG ← Data Memory Address[15:8]
DBG ← Data Memory Address[7:0]
DBG ← Size[15:8]
DBG ← Size[7:0]
DBG ← 1-65536 data bytes
```

- **Read Data Memory (0DH)**—The Read Data Memory command reads from Data Memory. This command is equivalent to the LDE and LDEI instructions. Data can be read 1 to 65536 bytes at a time (65536 bytes can be read by setting size to 0). If the device is not in DEBUG mode, this command returns FFH for the data.

```
DBG ← 0DH
DBG ← Data Memory Address[15:8]
DBG ← Data Memory Address[7:0]
DBG ← Size[15:8]
DBG ← Size[7:0]
DBG → 1-65536 data bytes
```



- **Read Program Memory CRC (0EH)**—The Read Program Memory CRC command computes and returns the Cyclic Redundancy Check (CRC) of Program Memory using the 16-bit CRC-CCITT polynomial. If the device is not in DEBUG mode, this command returns FFFFH for the CRC value. Unlike most other OCD Read commands, there is a delay from issuing of the command until the OCD returns the data. The OCD reads the Program Memory, calculates the CRC value, and returns the result. The delay is a function of the Program Memory size and is approximately equal to the system clock period multiplied by the number of bytes in the Program Memory.

```
DBG ← 0EH
DBG → CRC [15:8]
DBG → CRC [7:0]
```

- **Step Instruction (10H)**—The Step Instruction command steps one assembly instruction at the current Program Counter (PC) location. If the device is not in DEBUG mode or the Flash Read Protect Option bit is enabled, the OCD ignores this command.

```
DBG ← 10H
```

- **Stuff Instruction (11H)**—The Stuff Instruction command steps one assembly instruction and allows specification of the first byte of the instruction. The remaining 0-4 bytes of the instruction are read from Program Memory. This command is useful for stepping over instructions where the first byte of the instruction has been overwritten by a Breakpoint. If the device is not in DEBUG mode or the Flash Read Protect Option bit is enabled, the OCD ignores this command.

```
DBG ← 11H
DBG ← opcode [7:0]
```

- **Execute Instruction (12H)**—The Execute Instruction command allows sending an entire instruction to be executed to the eZ8 CPU. This command can also step over Breakpoints. The number of bytes to send for the instruction depends on the opcode. If the device is not in DEBUG mode or the Flash Read Protect Option bit is enabled, this command reads and discards one byte.

```
DBG ← 12H
DBG ← 1-5 byte opcode
```

On-Chip Debugger Control Register Definitions

OCD Control Register

The OCD Control register controls the state of the On-Chip Debugger. This register is used to enter or exit DEBUG mode and to enable the BRK instruction. It can also reset the 4K Series device.



A reset and stop function can be achieved by writing 81H to this register. A reset and go function can be achieved by writing 41H to this register. If the device is in DEBUG mode, a run function can be implemented by writing 40H to this register.

Table 86. OCD Control Register (OCDCTL)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------------|----------|----------|----------|----------|----------|----------|----------|----------|
| FIELD | DBGMODE | BRKEN | DBGACK | Reserved | | | | RST |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R | R | R | R | R/W |

DBGMODE—Debug Mode

The device enters DEBUG mode when this bit is 1. When in DEBUG mode, the eZ8 CPU stops fetching new instructions. Clearing this bit causes the eZ8 CPU to restart. This bit is automatically set when a BRK instruction is decoded and Breakpoints are enabled. If the Flash Read Protect Option Bit is enabled, this bit can only be cleared by resetting the device. It cannot be written to 0.

0 = The 4K Series device is operating in NORMAL mode.

1 = The 4K Series device is in DEBUG mode.

BRKEN—Breakpoint Enable

This bit controls the behavior of the BRK instruction (opcode 00H). By default, Breakpoints are disabled and the BRK instruction behaves similar to an NOP instruction. If this bit is 1, when a BRK instruction is decoded, the DBGMODE bit of the OCDCTL register is automatically set to 1.

0 = Breakpoints are disabled.

1 = Breakpoints are enabled.

DBGACK—Debug Acknowledge

This bit enables the debug acknowledge feature. If this bit is set to 1, the OCD sends a Debug Acknowledge character (FFH) to the host when a Breakpoint occurs.

0 = Debug Acknowledge is disabled.

1 = Debug Acknowledge is enabled.

Reserved—Must be 0.

RST—Reset

Setting this bit to 1 resets the Z8F04xA family device. The device goes through a normal Power-On Reset sequence with the exception that the On-Chip Debugger is not reset. This bit is automatically cleared to 0 at the end of reset.

0 = No effect.

1 = Reset the Flash Read Protect Option Bit device.



OCD Status Register

The OCD Status register reports status information about the current state of the debugger and the system.

Table 87. OCD Status Register (OCDSTAT)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------------|----------|----------|----------|----------|----------|----------|----------|----------|
| FIELD | DBG | HALT | FRPENB | Reserved | | | | |
| RESET | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R | R | R | R | R | R | R | R |

DBG—Debug Status

0 = NORMAL mode

1 = DEBUG mode

HALT—HALT Mode

0 = Not in HALT mode

1 = In HALT mode

FRPENB—Flash Read Protect Option Bit Enable

0 = FRP bit enabled, that allows disabling of many OCD commands

1 = FRP bit has no effect

Reserved—Must be 0.



Oscillator Control

Overview

The PartName devices uses three possible clocking schemes, each user-selectable:

- On-chip precision trimmed RC oscillator
- External clock drive
- On-chip low precision Watch-Dog Timer oscillator

Operation

This chapter discusses the logic used to select the system clock. A description of the specific operation of each oscillator is outlined elsewhere in this document. The detailed description of the Watch-Dog Timer Oscillator starts on page 79 and the Internal Precision Oscillator description begins on page 146.

System Clock Selection

The oscillator control block selects from the available clocks. Table 88 details each clock source and its usage.

Table 88. Oscillator Configuration and Selection

| Clock Source | Characteristics | Required Setup |
|------------------------------------|---|---|
| Internal Precision RC Oscillator | <ul style="list-style-type: none"> • 32.8KHz or 5.53MHz • $\pm 4\%$ accuracy when trimmed • No external components required | <ul style="list-style-type: none"> • Unlock and write Oscillator Control Register (OSCCTL) to enable and select oscillator at either 5.53MHz or 32.8KHz |
| External Clock Drive | <ul style="list-style-type: none"> • 0 to 5.53MHz • Accuracy dependent on external clock source | <ul style="list-style-type: none"> • Write GPIO registers to configure PB3 pin for external clock function • Unlock and write OSCCTL to select external system clock • Apply external clock signal to GPIO |
| Internal Watchdog Timer Oscillator | <ul style="list-style-type: none"> • 32KHz nominal • $\pm 40\%$ accuracy; no external components required • Low power consumption | <ul style="list-style-type: none"> • Enable WDT if not enabled and wait until WDT Oscillator is operating. • Unlock and write Oscillator Control Register (OSCCTL) to enable and select oscillator |



Caution: Unintentional accesses to the oscillator control register can actually stop the chip by switching to a non-functioning oscillator. To prevent this condition, the oscillator control block employs a register unlocking/locking scheme.

OSC Control Register Unlocking/Locking

To write the oscillator control register, unlock it by making two writes to the OSCCTL register with the values E7H followed by 18H. A third write to the OSCCTL register changes the value of the actual register and returns the register to a locked state. Any other sequence of oscillator control register writes has no effect. The values written to unlock the register must be ordered correctly, but are not necessarily consecutive. It is possible to write to or read from other registers within the unlocking/locking operation.

The internal precision oscillator is enabled by default. If the user code changes to a different oscillator, it may be appropriate to disable the IPO for power savings. Disabling the IPO does not occur automatically.

Oscillator Control Register Definitions

Oscillator Control Register

The Oscillator Control Register (OSCCTL) enables/disables the various oscillator circuits and selects the primary oscillator, which becomes the system clock.

The Oscillator Control Register must be unlocked before writing. Writing the two step sequence E7H followed by 18H to the Oscillator Control Register unlocks it. The register is locked at successful completion of a register write to the OSCCTL.

Table 89. Oscillator Control Register (OSCCTL)

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|-------|----------|-------|----------|----------|--------|-----|-----|
| FIELD | INTEN | Reserved | WDTEN | Reserved | Reserved | SCKSEL | | |
| RESET | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| ADDR | F86H | | | | | | | |

INTEN—Internal Precision Oscillator Enable

1 = Internal precision oscillator is enabled

0 = Internal precision oscillator is disabled

WDTEN—Watchdog Timer Oscillator Enable

1 = Watch-Dog Timer oscillator is enabled

0 = Watch-Dog Timer oscillator is disabled



SCKSEL—System Clock Oscillator Select

000 = Internal precision oscillator functions as system clock at 5.53MHz

001 = Internal precision oscillator functions as system clock at 32KHz

010 = Reserved - Must be 0

011 = Watch-Dog Timer oscillator functions as system

100 = External clock signal on PB3 functions as system clock

101 = Reserved - Must be 0

110 = Reserved - Must be 0

111 = Reserved - Must be 0



Caution: It is possible to disable all functioning clock sources. In this event the 4K Series device ceases functioning and can only be recovered by a Power-On Reset.

Internal Precision Oscillator

Overview

The Internal Precision Oscillator (IPO) is designed for use without external components. The user can either manually trim the oscillator for a non-standard frequency or use the automatic factory-trimmed version to achieve a 5.53MHz frequency with $\pm 4\%$ accuracy over the operating temperature and supply voltage range of the device. IPO features include:

- On-chip RC oscillator that does not require external components
- Output frequency of either 5.53MHz or 32.8KHz (contains both a fast and a slow mode)
- Trimming possible through Flash option bits with user override

Operation

The internal oscillator is an RC relaxation oscillator that has had its sensitivity to power supply variation minimized. By using ratio tracking thresholds, the effect of power supply voltage is cancelled out. The dominant source of oscillator error is the absolute variance of chip level fabricated components, such as capacitors. An 8-bit trimming register, incorporated into the design, compensates for absolute variation of oscillator frequency. Once trimmed the oscillator frequency is relatively stable and does not require subsequent calibration. Trimming was performed during manufacturing and is not necessary for the user to repeat unless a frequency other than 5.53MHz (fast mode) or 32.8kHz (slow mode) is required.

Power down this block for minimum system power.

By default, the oscillator is configured through the Flash Option bits. However, the user code can override these trim values as described in “Trim Bit Address Space” on page 129.

Select one of two frequencies for the oscillator: 5.53MHz and 32.8KHz, using the OSC-SEL bits in the “Oscillator Control” on page 143.

eZ8 CPU Instruction Set

Assembly Language Programming Introduction

The eZ8 CPU assembly language provides a means for writing an application program without concern for actual memory addresses or machine instruction formats. A program written in assembly language is called a source program. Assembly language allows the use of symbolic addresses to identify memory locations. It also allows mnemonic codes (opcodes and operands) to represent the instructions themselves. The opcodes identify the instruction while the operands represent memory locations, registers, or immediate data values.

Each assembly language program consists of a series of symbolic commands called statements. Each statement can contain labels, operations, operands and comments.

Labels can be assigned to a particular instruction step in a source program. The label identifies that step in the program as an entry point for use by other instructions.

The assembly language also includes assembler directives that supplement the machine instruction. The assembler directives, or pseudo-ops, are not translated into a machine instruction. Rather, the pseudo-ops are interpreted as directives that control or assist the assembly process.

The source program is processed (assembled) by the assembler to obtain a machine language program called the object code. The object code is executed by the eZ8 CPU. An example segment of an assembly language program is detailed in the following example.

Assembly Language Source Program Example

```
JP START      ; Everything after the semicolon is a comment.

START:        ; A label called "START". The first instruction (JP START) in this
              ; example causes program execution to jump to the point within the
              ; program where the START label occurs.

LD R4, R7     ; A Load (LD) instruction with two operands. The first operand,
              ; Working Register R4, is the destination. The second operand,
              ; Working Register R7, is the source. The contents of R7 is
              ; written into R4.

LD 234H, #%01 ; Another Load (LD) instruction with two operands.
              ; The first operand, Extended Mode Register Address 234H,
              ; identifies the destination. The second operand, Immediate Data
              ; value 01H, is the source. The value 01H is written into the
              ; Register at address 234H.
```



Assembly Language Syntax

For proper instruction execution, eZ8 CPU assembly language syntax requires that the operands be written as ‘destination, source’. After assembly, the object code usually has the operands in the order ‘source, destination’, but ordering is opcode-dependent. The following instruction examples illustrate the format of some basic assembly instructions and the resulting object code produced by the assembler. This binary format must be followed by users that prefer manual program coding or intend to implement their own assembler.

Example 1: If the contents of Registers 43H and 08H are added and the result is stored in 43H, the assembly syntax and resulting object code is:

Table 90. Assembly Language Syntax Example 1

| | | | | |
|-------------------------------|-----|------|-----|----------------|
| Assembly Language Code | ADD | 43H, | 08H | (ADD dst, src) |
| Object Code | 04 | 08 | 43 | (OPC src, dst) |

Example 2: In general, when an instruction format requires an 8-bit register address, that address can specify any register location in the range 0–255 or, using Escaped Mode Addressing, a Working Register R0–R15. If the contents of Register 43H and Working Register R8 are added and the result is stored in 43H, the assembly syntax and resulting object code is:

Table 91. Assembly Language Syntax Example 2

| | | | | |
|-------------------------------|-----|------|----|----------------|
| Assembly Language Code | ADD | 43H, | R8 | (ADD dst, src) |
| Object Code | 04 | E8 | 43 | (OPC src, dst) |

See the device-specific Product Specification to determine the exact register file range available. The register file size varies, depending on the device type.

eZ8 CPU Instruction Notation

In the eZ8 CPU Instruction Summary and Description sections, the operands, condition codes, status flags, and address modes are represented by a notational shorthand that is described in Table 92



Table 92. Notational Shorthand

| Notation | Description | Operand | Range |
|-----------------|--------------------------------|----------------|---|
| b | Bit | b | b represents a value from 0 to 7 (000B to 111B). |
| cc | Condition Code | — | See Condition Codes overview in the eZ8 CPU User Manual. |
| DA | Direct Address | Addr | Addr. represents a number in the range of 0000H to FFFFH |
| ER | Extended Addressing Register | Reg | Reg. represents a number in the range of 000H to FFFH |
| IM | Immediate Data | #Data | Data is a number between 00H to FFH |
| Ir | Indirect Working Register | @Rn | n = 0 –15 |
| IR | Indirect Register | @Reg | Reg. represents a number in the range of 00H to FFH |
| Irr | Indirect Working Register Pair | @RRp | p = 0, 2, 4, 6, 8, 10, 12, or 14 |
| IRR | Indirect Register Pair | @Reg | Reg. represents an even number in the range 00H to FEH |
| p | Polarity | p | Polarity is a single bit binary value of either 0B or 1B. |
| r | Working Register | Rn | n = 0 – 15 |
| R | Register | Reg | Reg. represents a number in the range of 00H to FFH |
| RA | Relative Address | X | X represents an index in the range of +127 to –128 which is an offset relative to the address of the next instruction |
| rr | Working Register Pair | RRp | p = 0, 2, 4, 6, 8, 10, 12, or 14 |
| RR | Register Pair | Reg | Reg. represents an even number in the range of 00H to FEH |
| Vector | Vector Address | Vector | Vector represents a number in the range of 00H to FFH |
| X | Indexed | #Index | The register or register pair to be indexed is offset by the signed Index value (#Index) in a +127 to -128 range. |

Table 93 contains additional symbols that are used throughout the Instruction Summary and Instruction Set Description sections.

**Table 93. Additional Symbols**

| Symbol | Definition |
|--------|---------------------------|
| dst | Destination Operand |
| src | Source Operand |
| @ | Indirect Address Prefix |
| SP | Stack Pointer |
| PC | Program Counter |
| FLAGS | Flags Register |
| RP | Register Pointer |
| # | Immediate Operand Prefix |
| B | Binary Number Suffix |
| % | Hexadecimal Number Prefix |
| H | Hexadecimal Number Suffix |

Assignment of a value is indicated by an arrow. For example,

$$\text{dst} \leftarrow \text{dst} + \text{src}$$

indicates the source data is added to the destination data and the result is stored in the destination location.

eZ8 CPU Instruction Classes

eZ8 CPU instructions can be divided functionally into the following groups:

- Arithmetic
- Bit Manipulation
- Block Transfer
- CPU Control
- Load
- Logical
- Program Control
- Rotate and Shift

Tables 94 through 101 contain the instructions belonging to each group and the number of operands required for each instruction. Some instructions appear in more than one table as these instruction can be considered as a subset of more than one category. Within these tables, the source operand is identified as 'src', the destination operand is 'dst' and a condition code is 'cc'.

Table 94. Arithmetic Instructions

| Mnemonic | Operands | Instruction |
|-----------------|-----------------|---|
| ADC | dst, src | Add with Carry |
| ADCX | dst, src | Add with Carry using Extended Addressing |
| ADD | dst, src | Add |
| ADDX | dst, src | Add using Extended Addressing |
| CP | dst, src | Compare |
| CPC | dst, src | Compare with Carry |
| CPCX | dst, src | Compare with Carry using Extended Addressing |
| CPX | dst, src | Compare using Extended Addressing |
| DA | dst | Decimal Adjust |
| DEC | dst | Decrement |
| DECW | dst | Decrement Word |
| INC | dst | Increment |
| INCW | dst | Increment Word |
| MULT | dst | Multiply |
| SBC | dst, src | Subtract with Carry |
| SBCX | dst, src | Subtract with Carry using Extended Addressing |
| SUB | dst, src | Subtract |
| SUBX | dst, src | Subtract using Extended Addressing |

Table 95. Bit Manipulation Instructions

| Mnemonic | Operands | Instruction |
|-----------------|-----------------|--|
| BCLR | bit, dst | Bit Clear |
| BIT | p, bit, dst | Bit Set or Clear |
| BSET | bit, dst | Bit Set |
| BSWAP | dst | Bit Swap |
| CCF | — | Complement Carry Flag |
| RCF | — | Reset Carry Flag |
| SCF | — | Set Carry Flag |
| TCM | dst, src | Test Complement Under Mask |
| TCMX | dst, src | Test Complement Under Mask using Extended Addressing |
| TM | dst, src | Test Under Mask |
| TMX | dst, src | Test Under Mask using Extended Addressing |

Table 96. Block Transfer Instructions

| Mnemonic | Operands | Instruction |
|-----------------|-----------------|---|
| LDCI | dst, src | Load Constant to/from Program Memory and Auto-Increment Addresses |
| LDEI | dst, src | Load External Data to/from Data Memory and Auto-Increment Addresses |

Table 97. CPU Control Instructions

| Mnemonic | Operands | Instruction |
|-----------------|-----------------|-----------------------|
| CCF | — | Complement Carry Flag |
| DI | — | Disable Interrupts |
| EI | — | Enable Interrupts |
| HALT | — | Halt Mode |
| NOP | — | No Operation |
| RCF | — | Reset Carry Flag |
| SCF | — | Set Carry Flag |
| SRP | src | Set Register Pointer |



Table 97. CPU Control Instructions

| Mnemonic | Operands | Instruction |
|----------|----------|-------------------------|
| STOP | — | STOP Mode |
| WDT | — | Watch-Dog Timer Refresh |

Table 98. Load Instructions

| Mnemonic | Operands | Instruction |
|----------|-------------|---|
| CLR | dst | Clear |
| LD | dst, src | Load |
| LDC | dst, src | Load Constant to/from Program Memory |
| LDCI | dst, src | Load Constant to/from Program Memory and Auto-Increment Addresses |
| LDE | dst, src | Load External Data to/from Data Memory |
| LDEI | dst, src | Load External Data to/from Data Memory and Auto-Increment Addresses |
| LDX | dst, src | Load using Extended Addressing |
| LEA | dst, X(src) | Load Effective Address |
| POP | dst | Pop |
| POPX | dst | Pop using Extended Addressing |
| PUSH | src | Push |
| PUSHX | src | Push using Extended Addressing |

Table 99. Logical Instructions

| Mnemonic | Operands | Instruction |
|----------|----------|--|
| AND | dst, src | Logical AND |
| ANDX | dst, src | Logical AND using Extended Addressing |
| COM | dst | Complement |
| OR | dst, src | Logical OR |
| ORX | dst, src | Logical OR using Extended Addressing |
| XOR | dst, src | Logical Exclusive OR |
| XORX | dst, src | Logical Exclusive OR using Extended Addressing |



Table 100. Program Control Instructions

| Mnemonic | Operands | Instruction |
|-----------------|-----------------|-------------------------------|
| BRK | — | On-Chip Debugger Break |
| BTJ | p, bit, src, DA | Bit Test and Jump |
| BTJNZ | bit, src, DA | Bit Test and Jump if Non-Zero |
| BTJZ | bit, src, DA | Bit Test and Jump if Zero |
| CALL | dst | Call Procedure |
| DJNZ | dst, src, RA | Decrement and Jump Non-Zero |
| IRET | — | Interrupt Return |
| JP | dst | Jump |
| JP cc | dst | Jump Conditional |
| JR | DA | Jump Relative |
| JR cc | DA | Jump Relative Conditional |
| RET | — | Return |
| TRAP | vector | Software Trap |

Table 101. Rotate and Shift Instructions

| Mnemonic | Operands | Instruction |
|-----------------|-----------------|----------------------------|
| BSWAP | dst | Bit Swap |
| RL | dst | Rotate Left |
| RLC | dst | Rotate Left through Carry |
| RR | dst | Rotate Right |
| RRC | dst | Rotate Right through Carry |
| SRA | dst | Shift Right Arithmetic |
| SRL | dst | Shift Right Logical |
| SWAP | dst | Swap Nibbles |



eZ8 CPU Instruction Summary

Table 102 summarizes the eZ8 CPU instructions. The table identifies the addressing modes employed by the instruction, the effect upon the Flags register, the number of CPU clock cycles required for the instruction fetch, and the number of CPU clock cycles required for the instruction execution.

Table 102. eZ8 CPU Instruction Summary

| Assembly Mnemonic | Symbolic Operation | Address Mode | | Opcode(s) (Hex) | Flags | | | | | | Fetch Cycles | Instr. Cycles |
|-------------------|---|--------------|-----|--------------------|----------------|---|---|---|---|---|--------------|---------------|
| | | dst | src | | C | Z | S | V | D | H | | |
| ADC dst, src | $dst \leftarrow dst + src + C$ | r | r | 12 | * | * | * | * | 0 | * | 2 | 3 |
| | | r | lr | 13 | | | | | | | 2 | 4 |
| | | R | R | 14 | | | | | | | 3 | 3 |
| | | R | IR | 15 | | | | | | | 3 | 4 |
| | | R | IM | 16 | | | | | | | 3 | 3 |
| | | IR | IM | 17 | | | | | | | 3 | 4 |
| ADCX dst, src | $dst \leftarrow dst + src + C$ | ER | ER | 18 | * | * | * | * | 0 | * | 4 | 3 |
| | | ER | IM | 19 | | | | | | | 4 | 3 |
| ADD dst, src | $dst \leftarrow dst + src$ | r | r | 02 | * | * | * | * | 0 | * | 2 | 3 |
| | | r | lr | 03 | | | | | | | 2 | 4 |
| | | R | R | 04 | | | | | | | 3 | 3 |
| | | R | IR | 05 | | | | | | | 3 | 4 |
| | | R | IM | 06 | | | | | | | 3 | 3 |
| | | IR | IM | 07 | | | | | | | 3 | 4 |
| ADDX dst, src | $dst \leftarrow dst + src$ | ER | ER | 08 | * | * | * | * | 0 | * | 4 | 3 |
| | | ER | IM | 09 | | | | | | | 4 | 3 |
| AND dst, src | $dst \leftarrow dst \text{ AND } src$ | r | r | 52 | - | * | * | 0 | - | - | 2 | 3 |
| | | r | lr | 53 | | | | | | | 2 | 4 |
| | | R | R | 54 | | | | | | | 3 | 3 |
| | | R | IR | 55 | | | | | | | 3 | 4 |
| | | R | IM | 56 | | | | | | | 3 | 3 |
| | | IR | IM | 57 | | | | | | | 3 | 4 |
| Flags Notation: | * = Value is a function of the result of the operation. | | | | 0 = Reset to 0 | | | | | | | |
| | - = Unaffected | | | | 1 = Set to 1 | | | | | | | |
| | X = Undefined | | | | | | | | | | | |



Table 102. eZ8 CPU Instruction Summary (Continued)

| Assembly Mnemonic | Symbolic Operation | Address Mode | | Opcode(s) (Hex) | Flags | | | | | | Fetch Cycles | Instr. Cycles |
|----------------------|---|--------------|----------------|--------------------|-------|---|---|---|---|---|--------------|---------------|
| | | dst | src | | C | Z | S | V | D | H | | |
| ANDX dst, src | dst ← dst AND src | ER | ER | 58 | - | * | * | 0 | - | - | 4 | 3 |
| | | ER | IM | 59 | | | | | | | 4 | 3 |
| BCLR bit, dst | dst[bit] ← 0 | r | | E2 | - | * | * | 0 | - | - | 2 | 2 |
| BIT p, bit, dst | dst[bit] ← p | r | | E2 | - | * | * | 0 | - | - | 2 | 2 |
| BRK | Debugger Break | | | 00 | - | - | - | - | - | - | 1 | 1 |
| BSET bit, dst | dst[bit] ← 1 | r | | E2 | - | * | * | 0 | - | - | 2 | 2 |
| BSWAP dst | dst[7:0] ← dst[0:7] | R | | D5 | X | * | * | 0 | - | - | 2 | 2 |
| BTJ p, bit, src, dst | if src[bit] = p PC ← PC + X | | r | F6 | - | - | - | - | - | - | 3 | 3 |
| | | | lr | F7 | | | | | | | 3 | 4 |
| BTJNZ bit, src, dst | if src[bit] = 1 PC ← PC + X | | r | F6 | - | - | - | - | - | - | 3 | 3 |
| | | | lr | F7 | | | | | | | 3 | 4 |
| BTJZ bit, src, dst | if src[bit] = 0 PC ← PC + X | | r | F6 | - | - | - | - | - | - | 3 | 3 |
| | | | lr | F7 | | | | | | | 3 | 4 |
| CALL dst | SP ← SP - 2 @SP ← PC PC ← dst | IRR | | D4 | - | - | - | - | - | - | 2 | 6 |
| | | DA | | D6 | | | | | | | 3 | 3 |
| CCF | C ← ~C | | | EF | * | - | - | - | - | - | 1 | 2 |
| CLR dst | dst ← 00H | R | | B0 | - | - | - | - | - | - | 2 | 2 |
| | | IR | | B1 | | | | | | | 2 | 3 |
| COM dst | dst ← ~dst | R | | 60 | - | * | * | 0 | - | - | 2 | 2 |
| | | IR | | 61 | | | | | | | 2 | 3 |
| CP dst, src | dst - src | r | r | A2 | * | * | * | * | - | - | 2 | 3 |
| | | r | lr | A3 | | | | | | | 2 | 4 |
| | | R | R | A4 | | | | | | | 3 | 3 |
| | | R | IR | A5 | | | | | | | 3 | 4 |
| | | R | IM | A6 | | | | | | | 3 | 3 |
| | | IR | IM | A7 | | | | | | | 3 | 4 |
| Flags Notation: | * = Value is a function of the result of the operation. | | 0 = Reset to 0 | | | | | | | | | |
| | - = Unaffected | | 1 = Set to 1 | | | | | | | | | |
| | X = Undefined | | | | | | | | | | | |



Table 102. eZ8 CPU Instruction Summary (Continued)

| Assembly Mnemonic | Symbolic Operation | Address Mode | | Opcode(s) (Hex) | Flags | | | | | | Fetch Cycles | Instr. Cycles |
|-------------------|---|--------------|-----|--------------------|----------------|---|---|---|---|---|-----------------|------------------|
| | | dst | src | | C | Z | S | V | D | H | | |
| CPC dst, src | dst - src - C | r | r | 1F A2 | * | * | * | * | - | - | 3 | 3 |
| | | r | lr | 1F A3 | | | | | | | 3 | 4 |
| | | R | R | 1F A4 | | | | | | | 4 | 3 |
| | | R | IR | 1F A5 | | | | | | | 4 | 4 |
| | | R | IM | 1F A6 | | | | | | | 4 | 3 |
| | | IR | IM | 1F A7 | | | | | | | 4 | 4 |
| CPCX dst, src | dst - src - C | ER | ER | 1F A8 | * | * | * | * | - | - | 5 | 3 |
| | | ER | IM | 1F A9 | | | | | | | 5 | 3 |
| CPX dst, src | dst - src | ER | ER | A8 | * | * | * | * | - | - | 4 | 3 |
| | | ER | IM | A9 | | | | | | | 4 | 3 |
| DA dst | dst ← DA(dst) | R | | 40 | * | * | * | X | - | - | 2 | 2 |
| | | IR | | 41 | | | | | | | 2 | 3 |
| DEC dst | dst ← dst - 1 | R | | 30 | - | * | * | * | - | - | 2 | 2 |
| | | IR | | 31 | | | | | | | 2 | 3 |
| DECW dst | dst ← dst - 1 | RR | | 80 | - | * | * | * | - | - | 2 | 5 |
| | | IRR | | 81 | | | | | | | 2 | 6 |
| DI | IRQCTL[7] ← 0 | | | 8F | - | - | - | - | - | - | 1 | 2 |
| DJNZ dst, RA | dst ← dst - 1 if dst ≠ 0 PC ← PC + X | r | | 0A-FA | - | - | - | - | - | - | 2 | 3 |
| EI | IRQCTL[7] ← 1 | | | 9F | - | - | - | - | - | - | 1 | 2 |
| HALT | Halt Mode | | | 7F | - | - | - | - | - | - | 1 | 2 |
| INC dst | dst ← dst + 1 | R | | 20 | - | * | * | - | - | - | 2 | 2 |
| | | IR | | 21 | | | | | | | 2 | 3 |
| | | r | | 0E-FE | | | | | | | 1 | 2 |
| INCW dst | dst ← dst + 1 | RR | | A0 | - | * | * | * | - | - | 2 | 5 |
| | | IRR | | A1 | | | | | | | 2 | 6 |
| Flags Notation: | * = Value is a function of the result of the operation. | | | | 0 = Reset to 0 | | | | | | | |
| | - = Unaffected | | | | 1 = Set to 1 | | | | | | | |
| | X = Undefined | | | | | | | | | | | |



Table 102. eZ8 CPU Instruction Summary (Continued)

| Assembly Mnemonic | Symbolic Operation | Address Mode | | Opcode(s) (Hex) | Flags | | | | | | Fetch Cycles | Instr. Cycles |
|-------------------|---|--------------|------|--------------------|----------------|---|---|---|---|--------------|--------------|---------------|
| | | dst | src | | C | Z | S | V | D | H | | |
| IRET | $FLAGS \leftarrow @SP$ $SP \leftarrow SP + 1$ $PC \leftarrow @SP$ $SP \leftarrow SP + 2$ $IRQCTL[7] \leftarrow 1$ | | | BF | * | * | * | * | * | * | 1 | 5 |
| JP dst | $PC \leftarrow dst$ | DA | | 8D | - | - | - | - | - | - | 3 | 2 |
| | | IRR | | C4 | | | | | | | 2 | 3 |
| JP cc, dst | if cc is true $PC \leftarrow dst$ | DA | | 0D-FD | - | - | - | - | - | - | 3 | 2 |
| JR dst | $PC \leftarrow PC + X$ | DA | | 8B | - | - | - | - | - | - | 2 | 2 |
| JR cc, dst | if cc is true $PC \leftarrow PC + X$ | DA | | 0B-FB | - | - | - | - | - | - | 2 | 2 |
| LD dst, rc | $dst \leftarrow src$ | r | IM | 0C-FC | - | - | - | - | - | - | 2 | 2 |
| | | r | X(r) | C7 | | | | | | | 3 | 3 |
| | | X(r) | r | D7 | | | | | | | 3 | 4 |
| | | r | lr | E3 | | | | | | | 2 | 3 |
| | | R | R | E4 | | | | | | | 3 | 2 |
| | | R | IR | E5 | | | | | | | 3 | 4 |
| | | R | IM | E6 | | | | | | | 3 | 2 |
| | | IR | IM | E7 | | | | | | | 3 | 3 |
| | | lr | r | F3 | | | | | | | 2 | 3 |
| | | IR | R | F5 | | | | | | | 3 | 3 |
| LDC dst, src | $dst \leftarrow src$ | r | lrr | C2 | - | - | - | - | - | - | 2 | 5 |
| | | lr | lrr | C5 | | | | | | | 2 | 9 |
| | | lrr | r | D2 | | | | | | | 2 | 5 |
| LDCI dst, src | $dst \leftarrow src$ $r \leftarrow r + 1$ $rr \leftarrow rr + 1$ | lr | lrr | C3 | - | - | - | - | - | - | 2 | 9 |
| | | lrr | lr | D3 | | | | | | | 2 | 9 |
| LDE dst, src | $dst \leftarrow src$ | r | lrr | 82 | - | - | - | - | - | - | 2 | 5 |
| | | lrr | r | 92 | | | | | | | 2 | 5 |
| Flags Notation: | * = Value is a function of the result of the operation. - = Unaffected X = Undefined | | | | 0 = Reset to 0 | | | | | 1 = Set to 1 | | |



Table 102. eZ8 CPU Instruction Summary (Continued)

| Assembly Mnemonic | Symbolic Operation | Address Mode | | Opcode(s) (Hex) | Flags | | | | | | Fetch Cycles | Instr. Cycles |
|-------------------|---|--------------|----------------|--------------------|-------|---|---|---|---|---|--------------|---------------|
| | | dst | src | | C | Z | S | V | D | H | | |
| LDEI dst, src | dst ← src | lr | lrr | 83 | - | - | - | - | - | - | 2 | 9 |
| | r ← r + 1 rr ← rr + 1 | lrr | lr | 93 | | | | | | | 2 | 9 |
| LDX dst, src | dst ← src | r | ER | 84 | - | - | - | - | - | - | 3 | 2 |
| | | lr | ER | 85 | | | | | | | 3 | 3 |
| | | R | IRR | 86 | | | | | | | 3 | 4 |
| | | IR | IRR | 87 | | | | | | | 3 | 5 |
| | | r | X(rr) | 88 | | | | | | | 3 | 4 |
| | | X(rr) | r | 89 | | | | | | | 3 | 4 |
| | | ER | r | 94 | | | | | | | 3 | 2 |
| | | ER | lr | 95 | | | | | | | 3 | 3 |
| | | IRR | R | 96 | | | | | | | 3 | 4 |
| | | IRR | IR | 97 | | | | | | | 3 | 5 |
| | | ER | ER | E8 | | | | | | | 4 | 2 |
| ER | IM | E9 | | | | | | | 4 | 2 | | |
| LEA dst, X(src) | dst ← src + X | r | X(r) | 98 | - | - | - | - | - | - | 3 | 3 |
| | | rr | X(rr) | 99 | | | | | | | 3 | 5 |
| MULT dst | dst[15:0] ← dst[15:8] * dst[7:0] | RR | | F4 | - | - | - | - | - | - | 2 | 8 |
| NOP | No operation | | | 0F | - | - | - | - | - | - | 1 | 2 |
| OR dst, src | dst ← dst OR src | r | r | 42 | - | * | * | 0 | - | - | 2 | 3 |
| | | r | lr | 43 | | | | | | | 2 | 4 |
| | | R | R | 44 | | | | | | | 3 | 3 |
| | | R | IR | 45 | | | | | | | 3 | 4 |
| | | R | IM | 46 | | | | | | | 3 | 3 |
| | | IR | IM | 47 | | | | | | | 3 | 4 |
| ORX dst, src | dst ← dst OR src | ER | ER | 48 | - | * | * | 0 | - | - | 4 | 3 |
| | | ER | IM | 49 | | | | | | | 4 | 3 |
| Flags Notation: | * = Value is a function of the result of the operation. | | 0 = Reset to 0 | | | | | | | | | |
| | - = Unaffected | | 1 = Set to 1 | | | | | | | | | |
| | X = Undefined | | | | | | | | | | | |



Table 102. eZ8 CPU Instruction Summary (Continued)

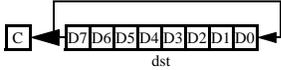
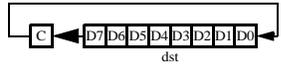
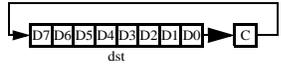
| Assembly Mnemonic | Symbolic Operation | Address Mode | | Opcode(s) (Hex) | Flags | | | | | | Fetch Cycles | Instr. Cycles |
|-------------------|---|--------------|----------------|--------------------|-------|---|---|---|---|---|--------------|---------------|
| | | dst | src | | C | Z | S | V | D | H | | |
| POP dst | dst ← @SP SP ← SP + 1 | R | | 50 | - | - | - | - | - | - | 2 | 2 |
| | | IR | | 51 | | | | | | | 2 | 3 |
| POPX dst | dst ← @SP SP ← SP + 1 | ER | | D8 | - | - | - | - | - | - | 3 | 2 |
| | | | | | | | | | | | | |
| PUSH src | SP ← SP - 1 @SP ← src | R | | 70 | - | - | - | - | - | - | 2 | 2 |
| | | IR | | 71 | | | | | | | 2 | 3 |
| PUSHX src | SP ← SP - 1 @SP ← src | ER | | C8 | - | - | - | - | - | - | 3 | 2 |
| | | | | | | | | | | | | |
| RCF | C ← 0 | | | CF | 0 | - | - | - | - | - | 1 | 2 |
| RET | PC ← @SP SP ← SP + 2 | | | AF | - | - | - | - | - | - | 1 | 4 |
| RL dst |  | R | | 90 | * | * | * | * | - | - | 2 | 2 |
| | | IR | | 91 | | | | | | | 2 | 3 |
| RLC dst |  | R | | 10 | * | * | * | * | - | - | 2 | 2 |
| | | IR | | 11 | | | | | | | 2 | 3 |
| RR dst |  | R | | E0 | * | * | * | * | - | - | 2 | 2 |
| | | IR | | E1 | | | | | | | 2 | 3 |
| RRC dst |  | R | | C0 | * | * | * | * | - | - | 2 | 2 |
| | | IR | | C1 | | | | | | | 2 | 3 |
| SBC dst, src | dst ← dst - src - C | r | r | 32 | * | * | * | * | 1 | * | 2 | 3 |
| | | r | lr | 33 | | | | | | | 2 | 4 |
| | | R | R | 34 | | | | | | | 3 | 3 |
| | | R | IR | 35 | | | | | | | 3 | 4 |
| | | R | IM | 36 | | | | | | | 3 | 3 |
| | | IR | IM | 37 | | | | | | | 3 | 4 |
| Flags Notation: | * = Value is a function of the result of the operation. | | 0 = Reset to 0 | | | | | | | | | |
| | - = Unaffected | | 1 = Set to 1 | | | | | | | | | |
| | X = Undefined | | | | | | | | | | | |



Table 102. eZ8 CPU Instruction Summary (Continued)

| Assembly Mnemonic | Symbolic Operation | Address Mode | | Opcode(s) (Hex) | Flags | | | | | | Fetch Cycles | Instr. Cycles |
|-------------------|---------------------|--------------|-----|--------------------|-------|---|---|---|---|---|--------------|---------------|
| | | dst | src | | C | Z | S | V | D | H | | |
| SBCX dst, src | dst ← dst – src - C | ER | ER | 38 | * | * | * | * | 1 | * | 4 | 3 |
| | | ER | IM | 39 | | | | | | | 4 | 3 |
| SCF | C ← 1 | | | DF | 1 | - | - | - | - | - | 1 | 2 |
| SRA dst | | R | | D0 | * | * | * | 0 | - | - | 2 | 2 |
| | | IR | | D1 | | | | | | | 2 | 3 |
| SRL dst | | R | | 1F C0 | * | * | 0 | * | - | - | 3 | 2 |
| | | IR | | 1F C1 | | | | | | | 3 | 3 |
| SRP src | RP ← src | | IM | 01 | - | - | - | - | - | - | 2 | 2 |
| STOP | STOP Mode | | | 6F | - | - | - | - | - | - | 1 | 2 |
| SUB dst, src | dst ← dst – src | r | r | 22 | * | * | * | * | 1 | * | 2 | 3 |
| | | r | lr | 23 | | | | | | | 2 | 4 |
| | | R | R | 24 | | | | | | | 3 | 3 |
| | | R | IR | 25 | | | | | | | 3 | 4 |
| | | R | IM | 26 | | | | | | | 3 | 3 |
| | | IR | IM | 27 | | | | | | | 3 | 4 |
| SUBX dst, src | dst ← dst – src | ER | ER | 28 | * | * | * | * | 1 | * | 4 | 3 |
| | | ER | IM | 29 | | | | | | | 4 | 3 |
| SWAP dst | dst[7:4] ↔ dst[3:0] | R | | F0 | X | * | * | X | - | - | 2 | 2 |
| | | IR | | F1 | | | | | | | 2 | 3 |
| TCM dst, src | (NOT dst) AND src | r | r | 62 | - | * | * | 0 | - | - | 2 | 3 |
| | | r | lr | 63 | | | | | | | 2 | 4 |
| | | R | R | 64 | | | | | | | 3 | 3 |
| | | R | IR | 65 | | | | | | | 3 | 4 |
| | | R | IM | 66 | | | | | | | 3 | 3 |
| | | IR | IM | 67 | | | | | | | 3 | 4 |
| TCMX dst, src | (NOT dst) AND src | ER | ER | 68 | - | * | * | 0 | - | - | 4 | 3 |
| | | ER | IM | 69 | | | | | | | 4 | 3 |

Flags Notation: * = Value is a function of the result of the operation. 0 = Reset to 0
 - = Unaffected 1 = Set to 1
 X = Undefined



Table 102. eZ8 CPU Instruction Summary (Continued)

| Assembly Mnemonic | Symbolic Operation | Address Mode | | Opcode(s) (Hex) | Flags | | | | | | Fetch Cycles | Instr. Cycles |
|-------------------|--|--------------|--------|--------------------|--------------------------------|---|---|---|---|---|-----------------|------------------|
| | | dst | src | | C | Z | S | V | D | H | | |
| TM dst, src | dst AND src | r | r | 72 | - | * | * | 0 | - | - | 2 | 3 |
| | | r | lr | 73 | | | | | | | 2 | 4 |
| | | R | R | 74 | | | | | | | 3 | 3 |
| | | R | IR | 75 | | | | | | | 3 | 4 |
| | | R | IM | 76 | | | | | | | 3 | 3 |
| | | IR | IM | 77 | | | | | | | 3 | 4 |
| TMX dst, src | dst AND src | ER | ER | 78 | - | * | * | 0 | - | - | 4 | 3 |
| | | ER | IM | 79 | | | | | | | 4 | 3 |
| TRAP Vector | SP ← SP - 2 @SP ← PC SP ← SP - 1 @SP ← FLAGS PC ← @Vector | | Vector | F2 | - | - | - | - | - | - | 2 | 6 |
| WDT | | | | 5F | - | - | - | - | - | - | 1 | 2 |
| XOR dst, src | dst ← dst XOR src | r | r | B2 | - | * | * | 0 | - | - | 2 | 3 |
| | | r | lr | B3 | | | | | | | 2 | 4 |
| | | R | R | B4 | | | | | | | 3 | 3 |
| | | R | IR | B5 | | | | | | | 3 | 4 |
| | | R | IM | B6 | | | | | | | 3 | 3 |
| | | IR | IM | B7 | | | | | | | 3 | 4 |
| XORX dst, src | dst ← dst XOR src | ER | ER | B8 | - | * | * | 0 | - | - | 4 | 3 |
| | | ER | IM | B9 | | | | | | | 4 | 3 |
| Flags Notation: | * = Value is a function of the result of the operation. - = Unaffected X = Undefined | | | | 0 = Reset to 0 1 = Set to 1 | | | | | | | |

Opcode Maps

A description of the opcode map data and the abbreviations are provided in Figure 26. Figures 27 and 32 provide information about each of the eZ8 CPU instructions. Table 103 lists Opcode Map abbreviations.

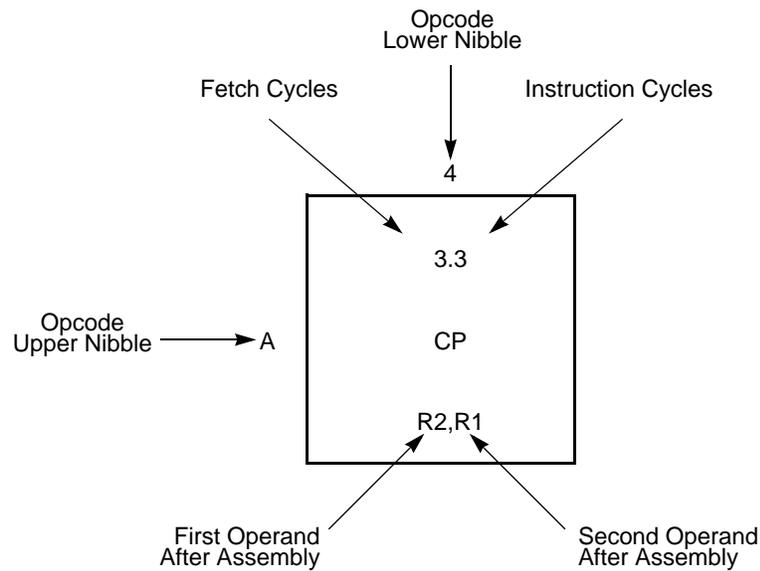


Figure 26.Opcode Map Cell Description



Table 103. Opcode Map Abbreviations

| Abbreviation | Description | Abbreviation | Description |
|---------------------|------------------------------------|---|------------------------|
| b | Bit position | IRR | Indirect Register Pair |
| cc | Condition code | p | Polarity (0 or 1) |
| X | 8-bit signed index or displacement | r | 4-bit Working Register |
| DA | Destination address | R | 8-bit register |
| ER | Extended Addressing register | r1, R1, Ir1, Irr1, IR1, rr1, RR1, IRR1, ER1 | Destination address |
| IM | Immediate data value | r2, R2, Ir2, Irr2, IR2, rr2, RR2, IRR2, ER2 | Source address |
| Ir | Indirect Working Register | RA | Relative |
| IR | Indirect register | rr | Working Register Pair |
| Irr | Indirect Working Register Pair | RR | Register Pair |



| | | Lower Nibble (Hex) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|--------------------|---|--------------------|---------------------|-----------------------|-------------------------|----------------------|------------------------|------------------------|--------------------------|------------------------|-------------------------|---------------------|-------------------|--------------------|--------------------|------------------|------------|--------------------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|------------|---|---|---|---|---|-------------|---|---|---|---|---|-------------|---|---|---|---|---|-----------|---|---|---|---|---|-----------|---|---|---|---|---|------------|---|---|---|---|---|-------------|---|---|---|---|---|------------|---|---|---|---|---|------------|---|---|---|---|---|------------|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Upper Nibble (Hex) | 0 | 1.1 BRK | 2.2 SRP IM | 2.3 ADD r1,r2 | 2.4 ADD r1,lr2 | 3.3 ADD R2,R1 | 3.4 ADD IR2,R1 | 3.3 ADD R1,IM | 3.4 ADD IR1,IM | 4.3 ADDX ER2,ER1 | 4.3 ADDX IM,ER1 | 2.3 DJNZ r1,X | 2.2 JR cc,X | 2.2 LD r1,IM | 3.2 JP cc,DA | 1.2 INC r1 | 1.2 NOP | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | 1 | 2.2 RLC R1 | 2.3 RLC IR1 | 2.3 ADC r1,r2 | 2.4 ADC r1,lr2 | 3.3 ADC R2,R1 | 3.4 ADC IR2,R1 | 3.3 ADC R1,IM | 3.4 ADC IR1,IM | 4.3 ADCX ER2,ER1 | 4.3 ADCX IM,ER1 | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | See 2nd Opcode Map | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | 2 | 2.2 INC R1 | 2.3 INC IR1 | 2.3 SUB r1,r2 | 2.4 SUB r1,lr2 | 3.3 SUB R2,R1 | 3.4 SUB IR2,R1 | 3.3 SUB R1,IM | 3.4 SUB IR1,IM | 4.3 SUBX ER2,ER1 | 4.3 SUBX IM,ER1 | | | | | | | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | 3 | 2.2 DEC R1 | 2.3 DEC IR1 | 2.3 SBC r1,r2 | 2.4 SBC r1,lr2 | 3.3 SBC R2,R1 | 3.4 SBC IR2,R1 | 3.3 SBC R1,IM | 3.4 SBC IR1,IM | 4.3 SBCX ER2,ER1 | 4.3 SBCX IM,ER1 | | | | | | | | | | | | | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | 4 | 2.2 DA R1 | 2.3 DA IR1 | 2.3 OR r1,r2 | 2.4 OR r1,lr2 | 3.3 OR R2,R1 | 3.4 OR IR2,R1 | 3.3 OR R1,IM | 3.4 OR IR1,IM | 4.3 ORX ER2,ER1 | 4.3 ORX IM,ER1 | | | | | | | | | | | | | | | | | | | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | 5 | 2.2 POP R1 | 2.3 POP IR1 | 2.3 AND r1,r2 | 2.4 AND r1,lr2 | 3.3 AND R2,R1 | 3.4 AND IR2,R1 | 3.3 AND R1,IM | 3.4 AND IR1,IM | 4.3 ANDX ER2,ER1 | 4.3 ANDX IM,ER1 | | | | | | | | | | | | | | | | | | | | | | | | | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | 1.2 WDT | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | 6 | 2.2 COM R1 | 2.3 COM IR1 | 2.3 TCM r1,r2 | 2.4 TCM r1,lr2 | 3.3 TCM R2,R1 | 3.4 TCM IR2,R1 | 3.3 TCM R1,IM | 3.4 TCM IR1,IM | 4.3 TCMX ER2,ER1 | 4.3 TCMX IM,ER1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | 1.2 STOP | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | 7 | 2.2 PUSH R2 | 2.3 PUSH IR2 | 2.3 TM r1,r2 | 2.4 TM r1,lr2 | 3.3 TM R2,R1 | 3.4 TM IR2,R1 | 3.3 TM R1,IM | 3.4 TM IR1,IM | 4.3 TMX ER2,ER1 | 4.3 TMX IM,ER1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | 1.2 HALT | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | 8 | 2.5 DECW RR1 | 2.6 DECW IRR1 | 2.5 LDE r1,lrr2 | 2.9 LDEI lr1,lrr2 | 3.2 LDX r1,ER2 | 3.3 LDX lr1,ER2 | 3.4 LDX IRR2,R1 | 3.5 LDX IRR2,IR1 | 3.4 LDX r1,rr2,X | 3.4 LDX rr1,rr2,X | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | 1.2 DI | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | 9 | 2.2 RL R1 | 2.3 RL IR1 | 2.5 LDE r2,lrr1 | 2.9 LDEI lr2,lrr1 | 3.2 LDX r2,ER1 | 3.3 LDX lr2,ER1 | 3.4 LDX R2,IRR1 | 3.5 LDX IR2,IRR1 | 3.3 LEA r1,r2,X | 3.5 LEA rr1,rr2,X | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | 1.2 EI | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | A | 2.5 INCW RR1 | 2.6 INCW IRR1 | 2.3 CP r1,r2 | 2.4 CP r1,lr2 | 3.3 CP R2,R1 | 3.4 CP IR2,R1 | 3.3 CP R1,IM | 3.4 CP IR1,IM | 4.3 CPX ER2,ER1 | 4.3 CPX IM,ER1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | 1.4 RET | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | B | 2.2 CLR R1 | 2.3 CLR IR1 | 2.3 XOR r1,r2 | 2.4 XOR r1,lr2 | 3.3 XOR R2,R1 | 3.4 XOR IR2,R1 | 3.3 XOR R1,IM | 3.4 XOR IR1,IM | 4.3 XORX ER2,ER1 | 4.3 XORX IM,ER1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | 1.5 IRET | | | | | | | | | | | | | | | | | | | | | | | |
| | C | 2.2 RRC R1 | 2.3 RRC IR1 | 2.5 LDC r1,lrr2 | 2.9 LDCI lr1,lrr2 | 2.3 JP IRR1 | 2.9 LDC lr1,lrr2 | | 3.4 LD r1,r2,X | 3.2 PUSHX ER2 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | 1.2 RCF | | | | | | | | | | | | | | | | | |
| | D | 2.2 SRA R1 | 2.3 SRA IR1 | 2.5 LDC r2,lrr1 | 2.9 LDCI lr2,lrr1 | 2.6 CALL IRR1 | 2.2 BSWAP R1 | 3.3 CALL DA | 3.4 LD r2,r1,X | 3.2 POPX ER1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | 1.2 SCF | | | | | | | | | | | |
| | E | 2.2 RR R1 | 2.3 RR IR1 | 2.2 BIT p,b,r1 | 2.3 LD r1,lr2 | 3.2 LD R2,R1 | 2.9 LD IR2,R1 | 3.3 LD R1,IM | 3.3 LD IR1,IM | 4.2 LDX ER2,ER1 | 4.2 LDX IM,ER1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ | 1.2 CCF | | | | | |
| | F | 2.2 SWAP R1 | 2.3 SWAP IR1 | 2.6 TRAP Vector | 2.3 LD lr1,r2 | 2.8 MULT RR1 | 3.3 LD R2,IR1 | 3.3 BTJ p,b,r1,X | 3.4 BTJ p,b,lrr1,X | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | ↓ | ↓ | ↓ | ↓ | ↓ | ↓ |

Figure 27. First Opcode Map



| | | Lower Nibble (Hex) | | | | | | | | | | | | | | | |
|--------------------|---|-------------------------|--------------------------|----------------------------|----------------------------|----------------------------|-----------------------------|----------------------------|-----------------------------|-------------------------------|------------------------------|---|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| Upper Nibble (Hex) | 0 | | | | | | | | | | | | | | | | |
| | 1 | | | | | | | | | | | | | | | | |
| | 2 | | | | | | | | | | | | | | | | |
| | 3 | | | | | | | | | | | | | | | | |
| | 4 | | | | | | | | | | | | | | | | |
| | 5 | | | | | | | | | | | | | | | | |
| | 6 | | | | | | | | | | | | | | | | |
| | 7 | | | | | | | | | | | | | | | | |
| | 8 | | | | | | | | | | | | | | | | |
| | 9 | | | | | | | | | | | | | | | | |
| | A | | | 3.3 CPC r1,r2 | 3.4 CPC r1,r2 | 4.3 CPC R2,R1 | 4.4 CPC IR2,R1 | 4.3 CPC R1,IM | 4.4 CPC IR1,IM | 5.3 CPCX ER2,ER1 | 5.3 CPCX IM,ER1 | | | | | | |
| | B | | | | | | | | | | | | | | | | |
| | C | 3.2 SRL R1 | 3.3 SRL IR1 | | | | | | | | | | | | | | |
| | D | | | | | | | | | | | | | | | | |
| | E | | | | | | | | | | | | | | | | |
| | F | | | | | | | | | | | | | | | | |

Figure 28. Second Opcode Map after 1FH

Electrical Characteristics

The data in this chapter is pre-qualification and pre-characterization and is subject to change. Additional electrical characteristics may be found in the individual chapters.

Absolute Maximum Ratings

Stresses greater than those listed in Table 104 may cause permanent damage to the device. These ratings are stress ratings only. Operation of the device at any condition outside those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability. For improved reliability, tie unused inputs to one of the supply voltages (V_{DD} or V_{SS}).

Table 104. Absolute Maximum Ratings

| Parameter | Minimum | Maximum | Units | Notes |
|--|---------|---------|-------|-------|
| Ambient temperature under bias | 0 | +105 | °C | 1 |
| Storage temperature | -65 | +150 | °C | |
| Voltage on any pin with respect to V_{SS} | -0.3 | +5.5 | V | 2 |
| Voltage on V_{DD} pin with respect to V_{SS} | -0.3 | +3.6 | V | |
| Maximum current on input and/or inactive output pin | -5 | +5 | μA | |
| Maximum output current from active output pin | -25 | +25 | mA | |
| 8-pin Packages Maximum Ratings at 0°C to 70°C | | | | |
| Total power dissipation | | 220 | mW | |
| Maximum current into V_{DD} or out of V_{SS} | | 60 | mA | |
| 20-pin Packages Maximum Ratings at 0°C to 70°C | | | | |
| Total power dissipation | | 430 | mW | |
| Maximum current into V_{DD} or out of V_{SS} | | 120 | mA | |
| 28-pin Packages Maximum Ratings at 0°C to 70°C | | | | |
| Total power dissipation | | 450 | mW | |
| Maximum current into V_{DD} or out of V_{SS} | | 125 | mA | |
| Operating temperature is specified in DC Characteristics | | | | |
| 1. This voltage applies to all pins except the following: V_{DD} , AV_{DD} , and pins supporting analog input (Port B[5:0], Port C[2:0]). On the 8-pin packages, this applies to all pins but V_{DD} . | | | | |



DC Characteristics

Table 105 lists the DC characteristics of the 4K Series products. All voltages are referenced to V_{SS} , the primary system ground.

Table 105. DC Characteristics

| Symbol | Parameter | $T_A = -40^{\circ}\text{C to }+105^{\circ}\text{C}$ | | | Units | Conditions |
|------------|---------------------------|---|---------|--------------------|---------------|---|
| | | Minimum | Typical | Maximum | | |
| V_{DD} | Supply Voltage | 2.7 | – | 3.6 | V | |
| V_{IL1} | Low Level Input Voltage | -0.3 | – | $0.3 \cdot V_{DD}$ | V | For all input pins except $\overline{\text{RESET}}$. |
| V_{IL2} | Low Level Input Voltage | -0.3 | – | 0.8 | V | For $\overline{\text{RESET}}$. |
| V_{IH1} | High Level Input Voltage | 2.0 | – | 5.5 | V | For all input pins without analog or oscillator function. For all signal pins on the 8-pin devices. |
| V_{IH2} | High Level Input Voltage | 2.0 | – | $V_{DD}+0.3$ | V | For those pins with analog or oscillator function. |
| V_{OL1} | Low Level Output Voltage | – | – | 0.4 | V | $I_{OL} = 2\text{mA}$; $V_{DD} = 3.0\text{V}$ High Output Drive disabled. |
| V_{OH1} | High Level Output Voltage | 2.4 | – | – | V | $I_{OH} = -2\text{mA}$; $V_{DD} = 3.0\text{V}$ High Output Drive disabled. |
| V_{OL2} | Low Level Output Voltage | – | – | 0.6 | V | $I_{OL} = 20\text{mA}$; $V_{DD} = 3.3\text{V}$ High Output Drive enabled. |
| V_{OH2} | High Level Output Voltage | 2.4 | – | – | V | $I_{OH} = -20\text{mA}$; $V_{DD} = 3.3\text{V}$ High Output Drive enabled. |
| I_{IL} | Input Leakage Current | -5 | – | +5 | μA | $V_{DD} = 3.6\text{V}$; $V_{IN} = V_{DD}$ or V_{SS}^1 |
| I_{TL} | Tristate Leakage Current | -5 | – | +5 | μA | $V_{DD} = 3.6\text{V}$ |
| C_{PAD} | GPIO Port Pad Capacitance | – | 8.0^2 | – | pF | TBD |
| C_{XIN} | XIN Pad Capacitance | – | 8.0^2 | – | pF | TBD |
| C_{XOUT} | XOUT Pad Capacitance | – | 9.5^2 | – | pF | TBD |
| I_{PU} | Weak Pull-up Current | 30 | 100 | 350 | μA | $V_{DD} = 3.0 - 3.6\text{V}$ |

Table 105. DC Characteristics (Continued)

| Symbol | Parameter | T _A = -40°C to +105°C | | | Units | Conditions |
|--------|-----------------------------|----------------------------------|---------|---------|-------|-----------------------------|
| | | Minimum | Typical | Maximum | | |
| ICCH | Supply Current in Halt Mode | | TBD | | μA | TBD |
| ICCS | Supply Current in STOP Mode | | 2 | | μA | With watchdog timer running |

¹ This condition excludes all pins that have on-chip pull-ups, when driven Low.
² These values are provided for design guidance only and are not tested in production.

Figure 29 illustrates the typical current consumption while operating at 25°C, 3.3V, versus the system clock frequency.

TBD

Figure 29. ICC Versus System Clock Frequency



AC Characteristics

The section provides information about the AC characteristics and timing. All AC timing information assumes a standard load of 50pF on all outputs.

Table 106. AC Characteristics

| Symbol | Parameter | V _{DD} = 2.7 to 3.6V T _A = -40°C to +105°C | | Units | Conditions |
|---------------------|---|---|---------|-------|---|
| | | Minimum | Maximum | | |
| F _{SYSCLK} | System Clock Frequency | – | 20.0 | MHz | Read-only from Flash memory |
| | | 0.032768 | 20.0 | MHz | Program or erasure of the Flash memory |
| F _{IPO} | Internal Precision Oscillator Frequency | 0.032768 | 5.5296 | MHz | Oscillator is not adjustable over the entire range. User may select Min or Max value only. |
| F _{IPO} | Internal Precision Oscillator Frequency | 4.8 | 5.2 | MHz | High speed with trimming |
| F _{IPO} | Internal Precision Oscillator Frequency | 3.75 | 6.25 | MHz | High speed without trimming |
| F _{IPO} | Internal Precision Oscillator Frequency | 30.7 | 33.3 | KHz | Low speed with trimming |
| F _{IPO} | Internal Precision Oscillator Frequency | 24 | 40 | KHz | Low speed without trimming |
| T _{XIN} | System Clock Period | 50 | – | ns | T _{CLK} = 1/F _{sysclk} |
| T _{XINH} | System Clock High Time | 20 | 30 | ns | T _{CLK} = 50ns |
| T _{XINL} | System Clock Low Time | 20 | 30 | ns | T _{CLK} = 50ns |
| T _{XINR} | System Clock Rise Time | – | 3 | ns | T _{CLK} = 50ns |
| T _{XINF} | System Clock Fall Time | – | 3 | ns | T _{CLK} = 50ns |



On-Chip Peripheral AC and DC Electrical Characteristics

Table 107. Power-On Reset and Voltage Brown-Out Electrical Characteristics and Timing

| Symbol | Parameter | $T_A = -40^{\circ}\text{C to } +105^{\circ}\text{C}$ | | | Units | Conditions |
|-------------------|---|--|----------------------|---------|---------------|--|
| | | Minimum | Typical ¹ | Maximum | | |
| V_{POR} | Power-On Reset Voltage Threshold | 2.20 | 2.45 | 2.70 | V | $V_{\text{DD}} = V_{\text{POR}}$ (default VBO trim) |
| V_{VBO} | Voltage Brown-Out Reset Voltage Threshold | 2.15 | 2.40 | 2.65 | V | $V_{\text{DD}} = V_{\text{VBO}}$ (default VBO trim) |
| | V_{POR} to V_{VBO} hysteresis | | 50 | 75 | mV | |
| | Starting V_{DD} voltage to ensure valid Power-On Reset. | – | V_{SS} | – | V | |
| T_{ANA} | Power-On Reset Analog Delay | – | 50 | – | μs | $V_{\text{DD}} > V_{\text{POR}}$; T_{POR} Digital Reset delay follows T_{ANA} |
| T_{POR} | Power-On Reset Digital Delay | TBD | 13 | TBD | μs | 43 Internal Precision Oscillator cycles |
| T_{POR} | Power-On Reset Digital Delay | TBD | 8 | TBD | ms | 5000 Internal Precision Oscillator cycles |
| T_{VBO} | Voltage Brown-Out Pulse Rejection Period | – | 10 | – | μs | $V_{\text{DD}} < V_{\text{VBO}}$ to generate a Reset. |
| T_{RAMP} | Time for V_{DD} to transition from V_{SS} to V_{POR} to ensure valid Reset | 0.10 | – | 100 | ms | |

1 Data in the typical column is from characterization at 3.3V and 0°C. These values are provided for design guidance only and are not tested in production.



Table 108. Flash Memory Electrical Characteristics and Timing

| Parameter | V _{DD} = 2.7 to 3.6V T _A = -40°C to +105°C | | | Units | Notes |
|--|---|---------|---------|--------|--|
| | Minimum | Typical | Maximum | | |
| Flash Byte Read Time | 100 | – | – | ns | |
| Flash Byte Program Time | 20 | – | 40 | μs | |
| Flash Page Erase Time | 10 | – | – | ms | |
| Flash Mass Erase Time | 200 | – | – | ms | |
| Writes to Single Address Before Next Erase | – | – | 2 | | |
| Flash Row Program Time | – | – | 8 | ms | Cumulative program time for single row cannot exceed limit before next erase. This parameter is only an issue when bypassing the Flash Controller. |
| Data Retention | 100 | – | – | years | 25°C |
| Endurance | 20,000 | – | – | cycles | Program / erase cycles |

Table 109. Watch-Dog Timer Electrical Characteristics and Timing

| Symbol | Parameter | V _{DD} = 2.7 - 3.6V T _A = -40°C to +105°C | | | Units | Conditions |
|------------------|--------------------------|--|---------|---------|-------|------------|
| | | Minimum | Typical | Maximum | | |
| F _{WDT} | WDT Oscillator Frequency | | 10 | | KHz | |



Table 110. Analog-to-Digital Converter Electrical Characteristics and Timing

| Symbol | Parameter | V _{DD} = 2.7 to 3.6V T _A = -40°C to +105°C | | | Units | Conditions |
|------------------|---------------------------------|---|-------------------|-------------------|--------|---|
| | | Minimum | Typical | Maximum | | |
| | Resolution | – | 10 | – | bits | External V _{REF} = 3.0V; R _S ← 3.0KΩ |
| | Differential Nonlinearity (DNL) | -1.0 | – | 1.0 | LSB | External V _{REF} = 3.0V; R _S ← 3.0KΩ |
| | Integral Nonlinearity (INL) | -3.0 | – | 3.0 | LSB | External V _{REF} = 3.0V; R _S ← 3.0KΩ |
| | DC Offset Error (single-ended) | -35 | – | 25 | mV | Unbuffered Mode |
| V _{REF} | Internal Reference Voltage | 0.9 1.8 2.0 | 1.0 2.0 2.2 | 1.1 2.2 2.4 | V | REFSEL=00 REFSEL=01 REFSEL=10 |
| | Single-Shot Conversion Time | – | 5129 | – | cycles | System clock cycles |
| | Continuous Conversion Time | – | 256 | – | cycles | System clock cycles |
| | Sampling Rate | System Clock /256 | | | Hz | |
| | Signal Input Bandwidth | – | 10 | | KHz | As defined by -3dB point |
| R _S | Analog Source Impedance | – | – | 10 | kΩ | In unbuffered mode |
| Z _{in} | Input Impedance | TBD | 150 | | kΩ | In unbuffered mode |
| V _{in} | Input Voltage Range | 0 | | V _{DD} | V | Unbuffered Mode |

¹ Analog source impedance affects the ADC offset voltage (because of pin leakage) and input settling time.

General Purpose I/O Port Input Data Sample Timing

Figure 30 illustrates timing of the GPIO Port input sampling. The input value on a GPIO Port pin is sampled on the rising edge of the system clock. The Port value is available to the eZ8 CPU on the second rising clock edge following the change of the Port value.

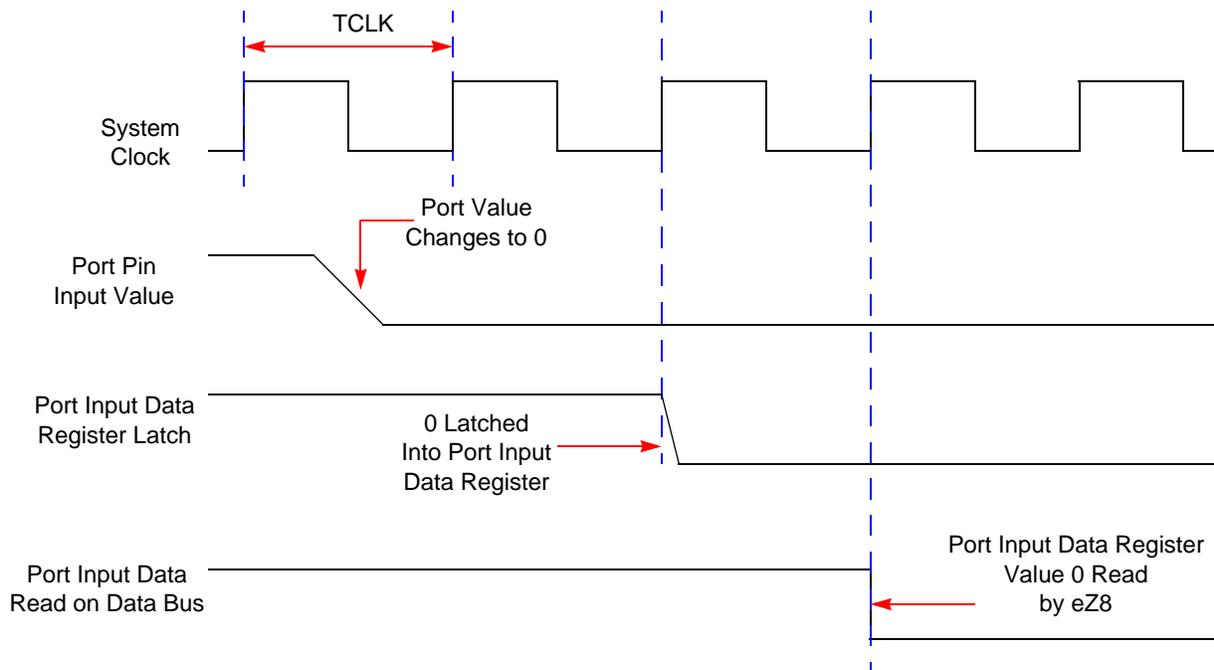


Figure 30. Port Input Sample Timing

Table 111. GPIO Port Input Timing

| Parameter | Abbreviation | Delay (ns) | |
|---------------|---|------------|---------|
| | | Minimum | Maximum |
| T_{S_PORT} | Port Input Transition to XIN Rise Setup Time (Not pictured) | 5 | – |
| T_{H_PORT} | XIN Rise to Port Input Transition Hold Time (Not pictured) | 0 | – |
| T_{SMR} | GPIO Port Pin Pulse Width to ensure STOP Mode Recovery (for GPIO Port Pins enabled as SMR sources) | 1 μ s | |

General Purpose I/O Port Output Timing

Figure 31 and Table 112 provide timing information for GPIO Port pins.

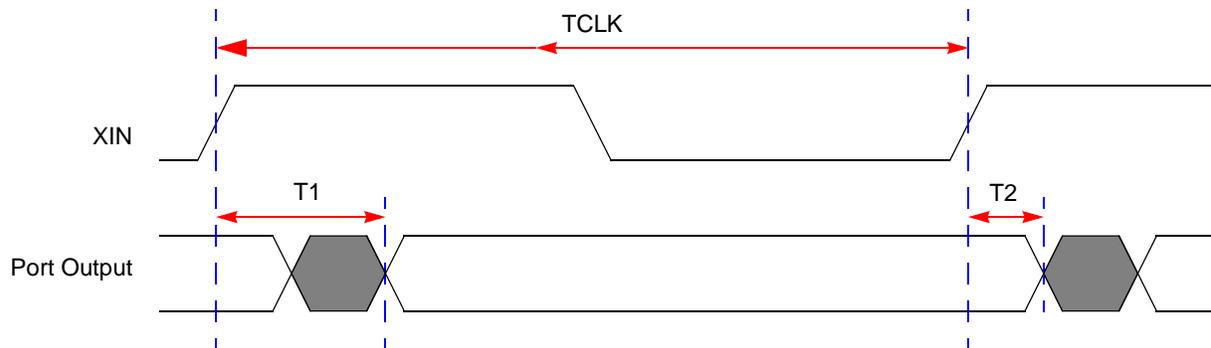


Figure 31.GPIO Port Output Timing

Table 112. GPIO Port Output Timing

| Parameter | Abbreviation | Delay (ns) | |
|-----------------------|-------------------------------------|------------|---------|
| | | Minimum | Maximum |
| GPIO Port pins | | | |
| T ₁ | XIN Rise to Port Output Valid Delay | – | 15 |
| T ₂ | XIN Rise to Port Output Hold Time | 2 | – |

On-Chip Debugger Timing

Figure 32 and Table 113 provide timing information for the DBG pin. The DBG pin timing specifications assume a 4ns maximum rise and fall time.

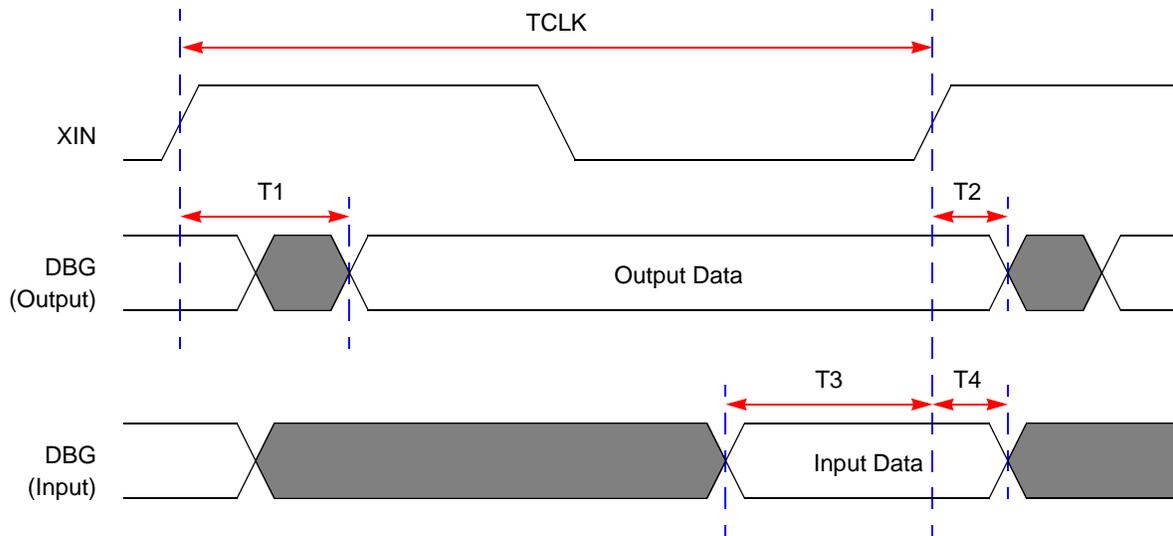


Figure 32. On-Chip Debugger Timing

Table 113. On-Chip Debugger Timing

| Parameter | Abbreviation | Delay (ns) | |
|----------------|----------------------------------|------------|---------|
| | | Minimum | Maximum |
| DBG | | | |
| T ₁ | XIN Rise to DBG Valid Delay | – | 15 |
| T ₂ | XIN Rise to DBG Output Hold Time | 2 | – |
| T ₃ | DBG to XIN Rise Input Setup Time | 5 | – |
| T ₄ | DBG to XIN Rise Input Hold Time | 5 | – |

UART Timing

Figure 33 and Table 114 provide timing information for UART pins for the case where CTS is used for flow control. The CTS to DE assertion delay (T1) assumes the transmit data register has been loaded with data prior to CTS assertion.

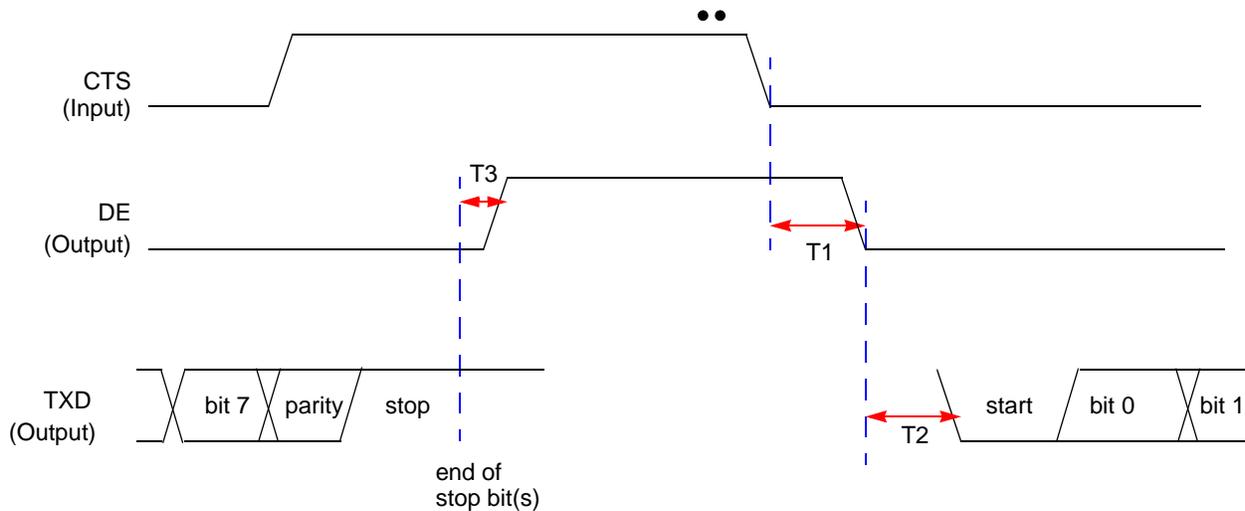


Figure 33. UART Timing With CTS

Table 114. UART Timing With CTS

| Parameter | Abbreviation | Delay (ns) | |
|----------------|--|----------------|-----------------------------|
| | | Minimum | Maximum |
| UART | | | |
| T ₁ | CTS Fall to DE output delay | 2 * XIN period | 2 * XIN period + 1 bit time |
| T ₂ | DE assertion to TXD falling edge (start bit) delay ± 5 | | |
| T ₃ | End of Stop Bit(s) to DE deassertion delay | ± 5 | |

Figure 34 and Table 115 provide timing information for UART pins for the case where CTS is not used for flow control. DE asserts after the transmit data register has been written. DE remains asserted for multiple characters as long as the transmit data register is written with the next character before the current character has completed.

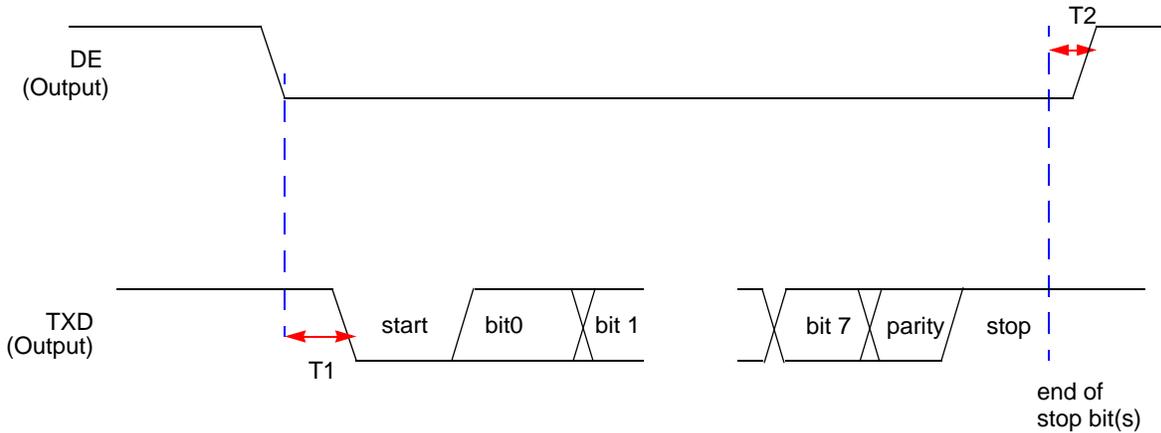


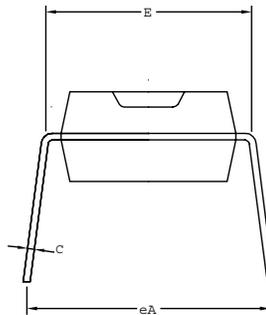
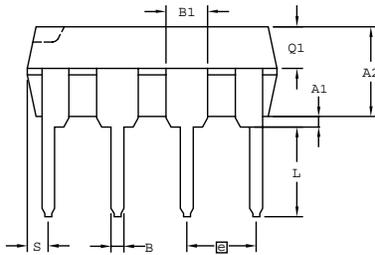
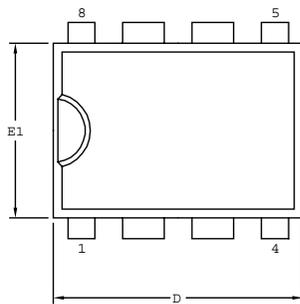
Figure 34. UART Timing Without CTS

Table 115. UART Timing Without CTS

| Parameter | Abbreviation | Delay (ns) | |
|----------------|--|----------------|-------------------|
| | | Minimum | Maximum |
| UART | | | |
| T ₁ | DE assertion to TXD falling edge (start bit) delay | 1 * XIN period | 1 bit time period |
| T ₂ | End of Stop Bit(s) to DE deassertion delay (Tx data register is empty) | ± 5 | |

Packaging

Figure 38 illustrates the 8-pin Plastic Dual Inline Package (PDIP) available for the Z8 Encore! XP™ 4K Series devices.

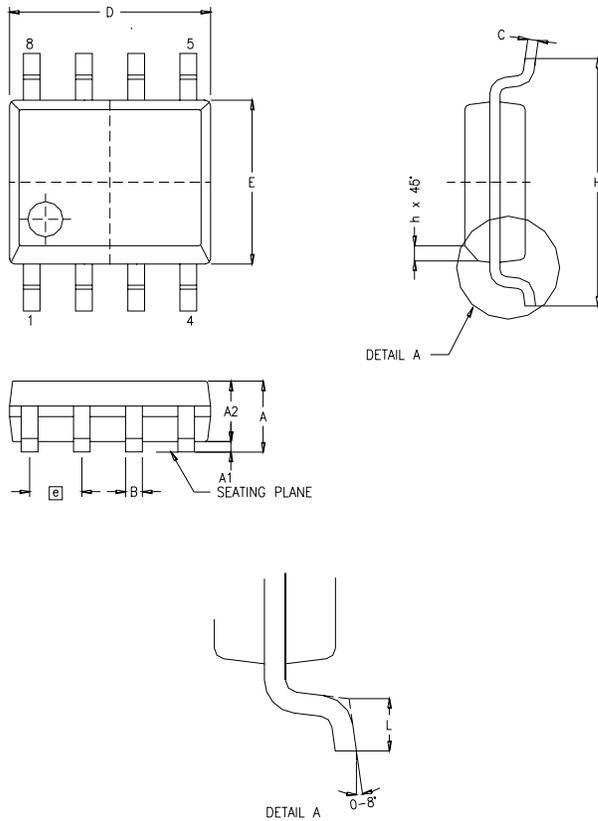


| SYMBOL | MILLIMETER | | INCH | |
|--------|------------|------|-----------|-------|
| | MIN | MAX | MIN | MAX |
| A1 | 0.38 | 0.81 | 0.015 | 0.032 |
| A2 | 3.25 | 3.81 | 0.128 | 0.150 |
| B | 0.38 | 0.53 | 0.015 | 0.021 |
| B1 | 1.40 | 1.65 | 0.055 | 0.065 |
| C | 0.20 | 0.30 | 0.008 | 0.012 |
| D | 9.02 | 9.78 | 0.355 | 0.385 |
| E | 7.62 | 8.26 | 0.300 | 0.325 |
| E1 | 6.10 | 6.60 | 0.240 | 0.260 |
| e | 2.54 BSC | | 0.100 BSC | |
| eA | 7.87 | 9.14 | 0.310 | 0.360 |
| L | 3.18 | 3.43 | 0.125 | 0.135 |
| Q1 | 1.40 | 1.65 | 0.055 | 0.065 |
| S | 0.64 | 0.89 | 0.025 | 0.035 |

CONTROLLING DIMENSIONS : MM.

Figure 35.8-Pin Plastic Dual Inline Package (PDIP)

Figure 36 illustrates the 8-pin Small Outline Integrated Circuit package (SOIC) available for the Z8 Encore! XP™ 4K Series devices.

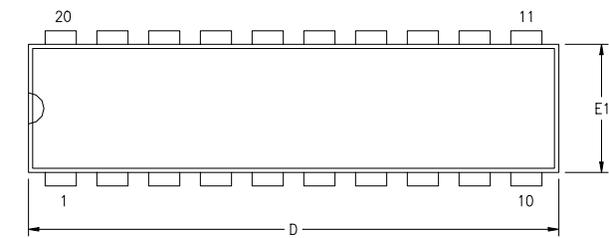


| SYMBOL | MILLIMETER | | INCH | |
|--------|------------|------|----------|-------|
| | MIN | MAX | MIN | MAX |
| A | 1.55 | 1.73 | 0.061 | 0.068 |
| A1 | 0.10 | 0.25 | 0.004 | 0.010 |
| A2 | 1.40 | 1.55 | 0.055 | 0.061 |
| B | 0.36 | 0.48 | 0.014 | 0.019 |
| C | 0.18 | 0.25 | 0.007 | 0.010 |
| D | 4.80 | 4.98 | 0.189 | 0.196 |
| E | 3.81 | 3.99 | 0.150 | 0.157 |
| ⓐ | 1.27 BSC | | .050 BSC | |
| H | 5.84 | 6.15 | 0.230 | 0.242 |
| h | 0.25 | 0.40 | 0.010 | 0.016 |
| L | 0.46 | 0.81 | 0.018 | 0.032 |

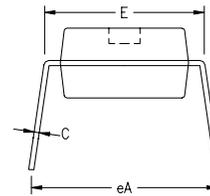
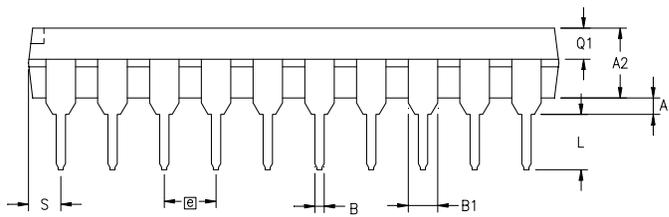
CONTROLLING DIMENSIONS : MM
LEADS ARE COPLANAR WITHIN .004 INCH.

Figure 36. 8-Pin Small Outline Integrated Circuit Package (SOIC)

Figure 38 illustrates the 20-pin Plastic Dual Inline Package (PDIP) available for the Z8 Encore! XP™ 4K Series devices.



| SYMBOL | MILLIMETER | | INCH | |
|--------|------------|-------|----------|-------|
| | MIN | MAX | MIN | MAX |
| A1 | 0.38 | 0.81 | .015 | .032 |
| A2 | 3.25 | 3.68 | .128 | .145 |
| B | 0.41 | 0.51 | .016 | .020 |
| B1 | 1.47 | 1.57 | .058 | .062 |
| C | 0.20 | 0.30 | .008 | .012 |
| D | 25.65 | 26.16 | 1.010 | 1.030 |
| E | 7.49 | 8.26 | .295 | .325 |
| E1 | 6.10 | 6.65 | .240 | .262 |
| Ⓢ | 2.54 BSC | | .100 BSC | |
| eA | 7.87 | 9.14 | .310 | .360 |
| L | 3.18 | 3.43 | .125 | .135 |
| Q1 | 1.42 | 1.65 | .056 | .065 |
| S | 1.52 | 1.65 | .060 | .065 |



CONTROLLING DIMENSIONS : INCH

Figure 37.20-Pin Plastic Dual Inline Package (PDIP)

Figure 38 illustrates the 20-pin Small Shrink Outline Package (SSOP) available for the Z8 Encore! XP™ 4K Series devices.

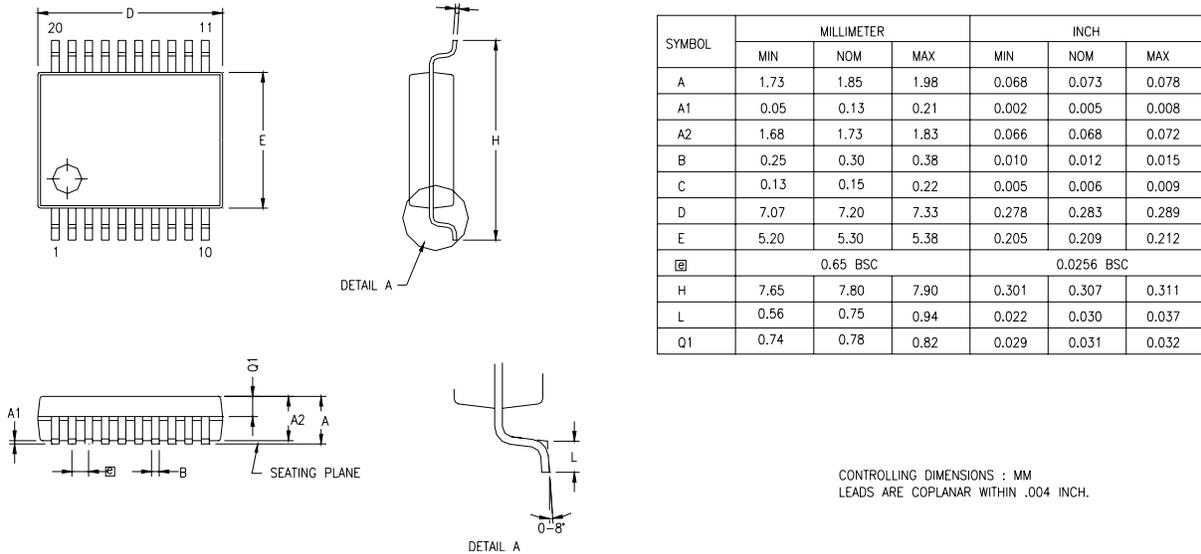
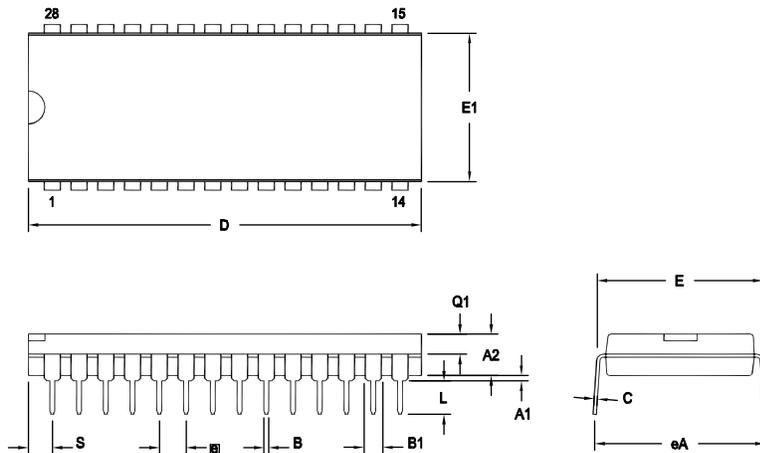


Figure 38.20-Pin Small Shrink Outline Package (SSOP)

Figure 39 illustrates the 28-pin Plastic Dual Inline Package (PDIP) available for the Z8 Encore! XP™ 4K Series devices.



| SYMBOL | OPT # | MILLIMETER | | INCH | |
|--------|-------|------------|-------|----------|-------|
| | | MIN | MAX | MIN | MAX |
| A1 | | 0.38 | 1.02 | .015 | .040 |
| A2 | | 3.18 | 4.19 | .125 | .165 |
| B | | 0.38 | 0.53 | .015 | .021 |
| B1 | 01 | 1.40 | 1.65 | .055 | .065 |
| | 02 | 1.14 | 1.40 | .045 | .055 |
| C | | 0.23 | 0.38 | .009 | .015 |
| D | 01 | 36.58 | 37.34 | 1.440 | 1.470 |
| | 02 | 35.31 | 35.94 | 1.390 | 1.415 |
| E | | 15.24 | 15.75 | .600 | .620 |
| E1 | 01 | 13.59 | 14.10 | .535 | .555 |
| | 02 | 12.83 | 13.08 | .505 | .515 |
| e | | 2.54 TYP | | .100 BSC | |
| eA | | 15.49 | 16.76 | .610 | .660 |
| L | | 3.05 | 3.81 | .120 | .150 |
| Q1 | 01 | 1.40 | 1.91 | .055 | .075 |
| | 02 | 1.40 | 1.78 | .055 | .070 |
| S | 01 | 1.52 | 2.29 | .060 | .090 |
| | 02 | 1.02 | 1.52 | .040 | .060 |

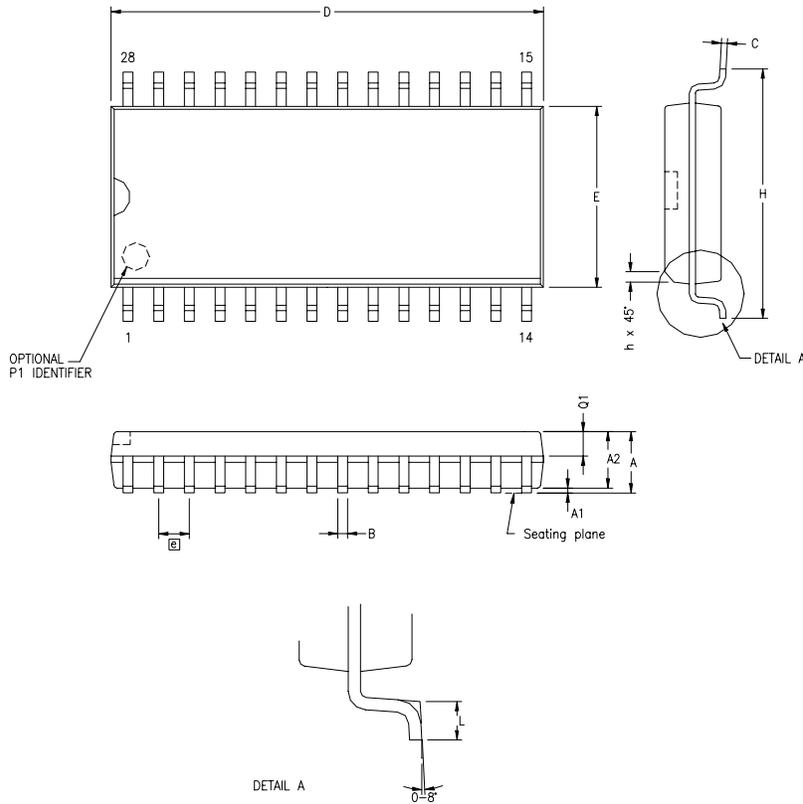
CONTROLLING DIMENSIONS : INCH

| OPTION TABLE | |
|--------------|----------|
| OPTION # | PACKAGE |
| 01 | STANDARD |
| 02 | IDF |

Note: ZILOG supplies both options for production. Component layout PCB design should cover bigger option 01.

Figure 39.28-Pin Plastic Dual Inline Package (PDIP)

Figure 40 illustrates the 28-pin Small Outline Integrated Circuit package (SOIC) available in the Z8 Encore! XP™ 4K Series devices.



| SYMBOL | MILLIMETER | | INCH | |
|--------|------------|-------|----------|------|
| | MIN | MAX | MIN | MAX |
| A | 2.40 | 2.64 | .094 | .104 |
| A1 | 0.10 | 0.30 | .004 | .012 |
| A2 | 2.24 | 2.44 | .088 | .096 |
| B | 0.36 | 0.46 | .014 | .018 |
| C | 0.23 | 0.30 | .009 | .012 |
| D | 17.78 | 18.00 | .700 | .710 |
| E | 7.40 | 7.60 | .291 | .299 |
| Ⓚ | 1.27 BSC | | .050 BSC | |
| H | 10.00 | 10.65 | .394 | .419 |
| h | 0.30 | 0.71 | .012 | .028 |
| L | 0.61 | 1.00 | .024 | .039 |
| Q1 | 0.97 | 1.09 | .038 | .043 |

CONTROLLING DIMENSIONS : MM
LEADS ARE COPLANAR WITHIN .004 INCH.

Figure 40.28-Pin Small Outline Integrated Circuit Package (SOIC)



Ordering Information

| Part Number | Flash | RAM | I/O Lines | Interrupts | 16-Bit Timers w/PWM | 10-Bit A/D Channels | UART with IrDA | Description |
|---|-------|-----|-----------|------------|---------------------|---------------------|----------------|---------------------|
| Z8 Encore!® with 4KB Flash, 10-Bit Analog-to-Digital Converter | | | | | | | | |
| Standard Temperature: 0° to 70°C | | | | | | | | |
| Z8F0423PB005SC | 4KB | 1KB | 6 | 12 | 2 | 4 | 1 | PDIP 8-pin package |
| Z8F0423SB005SC | 4KB | 1KB | 6 | 12 | 2 | 4 | 1 | SOIC 8-pin package |
| Z8F0423SH005SC | 4KB | 1KB | 16 | 18 | 2 | 7 | 1 | SOIC 20-pin package |
| Z8F0423HH005SC | 4KB | 1KB | 16 | 18 | 2 | 7 | 1 | SSOP 20-pin package |
| Z8F0423PH005SC | 4KB | 1KB | 16 | 18 | 2 | 7 | 1 | PDIP 20-pin package |
| Z8F0423SJ005SC | 4KB | 1KB | 22 | 18 | 2 | 8 | 1 | SOIC 28-pin package |
| Z8F0423HJ005SC | 4KB | 1KB | 22 | 18 | 2 | 8 | 1 | SSOP 28-pin package |
| Z8F0423PJ005SC | 4KB | 1KB | 22 | 18 | 2 | 8 | 1 | PDIP 28-pin package |
| Extended Temperature: -40° to 105°C | | | | | | | | |
| Z8F0423PB005EC | 4KB | 1KB | 6 | 12 | 2 | 4 | 1 | PDIP 8-pin package |
| Z8F0423SB005EC | 4KB | 1KB | 6 | 12 | 2 | 4 | 1 | SOIC 8-pin package |
| Z8F0423SH005EC | 4KB | 1KB | 16 | 18 | 2 | 7 | 1 | SOIC 20-pin package |
| Z8F0423HH005EC | 4KB | 1KB | 16 | 18 | 2 | 7 | 1 | SSOP 20-pin package |
| Z8F0423PH005EC | 4KB | 1KB | 16 | 18 | 2 | 7 | 1 | PDIP 20-pin package |
| Z8F0423SJ005EC | 4KB | 1KB | 22 | 18 | 2 | 8 | 1 | SOIC 28-pin package |
| Z8F0423HJ005EC | 4KB | 1KB | 22 | 18 | 2 | 8 | 1 | SSOP 28-pin package |
| Z8F0423PJ005EC | 4KB | 1KB | 22 | 18 | 2 | 8 | 1 | PDIP 28-pin package |



| Part Number | Flash | RAM | I/O Lines | Interrupts | 16-Bit Timers w/PWM | 10-Bit A/D Channels | UART with IrDA | Description |
|--|-------|-----|-----------|------------|---------------------|---------------------|----------------|---------------------|
| Z8 Encore!® with 4KB Flash | | | | | | | | |
| Standard Temperature: 0° to 70°C | | | | | | | | |
| Z8F0413PB005SC | 4KB | 1KB | 6 | 12 | 2 | 0 | 1 | PDIP 8-pin package |
| Z8F0413SB005SC | 4KB | 1KB | 6 | 12 | 2 | 0 | 1 | SOIC 8-pin package |
| Z8F0413SH005SC | 4KB | 1KB | 16 | 18 | 2 | 0 | 1 | SOIC 20-pin package |
| Z8F0413HH005SC | 4KB | 1KB | 16 | 18 | 2 | 0 | 1 | SSOP 20-pin package |
| Z8F0413PH005SC | 4KB | 1KB | 16 | 18 | 2 | 0 | 1 | PDIP 20-pin package |
| Z8F0413SJ005SC | 4KB | 1KB | 24 | 18 | 2 | 0 | 1 | SOIC 28-pin package |
| Z8F0413HJ005SC | 4KB | 1KB | 24 | 18 | 2 | 0 | 1 | SSOP 28-pin package |
| Z8F0413PJ005SC | 4KB | 1KB | 24 | 18 | 2 | 0 | 1 | PDIP 28-pin package |
| Extended Temperature: -40° to 105°C | | | | | | | | |
| Z8F0413PB005EC | 4KB | 1KB | 6 | 12 | 2 | 0 | 1 | PDIP 8-pin package |
| Z8F0413SB005EC | 4KB | 1KB | 6 | 12 | 2 | 0 | 1 | SOIC 8-pin package |
| Z8F0413SH005EC | 4KB | 1KB | 16 | 18 | 2 | 0 | 1 | SOIC 20-pin package |
| Z8F0413HH005EC | 4KB | 1KB | 16 | 18 | 2 | 0 | 1 | SSOP 20-pin package |
| Z8F0413PH005EC | 4KB | 1KB | 16 | 18 | 2 | 0 | 1 | PDIP 20-pin package |
| Z8F0413SJ005EC | 4KB | 1KB | 24 | 18 | 2 | 0 | 1 | SOIC 28-pin package |
| Z8F0413HJ005EC | 4KB | 1KB | 24 | 18 | 2 | 0 | 1 | SSOP 28-pin package |
| Z8F0413PJ005EC | 4KB | 1KB | 24 | 18 | 2 | 0 | 1 | PDIP 28-pin package |



| Part Number | Flash | RAM | I/O Lines | Interrupts | 16-Bit Timers w/PWM | 10-Bit A/D Channels | UART with IrDA | Description |
|---|-------|------|-----------|------------|---------------------|---------------------|----------------|---------------------|
| Z8 Encore!® with 2KB Flash, 10-Bit analog-to-Digital Converter | | | | | | | | |
| Standard Temperature: 0° to 70°C | | | | | | | | |
| Z8F0223PB005SC | 2KB | 512B | 6 | 12 | 2 | 4 | 1 | PDIP 8-pin package |
| Z8F0223SB005SC | 2KB | 512B | 6 | 12 | 2 | 4 | 1 | SOIC 8-pin package |
| Z8F0223SH005SC | 2KB | 512B | 16 | 18 | 2 | 7 | 1 | SOIC 20-pin package |
| Z8F0223HH005SC | 2KB | 512B | 16 | 18 | 2 | 7 | 1 | SSOP 20-pin package |
| Z8F0223PH005SC | 2KB | 512B | 16 | 18 | 2 | 7 | 1 | PDIP 20-pin package |
| Z8F0223SJ005SC | 2KB | 512B | 22 | 18 | 2 | 8 | 1 | SOIC 28-pin package |
| Z8F0223HJ005SC | 2KB | 512B | 22 | 18 | 2 | 8 | 1 | SSOP 28-pin package |
| Z8F0223PJ005SC | 2KB | 512B | 22 | 18 | 2 | 8 | 1 | PDIP 28-pin package |
| Extended Temperature: -40° to 105°C | | | | | | | | |
| Z8F0223PB005EC | 2KB | 512B | 6 | 12 | 2 | 4 | 1 | PDIP 8-pin package |
| Z8F0223SB005EC | 2KB | 512B | 6 | 12 | 2 | 4 | 1 | SOIC 8-pin package |
| Z8F0223SH005EC | 2KB | 512B | 16 | 18 | 2 | 7 | 1 | SOIC 20-pin package |
| Z8F0223HH005EC | 2KB | 512B | 16 | 18 | 2 | 7 | 1 | SSOP 20-pin package |
| Z8F0223PH005EC | 2KB | 512B | 16 | 18 | 2 | 7 | 1 | PDIP 20-pin package |
| Z8F0223SJ005EC | 2KB | 512B | 22 | 18 | 2 | 8 | 1 | SOIC 28-pin package |
| Z8F0223HJ005EC | 2KB | 512B | 22 | 18 | 2 | 8 | 1 | SSOP 28-pin package |
| Z8F0223PJ005EC | 2KB | 512B | 22 | 18 | 2 | 8 | 1 | PDIP 28-pin package |



| Part Number | Flash | RAM | I/O Lines | Interrupts | 16-Bit Timers w/PWM | 10-Bit A/D Channels | UART with IrDA | Description |
|--|-------|------|-----------|------------|---------------------|---------------------|----------------|---------------------|
| Z8 Encore!® with 2KB Flash | | | | | | | | |
| Standard Temperature: 0° to 70°C | | | | | | | | |
| Z8F0213PB005SC | 2KB | 512B | 6 | 12 | 2 | 0 | 1 | PDIP 8-pin package |
| Z8F0213SB005SC | 2KB | 512B | 6 | 12 | 2 | 0 | 1 | SOIC 8-pin package |
| Z8F0213SH005SC | 2KB | 512B | 16 | 18 | 2 | 0 | 1 | SOIC 20-pin package |
| Z8F0213HH005SC | 2KB | 512B | 16 | 18 | 2 | 0 | 1 | SSOP 20-pin package |
| Z8F0213PH005SC | 2KB | 512B | 16 | 18 | 2 | 0 | 1 | PDIP 20-pin package |
| Z8F0213SJ005SC | 2KB | 512B | 24 | 18 | 2 | 0 | 1 | SOIC 28-pin package |
| Z8F0213HJ005SC | 2KB | 512B | 24 | 18 | 2 | 0 | 1 | SSOP 28-pin package |
| Z8F0213PJ005SC | 2KB | 512B | 24 | 18 | 2 | 0 | 1 | PDIP 28-pin package |
| Extended Temperature: -40° to 105°C | | | | | | | | |
| Z8F0213PB005EC | 2KB | 512B | 6 | 12 | 2 | 0 | 1 | PDIP 8-pin package |
| Z8F0213SB005EC | 2KB | 512B | 6 | 12 | 2 | 0 | 1 | SOIC 8-pin package |
| Z8F0213SH005EC | 2KB | 512B | 16 | 18 | 2 | 0 | 1 | SOIC 20-pin package |
| Z8F0213HH005EC | 2KB | 512B | 16 | 18 | 2 | 0 | 1 | SSOP 20-pin package |
| Z8F0213PH005EC | 2KB | 512B | 16 | 18 | 2 | 0 | 1 | PDIP 20-pin package |
| Z8F0213SJ005EC | 2KB | 512B | 24 | 18 | 2 | 0 | 1 | SOIC 28-pin package |
| Z8F0213HJ005EC | 2KB | 512B | 24 | 18 | 2 | 0 | 1 | SSOP 28-pin package |
| Z8F0213PJ005EC | 2KB | 512B | 24 | 18 | 2 | 0 | 1 | PDIP 28-pin package |



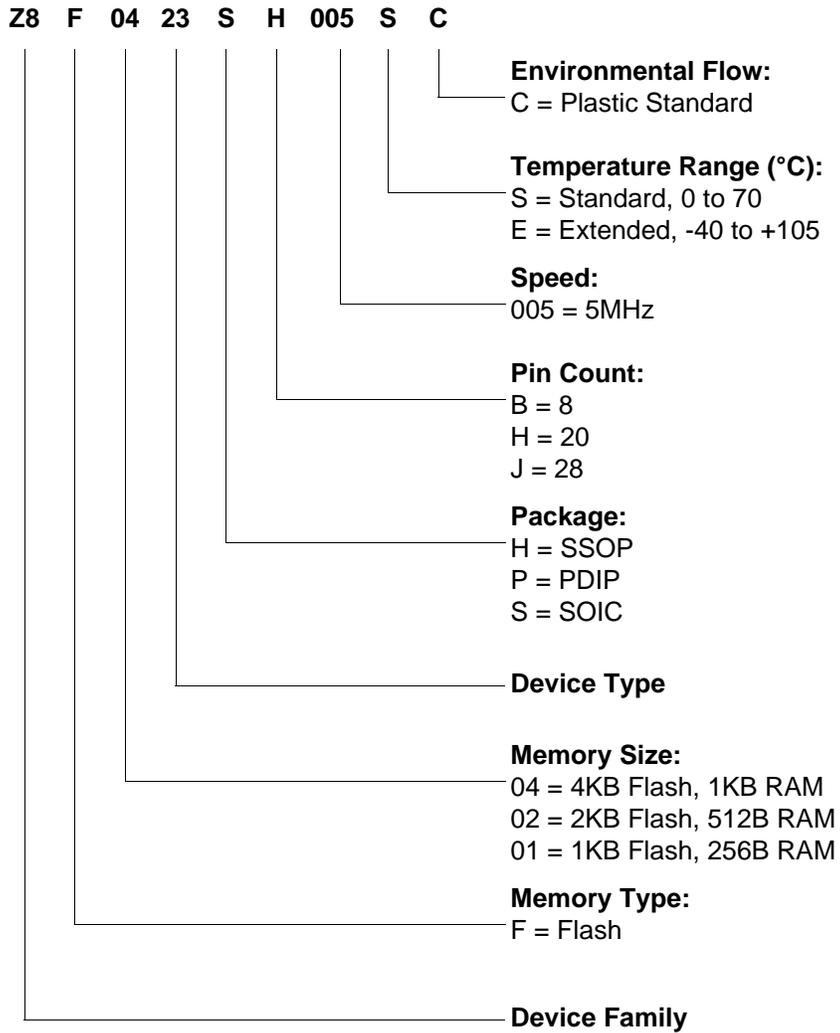
| Part Number | Flash | RAM | I/O Lines | Interrupts | 16-Bit Timers w/PWM | 10-Bit A/D Channels | UART with IrDA | Description |
|---|-------|------|-----------|------------|---------------------|---------------------|----------------|---------------------|
| Z8 Encore!® with 1KB Flash, 10-Bit Analog-to-Digital Converter | | | | | | | | |
| Standard Temperature: 0° to 70°C | | | | | | | | |
| Z8F0123PB005SC | 1KB | 256B | 6 | 12 | 2 | 4 | 1 | PDIP 8-pin package |
| Z8F0123SB005SC | 1KB | 256B | 6 | 12 | 2 | 4 | 1 | SOIC 8-pin package |
| Z8F0123SH005SC | 1KB | 256B | 16 | 18 | 2 | 7 | 1 | SOIC 20-pin package |
| Z8F0123HH005SC | 1KB | 256B | 16 | 18 | 2 | 7 | 1 | SSOP 20-pin package |
| Z8F0123PH005SC | 1KB | 256B | 16 | 18 | 2 | 7 | 1 | PDIP 20-pin package |
| Z8F0123SJ005SC | 1KB | 256B | 22 | 18 | 2 | 8 | 1 | SOIC 28-pin package |
| Z8F0123HJ005SC | 1KB | 256B | 22 | 18 | 2 | 8 | 1 | SSOP 28-pin package |
| Z8F0123PJ005SC | 1KB | 256B | 22 | 18 | 2 | 8 | 1 | PDIP 28-pin package |
| Extended Temperature: -40° to 105°C | | | | | | | | |
| Z8F0123PB005EC | 1KB | 256B | 6 | 12 | 2 | 4 | 1 | PDIP 8-pin package |
| Z8F0123SB005EC | 1KB | 256B | 6 | 12 | 2 | 4 | 1 | SOIC 8-pin package |
| Z8F0123SH005EC | 1KB | 256B | 16 | 18 | 2 | 7 | 1 | SOIC 20-pin package |
| Z8F0123HH005EC | 1KB | 256B | 16 | 18 | 2 | 7 | 1 | SSOP 20-pin package |
| Z8F0123PH005EC | 1KB | 256B | 16 | 18 | 2 | 7 | 1 | PDIP 20-pin package |
| Z8F0123SJ005EC | 1KB | 256B | 22 | 18 | 2 | 8 | 1 | SOIC 28-pin package |
| Z8F0123HJ005EC | 1KB | 256B | 22 | 18 | 2 | 8 | 1 | SSOP 28-pin package |
| Z8F0123PJ005EC | 1KB | 256B | 22 | 18 | 2 | 8 | 1 | PDIP 28-pin package |



| Part Number | Flash | RAM | I/O Lines | Interrupts | 16-Bit Timers w/PWM | 10-Bit A/D Channels | UART with IrDA | Description |
|--|-------|------|-----------|------------|---------------------|---------------------|----------------|------------------------|
| Z8 Encore!® with 1KB Flash | | | | | | | | |
| Standard Temperature: 0° to 70°C | | | | | | | | |
| Z8F0113PB005SC | 1KB | 256B | 6 | 12 | 2 | 0 | 1 | PDIP 8-pin package |
| Z8F0113SB005SC | 1KB | 256B | 6 | 12 | 2 | 0 | 1 | SOIC 8-pin package |
| Z8F0113SH005SC | 1KB | 256B | 16 | 18 | 2 | 0 | 1 | SOIC 20-pin package |
| Z8F0113HH005SC | 1KB | 256B | 16 | 18 | 2 | 0 | 1 | SSOP 20-pin package |
| Z8F0113PH005SC | 1KB | 256B | 16 | 18 | 2 | 0 | 1 | PDIP 20-pin package |
| Z8F0113SJ005SC | 1KB | 256B | 24 | 18 | 2 | 0 | 1 | SOIC 28-pin package |
| Z8F0113HJ005SC | 1KB | 256B | 24 | 18 | 2 | 0 | 1 | SSOP 28-pin package |
| Z8F0113PJ005SC | 1KB | 256B | 24 | 18 | 2 | 0 | 1 | PDIP 28-pin package |
| Extended Temperature: -40° to 105°C | | | | | | | | |
| Z8F0113PB005EC | 1KB | 256B | 6 | 12 | 2 | 0 | 1 | PDIP 8-pin package |
| Z8F0113SB005EC | 1KB | 256B | 6 | 12 | 2 | 0 | 1 | SOIC 8-pin package |
| Z8F0113SH005EC | 1KB | 256B | 16 | 18 | 2 | 0 | 1 | SOIC 20-pin package |
| Z8F0113HH005EC | 1KB | 256B | 16 | 18 | 2 | 0 | 1 | SSOP 20-pin package |
| Z8F0113PH005EC | 1KB | 256B | 16 | 18 | 2 | 0 | 1 | PDIP 20-pin package |
| Z8F0113SJ005EC | 1KB | 256B | 24 | 18 | 2 | 0 | 1 | SOIC 28-pin package |
| Z8F0113HJ005EC | 1KB | 256B | 24 | 18 | 2 | 0 | 1 | SSOP 28-pin package |
| Z8F0113PJ005EC | 1KB | 256B | 24 | 18 | 2 | 0 | 1 | PDIP 28-pin package |
| Z8F04328100KIT | | | | | | | | Development Kit |



Part Number Suffix Designations





Precharacterization Product

The product represented by this document is newly introduced and ZiLOG has not completed the full characterization of the product. The document states what ZiLOG knows about this product at this time, but additional features or nonconformance with some aspects of the document might be found, either by ZiLOG or its customers in the course of further application and characterization work. In addition, ZiLOG cautions that delivery might be uncertain at times, because of start-up yield issues.

ZiLOG, Inc.
532 Race Street
San Jose, CA 95126
Telephone (408) 558-8500
FAX 408 558-8300
Internet: www.zilog.com

Customer Support

For valuable information about downloading other relevant documents or for hardware and software development tools, visit the ZiLOG web site at www.zilog.com.



Customer Feedback Form

Z8 Encore!® 4K Series High Performance MCUs with Flash Memory and 10-Bit A/D Converter Product Specification

If you experience any problems while operating this product, or if you note any inaccuracies while reading this document, please copy and complete this form and mail it to the address below or go to www.zilog.com (see Return Information, below). We also welcome your suggestions!

Customer Information

| | |
|----------------|---------|
| Name | Country |
| Company | Phone |
| Address | Fax |
| City/State/Zip | Email |

Product Information

| |
|--------------------------------|
| Serial # or Board Fab #/Rev. # |
| Software Version |
| Document Number |
| Host Computer Description/Type |



Index

Symbols

150
% 150
@ 150

Numerics

10-bit ADC 4
40-lead plastic dual-inline package 184

A

absolute maximum ratings 167
AC characteristics 170
ADC 151
 architecture 107
 automatic power-down 108
 block diagram 108
 continuous conversion 110
 control register 111, 112
 control register definitions 111
 data high byte register 113
 data low bits register 113
 electrical characteristics and timing 173
 operation 108
 single-shot conversion 109
ADCCTL register 111, 112
ADCDH register 113
ADC DL register 113
ADCX 151
ADD 151
add - extended addressing 151
add with carry 151
add with carry - extended addressing 151
additional symbols 150
address space 13
ADDX 151
analog signals 9
analog-to-digital converter (ADC) 107
AND 153

ANDX 153
arithmetic instructions 151
assembly language programming 147
assembly language syntax 148

B

B 150
b 149
baud rate generator, UART 94
BCLR 152
binary number suffix 150
BIT 152
bit 149
 clear 152
 manipulation instructions 152
 set 152
 set or clear 152
 swap 152
 test and jump 154
 test and jump if non-zero 154
 test and jump if zero 154
bit jump and test if non-zero 154
bit swap 154
block diagram 2
block transfer instructions 152
BRK 154
BSET 152
BSWAP 152, 154
BTJ 154
BTJNZ 154
BTJZ 154

C

CALL procedure 154
capture mode 76, 77
capture/compare mode 76
cc 149
CCF 152
characteristics, electrical 167
clear 153
CLR 153
COM 153



- compare 76
- compare - extended addressing 151
- compare mode 76
- compare with carry 151
- compare with carry - extended addressing 151
- complement 153
- complement carry flag 152
- condition code 149
- continuous conversion (ADC) 110
- continuous mode 76
- control register definition, UART 95
- Control Registers 13, 16
- counter modes 76
- CP 151
- CPC 151
- CPCX 151
- CPU and peripheral overview 4
- CPU control instructions 152
- CPX 151
- Customer Feedback Form 193
- Customer Information 193

D

- DA 149, 151
- data memory 15
- DC characteristics 168
- debugger, on-chip 131
- DEC 151
- decimal adjust 151
- decrement 151
- decrement and jump non-zero 154
- decrement word 151
- DECW 151
- destination operand 150
- device, port availability 31
- DI 152
- direct address 149
- disable interrupts 152
- DJNZ 154
- dst 150

E

- EI 152
- electrical characteristics 167
 - ADC 173
 - flash memory and timing 172
 - GPIO input data sample timing 173
 - watch-dog timer 172
- enable interrupt 152
- ER 149
- extended addressing register 149
- external pin reset 24
- eZ8 CPU features 4
- eZ8 CPU instruction classes 150
- eZ8 CPU instruction notation 148
- eZ8 CPU instruction set 147
- eZ8 CPU instruction summary 155

F

- FCTL register 122, 127
- features, Z8 Encore! 1
- first opcode map 165
- FLAGS 150
- flags register 150
- flash
 - controller 4
 - option bit address space 128
 - option bit configuration - reset 126
 - program memory address 0000H 128
 - program memory address 0001H 129
- flash memory 115
 - arrangement 116
 - byte programming 121
 - code protection 119
 - configurations 115
 - control register definitions 122, 127
 - controller bypass 122
 - electrical characteristics and timing 172
 - flash control register 122, 127
 - flash option bits 120
 - flash status register 123
 - flow chart 118
 - frequency high and low byte registers 125
 - mass erase 121



- operation 117
- operation timing 119
- page erase 121
- page select register 123, 124
- FPS register 123, 124
- FSTAT register 123

G

- gated mode 76
- general-purpose I/O 31
- GPIO 4, 31
 - alternate functions 32
 - architecture 32
 - control register definitions 38
 - input data sample timing 173
- interrupts 38
- port A-C pull-up enable sub-registers 43, 44
- port A-H address registers 39
- port A-H alternate function sub-registers 40
- port A-H control registers 39
- port A-H data direction sub-registers 40
- port A-H high drive enable sub-registers 42
- port A-H input data registers 44
- port A-H output control sub-registers 41
- port A-H output data registers 45
- port A-H stop mode recovery sub-registers 42
- port availability by device 31
- port input timing 174
- port output timing 175

H

- H 150
- HALT 152
- halt mode 29, 152
- hexadecimal number prefix/suffix 150

I

- I2C 4
- IM 149

- immediate data 149
- immediate operand prefix 150
- INC 151
- increment 151
- increment word 151
- INCW 151
- indexed 149
- indirect address prefix 150
- indirect register 149
- indirect register pair 149
- indirect working register 149
- indirect working register pair 149
- infrared encoder/decoder (IrDA) 103
- Instruction Set 147
- instruction set, ez8 CPU 147
- instructions
 - ADC 151
 - ADCX 151
 - ADD 151
 - ADDX 151
 - AND 153
 - ANDX 153
 - arithmetic 151
 - BCLR 152
 - BIT 152
 - bit manipulation 152
 - block transfer 152
 - BRK 154
 - BSET 152
 - BSWAP 152, 154
 - BTJ 154
 - BTJNZ 154
 - BTJZ 154
 - CALL 154
 - CCF 152
 - CLR 153
 - COM 153
 - CP 151
 - CPC 151
 - CPCX 151
 - CPU control 152
 - CPX 151
 - DA 151
 - DEC 151



DECW 151
 DI 152
 DJNZ 154
 EI 152
 HALT 152
 INC 151
 INCW 151
 IRET 154
 JP 154
 LD 153
 LDC 153
 LDCI 152, 153
 LDE 153
 LDEI 152
 LDX 153
 LEA 153
 load 153
 logical 153
 MULT 151
 NOP 152
 OR 153
 ORX 153
 POP 153
 POPX 153
 program control 154
 PUSH 153
 PUSHX 153
 RCF 152
 RET 154
 RL 154
 RLC 154
 rotate and shift 154
 RR 154
 RRC 154
 SBC 151
 SCF 152
 SRA 154
 SRL 154
 SRP 152
 STOP 153
 SUB 151
 SUBX 151
 SWAP 154
 TCM 152
 TCMX 152
 TM 152
 TMX 152
 TRAP 154
 watch-dog timer refresh 153
 XOR 153
 XORX 153
 instructions, eZ8 classes of 150
 interrupt control register 56
 interrupt controller 46
 architecture 46
 interrupt assertion types 49
 interrupt vectors and priority 49
 operation 48
 register definitions 50
 software interrupt assertion 50
 interrupt edge select register 56
 interrupt request 0 register 50
 interrupt request 1 register 51
 interrupt request 2 register 52
 interrupt return 154
 interrupt vector listing 46
 interrupts
 UART 92
 IR 149
 Ir 149
 IrDA
 architecture 103
 block diagram 103
 control register definitions 106
 operation 103
 receiving data 105
 transmitting data 104
 IRET 154
 IRQ0 enable high and low bit registers 52
 IRQ1 enable high and low bit registers 54
 IRQ2 enable high and low bit registers 55
 IRR 149
 Irr 149

J
 JP 154
 jump, conditional, relative, and relative condi-



tional 154

L

LD 153
LDC 153
LDCI 152, 153
LDE 153
LDEI 152, 153
LDX 153
LEA 153
load 153
load constant 152
load constant to/from program memory 153
load constant with auto-increment addresses 153
load effective address 153
load external data 153
load external data to/from data memory and auto-increment addresses 152
load external to/from data memory and auto-increment addresses 153
load instructions 153
load using extended addressing 153
logical AND 153
logical AND/extended addressing 153
logical exclusive OR 153
logical exclusive OR/extended addressing 153
logical instructions 153
logical OR 153
logical OR/extended addressing 153
low power modes 28

M

master interrupt enable 48
memory
 data 15
 program 14
mode
 capture 76, 77
 capture/compare 76
 continuous 76
 counter 76

gated 76
one-shot 75
PWM 76

modes 76
MULT 151
multiply 151
multiprocessor mode, UART 90

N

NOP (no operation) 152
notation
 b 149
 cc 149
 DA 149
 ER 149
 IM 149
 IR 149
 I_r 149
 IRR 149
 I_{rr} 149
 p 149
 R 149
 r 149
 RA 149
 RR 149
 rr 149
 vector 149
 X 149
notational shorthand 149

O

OCD
 architecture 131
 auto-baud detector/generator 134
 baud rate limits 134
 block diagram 131
 breakpoints 135
 commands 136
 control register 140
 data format 134
 DBG pin to RS-232 Interface 132
 debug mode 133



- debugger break 154
- interface 132
- serial errors 135
- status register 142
- timing 176
- OCD commands
 - execute instruction (12H) 140
 - read data memory (0DH) 139
 - read OCD control register (05H) 137
 - read OCD revision (00H) 137
 - read OCD status register (02H) 137
 - read program counter (07H) 138
 - read program memory (0BH) 139
 - read program memory CRC (0EH) 140
 - read register (09H) 138
 - read runtime counter (03H) 137
 - step instruction (10H) 140
 - stuff instruction (11H) 140
 - write data memory (0CH) 139
 - write OCD control register (04H) 137
 - write program counter (06H) 138
 - write program memory (0AH) 138
 - write register (08H) 138
- on-chip debugger (OCD) 131
- on-chip debugger signals 10
- one-shot mode 75
- opcode map
 - abbreviations 164
 - cell description 163
 - first 165
 - second after 1FH 166
- Operational Description 20, 28, 31, 46, 58, 79, 84, 103, 107, 115, 126, 131, 143, 146
- OR 153
- ordering information 185
- ORX 153

- P**
- p 149
- packaging
 - 20-pin PDIP 180
 - 20-pin SSOP 182
 - 28-pin PDIP 183
 - 28-pin SOIC 183
 - 8-pin PDIP 179
 - 8-pin SOIC 179
 - PDIP 184
- part selection guide 2
- PC 150
- PDIP 184
- peripheral AC and DC electrical characteristics 171
- pin characteristics 10
- Pin Descriptions 7
- polarity 149
- POP 153
- pop using extended addressing 153
- POPX 153
- port availability, device 31
- port input timing (GPIO) 174
- port output timing, GPIO 175
- power supply signals 10
- power-down, automatic (ADC) 108
- power-on and voltage brown-out electrical characteristics and timing 171
- power-on reset (POR) 22
- Problem Description or Suggestion 194
- Product Information 193
- program control instructions 154
- program counter 150
- program memory 14
- PUSH 153
- push using extended addressing 153
- PUSHX 153
- PWM mode 76
- PxADDR register 39
- PxCTL register 40

- R**
- R 149
- r 149
- RA
 - register address 149
- RCF 152
- receive
 - IrDA data 105



- receiving UART data-interrupt-driven method 89
- receiving UART data-pollled method 88
- register 149
 - ADC control (ADCCTL) 111, 112
 - ADC data high byte (ADCDH) 113
 - ADC data low bits (ADC DL) 113
 - flash control (FCTL) 122, 127
 - flash high and low byte (FFREQH and FREEQL) 125
 - flash page select (FPS) 123, 124
 - flash status (FSTAT) 123
 - GPIO port A-H address (PxADDR) 39
 - GPIO port A-H alternate function sub-registers 41
 - GPIO port A-H control address (PxCTL) 40
 - GPIO port A-H data direction sub-registers 40
 - OCD control 140
 - OCD status 142
 - UARTx baud rate high byte (UxBRH) 101
 - UARTx baud rate low byte (UxBRL) 101
 - UARTx Control 0 (UxCTL0) 98, 101
 - UARTx control 1 (UxCTL1) 99
 - UARTx receive data (UxRXD) 96
 - UARTx status 0 (UxSTAT0) 96
 - UARTx status 1 (UxSTAT1) 97
 - UARTx transmit data (UxTXD) 95
 - watch-dog timer control (WDTCTL) 26, 82, 144
 - watch-dog timer reload high byte (WDTH) 83
 - watch-dog timer reload low byte (WDTL) 83
 - watch-dog timer reload upper byte (WDTU) 83
- register file 13
- register pair 149
- register pointer 150
- reset
 - and stop mode characteristics 20
 - and stop mode recovery 20
 - carry flag 152
 - sources 21
- RET 154
- return 154
- Return Information 194
- RL 154
- RLC 154
- rotate and shift instructions 154
- rotate left 154
- rotate left through carry 154
- rotate right 154
- rotate right through carry 154
- RP 150
- RR 149, 154
- rr 149
- RRC 154

- S**
- SBC 151
- SCF 152
- second opcode map after 1FH 166
- set carry flag 152
- set register pointer 152
- shift right arithmetic 154
- shift right logical 154
- signal descriptions 9
- single-shot conversion (ADC) 109
- software trap 154
- source operand 150
- SP 150
- SRA 154
- src 150
- SRL 154
- SRP 152
- stack pointer 150
- STOP 153
- stop mode 28, 153
- stop mode recovery
 - sources 24
 - using a GPIO port pin transition 25, 26
 - using watch-dog timer time-out 25
- SUB 151
- subtract 151
- subtract - extended addressing 151
- subtract with carry 151



subtract with carry - extended addressing 151
SUBX 151
SWAP 154
swap nibbles 154
symbols, additional 150

T

TCM 152
TCMX 152
test complement under mask 152
test complement under mask - extended addressing 152
test under mask 152
test under mask - extended addressing 152
timer signals 9
timers 58
 architecture 58
 block diagram 59
 capture mode 66, 67, 76, 77
 capture/compare mode 70, 76
 compare mode 68, 76
 continuous mode 60, 76
 counter mode 61, 62
 counter modes 76
 gated mode 69, 76
 one-shot mode 59, 75
 operating mode 59
 PWM mode 63, 64, 76
 reading the timer count values 71
 reload high and low byte registers 72
 timer control register definitions 72
 timer output signal operation 71
timers 0-3
 control registers 74, 75
 high and low byte registers 72, 73
TM 152
TMX 152
tools, hardware and software 192
transmit
 IrDA data 104
transmitting UART data-polled method 86
transmitting UART data-interrupt-driven method 87

TRAP 154

U

UART 4
 architecture 84
 baud rate generator 94
 baud rates table 102
 control register definitions 95
 controller signals 9
 data format 85
 interrupts 92
 multiprocessor mode 90
 receiving data using interrupt-driven method 89
 receiving data using the polled method 88
 transmitting data using the interrupt-driven method 87
 transmitting data using the polled method 86
 x baud rate high and low registers 101
 x control 0 and control 1 registers 98
 x status 0 and status 1 registers 96, 97
UxBRH register 101
UxBRL register 101
UxCTL0 register 98, 101
UxCTL1 register 99
UxRXD register 96
UxSTAT0 register 96
UxSTAT1 register 97
UxTXD register 95

V

vector 149
voltage brown-out reset (VBR) 22

W

watch-dog timer
 approximate time-out delay 80
 approximate time-out delays 79, 143, 146
CNTL 23
control register 82, 144



- electrical characteristics and timing 172
- interrupt in normal operation 80
- interrupt in stop mode 81
- operation 79, 143, 146
- refresh 80, 153
- reload unlock sequence 81
- reload upper, high and low registers 82
- reset 23
- reset in normal operation 81
- reset in Stop mode 81
- time-out response 80
- WDTCTL register 26, 82, 144
- WDTH register 83
- WDTL register 83
- working register 149
- working register pair 149
- WTDU register 83

X

- X 149
- XOR 153
- XORX 153

Z

- Z8 Encore!
 - block diagram 2
 - features 1
 - part selection guide 2