



Welcome to [E-XFL.COM](https://www.e-xfl.com)

What is "[Embedded - Microcontrollers](#)"?

"[Embedded - Microcontrollers](#)" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "[Embedded - Microcontrollers](#)"

Details

Product Status	Obsolete
Core Processor	eZ8
Core Size	8-Bit
Speed	20MHz
Connectivity	I ² C, IrDA, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	31
Program Memory Size	24KB (24K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 3.6V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Through Hole
Package / Case	40-DIP (0.620", 15.75mm)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f2401pm020ec



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

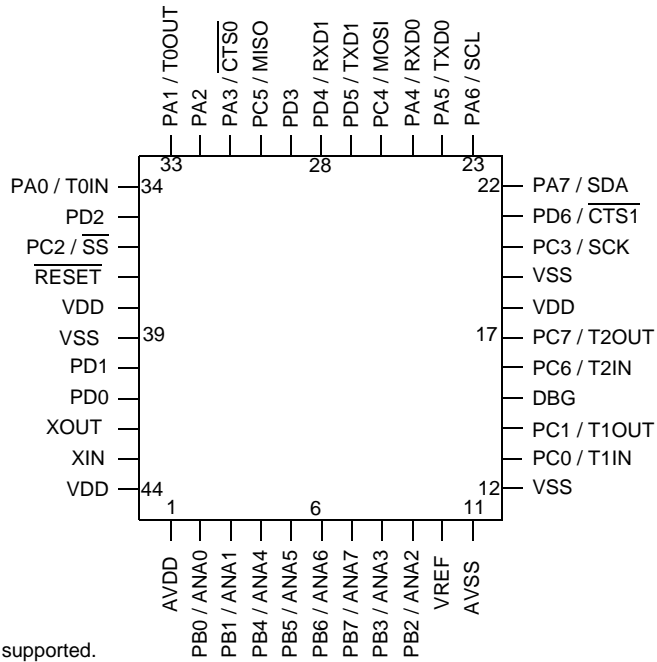
Document Disclaimer

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concerning the devices, applications, or technology described is intended to suggest possible uses and may be superseded.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 101. Absolute Maximum Ratings	167
Table 102. DC Characteristics	169
Table 103. AC Characteristics	172
Table 104. Power-On Reset and Voltage Brown-Out Electrical Characteristics and Timing	173
Table 105. Flash Memory Electrical Characteristics and Timing . . .	173
Table 106. Watch-Dog Timer Electrical Characteristics and Timing	174
Table 107. Analog-to-Digital Converter Electrical Characteristics and Timing	174
Table 108. GPIO Port Input Timing	176
Table 109. GPIO Port Output Timing	177
Table 110. On-Chip Debugger Timing	178
Table 111. SPI Master Mode Timing	179
Table 112. SPI Slave Mode Timing	180
Table 113. I2C Timing	181
Table 114. Assembly Language Syntax Example 1	183
Table 115. Assembly Language Syntax Example 2	183
Table 116. Notational Shorthand	184
Table 117. Additional Symbols	185
Table 118. Condition Codes	186
Table 119. Arithmetic Instructions	187
Table 120. Bit Manipulation Instructions	188
Table 121. Block Transfer Instructions	188
Table 122. CPU Control Instructions	189
Table 123. Load Instructions	189
Table 124. Logical Instructions	190
Table 125. Program Control Instructions	190
Table 126. Rotate and Shift Instructions	191
Table 127. eZ8 CPU Instruction Summary	191
Table 128. Opcode Map Abbreviations	203
Table 129. Ordering Information	211



Note: Timer 3 is not supported.

Figure 58. Z8Fxx01 in 44-Pin Low-Profile Quad Flat Package (LQFP)

Architecture

Figure 65 illustrates a block diagram of the interrupt controller.

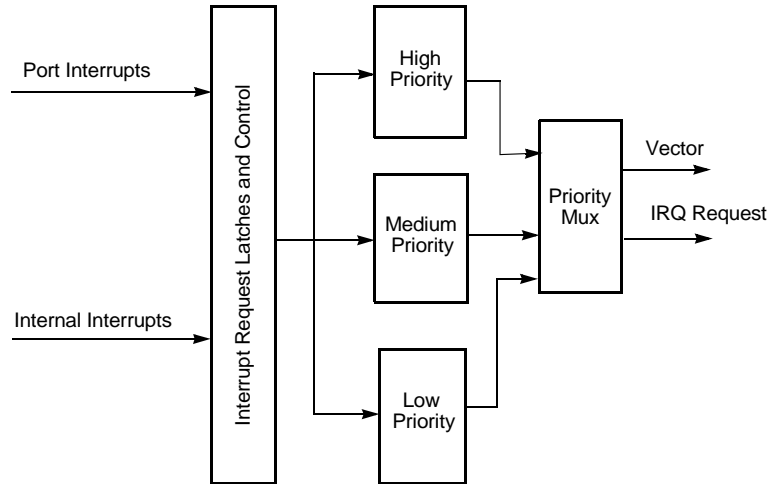


Figure 65. Interrupt Controller Block Diagram

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



Timers

Overview

The Z8F640x family products contain three to four 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 Generators for any unused UART, SPI, or I²C peripherals may also be used to provide basic timing functionality. Refer to the respective serial communication peripheral chapters for information on using the Baud Rate Generators as timers. Timer 3 is unavailable in the 40- and 44-pin packages.

Architecture

Figure 66 illustrates the architecture of the timers.



- 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. This 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. If desired, 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. 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 desired 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 desired 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.

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 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. If desired, enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers.



UART

Overview

The Universal Asynchronous Receiver/Transmitter (UART) is a full-duplex communication channel capable of handling asynchronous data transfers. The Z8F640x family device contains two fully independent UARTs. 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
- Selectable 9-bit multiprocessor (9-bit) mode
- 16-bit Baud Rate Generator (BRG)

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 67 illustrates the UART architecture.

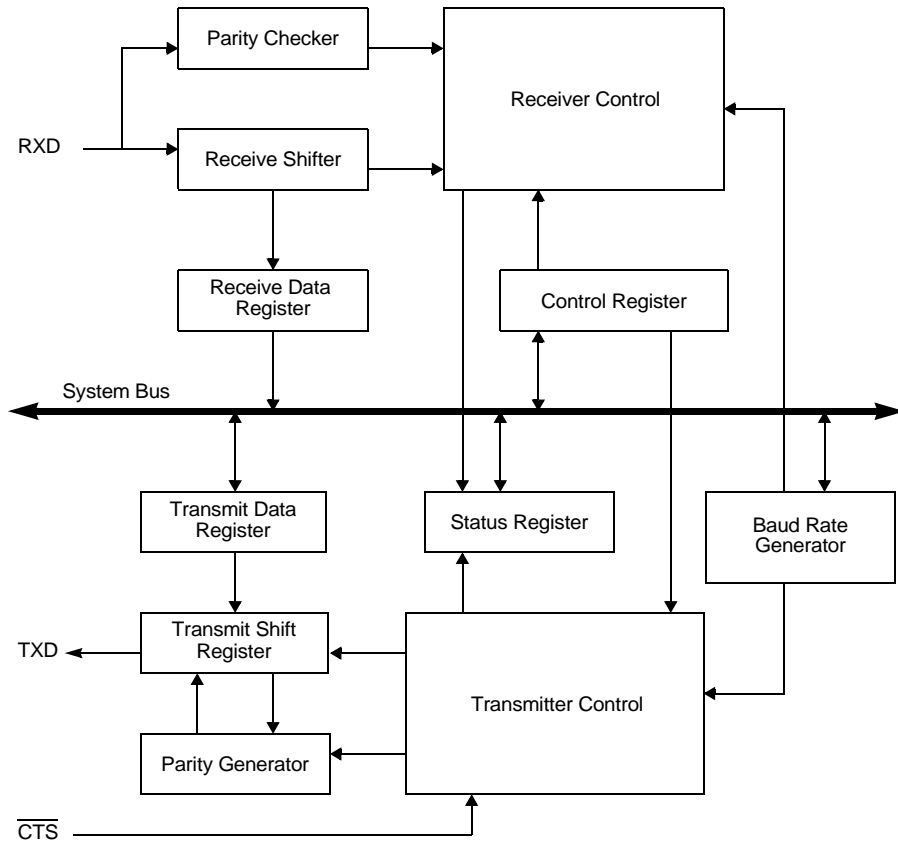


Figure 67. UART Block Diagram

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 optionally 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.

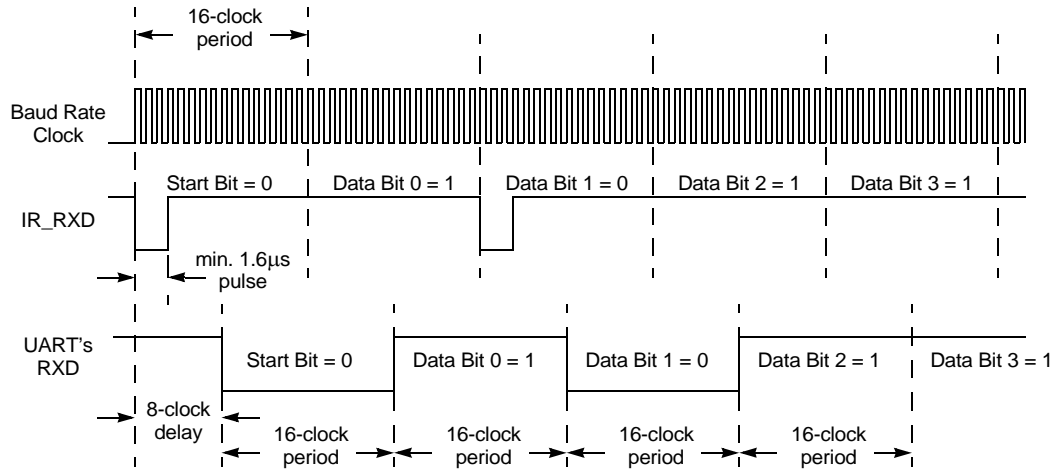


Figure 73. Infrared Data Reception

Jitter

Because of the inherent sampling of the received IR_RXD signal by the bit rate clock, some jitter can be expected on the first bit in any sequence of data. All subsequent bits in the received data stream are a fixed 16-clock periods wide.

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 86](#).



Caution:

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



mitter and receiver sections, a Baud Rate (clock) Generator and a control unit. The transmitter and receiver sections use the same clock.

During an SPI transfer, data is sent and received simultaneously by both the Master and the Slave SPI devices. Separate signals are required for data and the serial clock. When an SPI transfer occurs, a multi-bit (typically 8-bit) character is shifted out one data pin and a multi-bit character is simultaneously shifted in on a second data pin. An 8-bit shift register in the Master and another 8-bit shift register in the Slave are connected as a circular buffer. The SPI shift register is single-buffered in the transmit and receive directions. New data to be transmitted cannot be written into the shift register until the previous transmission is complete and receive data (if valid) has been read.

SPI Signals

The four basic SPI signals are:

- MISO (Master-In, Slave-Out)
- MOSI (Master-Out, Slave-In)
- SCK (SPI Serial Clock)
- \overline{SS} (Slave Select)

The following paragraphs discuss these SPI signals. Each signal is described in both Master and Slave modes.

Master-In, Slave-Out

The Master-In, Slave-Out (MISO) pin is configured as an input in a Master device and as an output in a Slave device. It is one of the two lines that transfer serial data, with the most significant bit sent first. The MISO pin of a Slave device is placed in a high-impedance state if the Slave is not selected. When the SPI is not enabled, this signal is in a high-impedance state.

Master-Out, Slave-In

The Master-Out, Slave-In (MOSI) pin is configured as an output in a Master device and as an input in a Slave device. It is one of the two lines that transfer serial data, with the most significant bit sent first. When the SPI is not enabled, this signal is in a high-impedance state.

Serial Clock

The Serial Clock (SCK) is used to synchronize data movement both in and out of the device through its MOSI and MISO pins. In Master mode, the SPI's Baud Rate Generator creates the serial clock. The Master drives the serial clock out its own SCK pin to the Slave's SCK pin. When the SPI is configured as a Slave, the SCK pin is an input and the clock signal from the Master synchronizes the data transfer between the Master and Slave devices. Slave devices ignore the SCK signal, unless the \overline{SS} pin is asserted.

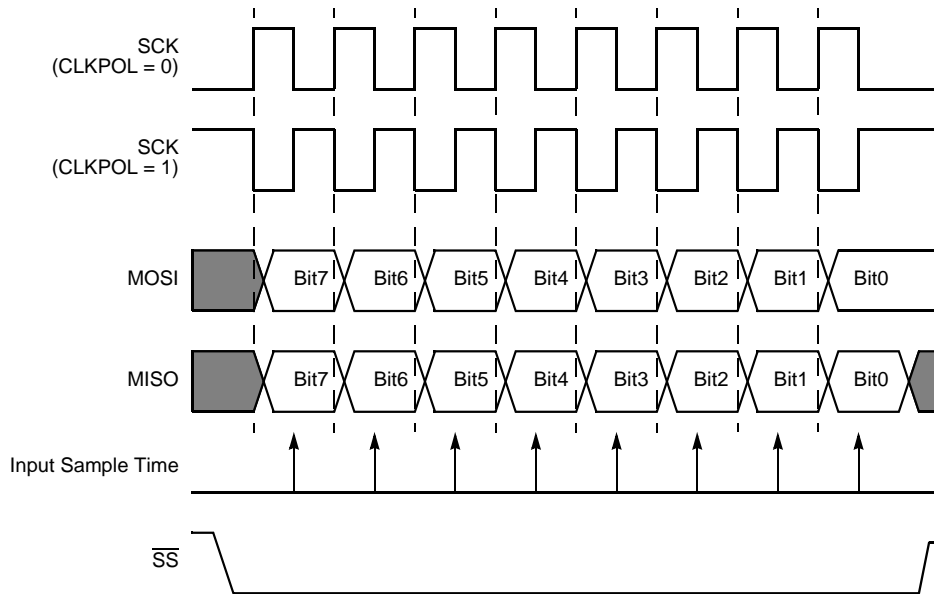


Figure 78. SPI Timing When PHASE is 1

Multi-Master Operation

In a multi-master SPI system, all SCK pins are tied together, all MOSI pins are tied together and all MISO pins are tied together. All SPI pins must then be configured in open-drain mode to prevent bus contention. At any one time, only one SPI device is configured as the Master and all other SPI devices on the bus are configured as Slaves. The Master enables a single Slave by asserting the \overline{SS} pin on that Slave only. Then, the single Master drives data out its SCK and MOSI pins to the SCK and MOSI pins on the Slaves (including those which are not enabled). The enabled Slave drives data out its MISO pin to the MISO Master pin.

For a Master device operating in a multi-master system, if the \overline{SS} pin is configured as an input and is driven Low by another Master, the COL bit is set to 1 in the SPI Status Register. The COL bit indicates the occurrence of a multi-master collision (mode fault error condition).

1. Disable the SPI by clearing the `SPIEN` bit in the SPI Control register to 0.
2. Load the desired 16-bit count value into the SPI 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 SPI Control register to 1.

SPI Control Register Definitions

SPI Data Register

The SPI Data register stores both the outgoing (transmit) data and the incoming (received) data. Reads from the SPI Data register always return the current contents of the 8-bit shift register.

With the SPI configured as a Master, writing a data byte to this register initiates the data transmission. With the SPI configured as a Slave, writing a data byte to this register loads the shift register in preparation for the next data transfer with the external Master. In either the Master or Slave modes, if a transmission is already in progress, writes to this register are ignored and the Overrun error flag, `OVR`, is set in the SPI Status register.

When the character length is less than 8 bits (as set by the `NUMBITS` field in the SPI Mode register), the transmit character must be left justified in the SPI Data register. A received character of less than 8 bits will be right justified. For example, if the SPI is configured for 4-bit characters, the transmit characters must be written to `SPIDATA[7:4]` and the received characters are read from `SPIDATA[3:0]`.

Table 60. SPI Data Register (SPIDATA)

BITS	7	6	5	4	3	2	1	0
FIELD	DATA							
RESET	X	X	X	X	X	X	X	X
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
ADDR	F60H							

DATA—Data
Transmit and/or receive data.



I²C Controller

Overview

The I²C Controller makes the Z8F640x family device bus-compatible with the I²C™ protocol. The I²C Controller consists of two bidirectional bus lines—a serial data signal (SDA) and a serial clock signal (SCL). Features of the I²C Controller include:

- Transmit and Receive Operation in Master mode
- Maximum data rate of 400kbit/sec
- 7- and 10-bit Addressing Modes for Slaves
- Unrestricted Number of Data Bytes Transmitted per Transfer

The I²C Controller in the Z8F640x family device does not operate in Slave mode.

Operation

The I²C Controller operates in Master mode to transmit and receive data. Only a single master is supported. Arbitration between two masters must be accomplished in software. I²C supports the following operations:

- Master transmits to a 7-bit slave
- Master transmits to a 10-bit slave
- Master receives from a 7-bit slave
- Master receives from a 10-bit slave

SDA and SCL Signals

I²C sends all addresses, data and acknowledge signals over the SDA line, most-significant bit first. SCL is the common clock for the I²C Controller. When the SDA and SCL pin alternate functions are selected for their respective GPIO ports, the pins are automatically configured for open-drain operation.

The master (I²C) is responsible for driving the SCL clock signal, although the clock signal can become skewed by a slow slave device. During the high period of the clock, the slave pulls the SCL signal Low to suspend the transaction. When the slave has released the line, the I²C Controller continues the transaction. All data is transferred in bytes and there is no

Table 74. DMAx Start/Current Address Low Byte Register (DMAxSTART)

BITS	7	6	5	4	3	2	1	0
FIELD	DMA_START							
RESET	X	X	X	X	X	X	X	X
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
ADDR	FB3H, FHBH							

DMA_START—DMAx Start/Current Address Low

These bits, with the four lower bits of the DMAx_H register, form the 12-bit Start/Current address. The full 12-bit address is given by {DMA_START_H[3:0], DMA_START[7:0]}.

DMAx End Address Low Byte Register

The DMAx End Address Low Byte register, in conjunction with the DMAx_H register, forms a 12-bit End Address.

Table 75. DMAx End Address Low Byte Register (DMAxEND)

BITS	7	6	5	4	3	2	1	0
FIELD	DMA_END							
RESET	X	X	X	X	X	X	X	X
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
ADDR	FB4H, FBCH							

DMA_END—DMAx End Address Low

These bits, with the four upper bits of the DMAx_H register, form a 12-bit address. This address is the ending location of the DMAx transfer. The full 12-bit address is given by {DMA_END_H[3:0], DMA_END[7:0]}.

DMA_ADC Address Register

The DMA_ADC Address register points to a block of the Register File to store ADC conversion values as illustrated in Table 76. This register contains the seven most-significant bits of the 12-bit Register File addresses. The five least-significant bits are calculated from the ADC Analog Input number (5-bit base address is equal to twice the ADC Analog Input number). The 10-bit ADC conversion data is stored as two bytes with the most significant byte of the ADC data stored at the even numbered Register File address.

Flash Control Register Definitions

Flash Control Register

The Flash Controller must be unlocked via 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, writing to the Flash Control register can initiate either Page Erase or Mass Erase of the Flash memory. 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 85. 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).



Option Bits

Overview

Option Bits allow user configuration of certain aspects of Z8F640x family device operation. The feature configuration data is stored in the Program Memory and read during Reset. The features available for control via the Option Bits are:

- Watch-Dog Timer time-out response selection—interrupt or Short 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.

Operation

Option Bit Configuration By Reset

Each time the Option Bits are programmed or erased, the Z8F640x family device must be Reset for the change to take place. During any reset operation (System Reset, Short Reset, or Stop Mode Recovery), the Option Bits are automatically read from the Program Memory and written to Option Configuration registers. The Option Configuration registers control operation of the Z8F640x family device. Option Bit control of the Z8F640x family device 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 Address Space

The first two bytes of Program Memory at addresses 0000H and 0001H are reserved for the user Option Bits. The byte at Program Memory address 0000H is used to configure user options. The byte at Program Memory address 0001H is reserved for future use and must be left in its unprogrammed state.

```
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 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 putting the Z8F640x family device back into 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 Z8F640x family device is not in Debug mode or if the 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 Z8F640x family device is not in Debug mode or if the 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 zero). If the Z8F640x family device is not in Debug mode, the address and data values are discarded. If the Read Protect Option Bit is enabled, then 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



Electrical Characteristics

Absolute Maximum Ratings

Stresses greater than those listed in Table 100 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, unused inputs must be tied to one of the supply voltages (V_{DD} or V_{SS}).

Table 100. Absolute Maximum Ratings

Parameter	Minimum	Maximum	Units	Notes
Ambient temperature under bias	-40	+105	C	
Storage temperature	-65	+150	C	
Voltage on any pin with respect to V_{SS}	-0.3	+5.5	V	1
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	μ A	
Maximum output current from active output pin	-25	+25	mA	
80-Pin QFP Maximum Ratings at -40°C to 70°C				
Total power dissipation		550	mW	
Maximum current into V_{DD} or out of V_{SS}		150	mA	
80-Pin QFP Maximum Ratings at 70°C to 105°C				
Total power dissipation		200	mW	
Maximum current into V_{DD} or out of V_{SS}		56	mA	
68-Pin PLCC Maximum Ratings at -40°C to 70°C				
Total power dissipation		1000	mW	
Maximum current into V_{DD} or out of V_{SS}		275	mA	

Notes:

1. This voltage applies to all pins except the following: V_{DD} , AV_{DD} , pins supporting analog input (Port B and Port H), RESET, and where noted otherwise.

Table 123. 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 124. 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 126. 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		
ADD dst, src	$\text{dst} \leftarrow \text{dst} + \text{src}$	r	r	02	*	*	*	*	0	*	2	3
		r	Ir	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	$\text{dst} \leftarrow \text{dst} + \text{src}$	ER	ER	08	*	*	*	*	0	*	4	3
		ER	IM	09							4	3
AND dst, src	$\text{dst} \leftarrow \text{dst} \text{ AND } \text{src}$	r	r	52	-	*	*	0	-	-	2	3
		r	Ir	53							2	4
		R	R	54							3	3
		R	IR	55							3	4
		R	IM	56							3	3
		IR	IM	57							3	4
ANDX dst, src	$\text{dst} \leftarrow \text{dst} \text{ AND } \text{src}$	ER	ER	58	-	*	*	0	-	-	4	3
		ER	IM	59							4	3
BCLR bit, dst	$\text{dst}[\text{bit}] \leftarrow 0$	r		E2	-	*	*	0	-	-	2	2
BIT p, bit, dst	$\text{dst}[\text{bit}] \leftarrow \text{p}$	r		E2	-	*	*	0	-	-	2	2
BRK	Debugger Break			00	-	-	-	-	-	-	1	1
BSET bit, dst	$\text{dst}[\text{bit}] \leftarrow 1$	r		E2	-	*	*	0	-	-	2	2
BSWAP dst	$\text{dst}[7:0] \leftarrow \text{dst}[0:7]$	R		D5	X	*	*	0	-	-	2	2
BTJ p, bit, src, dst	if $\text{src}[\text{bit}] = \text{p}$ $\text{PC} \leftarrow \text{PC} + \text{X}$	r		F6	-	-	-	-	-	-	3	3
		Ir		F7							3	4
BTJNZ bit, src, dst	if $\text{src}[\text{bit}] = 1$ $\text{PC} \leftarrow \text{PC} + \text{X}$	r		F6	-	-	-	-	-	-	3	3
		Ir		F7							3	4
Flags Notation:	* = Value is a function of the result of the operation. - = Unaffected X = Undefined				0 = Reset to 0 1 = Set to 1							