E·XFL

Zilog - Z8F6401AN020EC00TR Datasheet



Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

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

Applications of "<u>Embedded -</u> <u>Microcontrollers</u>"

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	64KB (64K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	4K 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	Surface Mount
Package / Case	44-LQFP
Supplier Device Package	44-LQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f6401an020ec00tr

Email: info@E-XFL.COM

Address: Room A, 16/F, Full Win Commercial Centre, 573 Nathan Road, Mongkok, Hong Kong



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concerning the devices, applications, or technology described is intended to suggest possible uses and may be superseded.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.



Low-Power Modes

Overview

The Z8F640x family products contain power-saving features. The highest level of power reduction is provided by Stop mode. The next level of power reduction is provided by the Halt mode.

Stop Mode

Execution of the eZ8 CPU's STOP instruction places the Z8F640x family device into Stop mode. In Stop mode, the operating characteristics are:

- Primary crystal 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, the Watch-Dog Timer continues to operate
- 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 Z8F640x family device can be brought out of Stop mode using Stop Mode Recovery. For more information on STOP Mode Recovery refer to the **Reset and Stop Mode Recovery** chapter.

Halt Mode

Execution of the eZ8 CPU's HALT instruction places the Z8F640x family device into Halt mode. In Halt mode, the operating characteristics are:

- Primary crystal oscillator is enabled and continues to operate
- System clock is enabled and continues to operate
- eZ8 CPU is idle
- Program counter (PC) stops incrementing



of the port pin direction (input/output) is passed from the Port A-H Data Direction registers to the alternate function assigned to this pin. Table 11 lists the alternate functions associated with each port pin.

Pin	Mnemonic	Alternate Function Description
PA0	T0IN	Timer 0 Input
PA1	TOOUT	Timer 0 Output
PA2	N/A	No alternate function
PA3	CTS0	UART 0 Clear to Send
PA4	RXD0 / IRRX0	UART 0 / IrDA 0 Receive Data
PA5	TXD0 / IRTX0	UART 0 / IrDA 0 Transmit Data
PA6	SCL	I ² C Clock (automatically open-drain)
PA7	SDA	I ² C Data (automatically open-drain)
PB0	ANA0	ADC Analog Input 0
PB1	ANA1	ADC Analog Input 1
PB2	ANA2	ADC Analog Input 2
PB3	ANA3	ADC Analog Input 3
PB4	ANA4	ADC Analog Input 4
PB5	ANA5	ADC Analog Input 5
PB6	ANA6	ADC Analog Input 6
PB7	ANA7	ADC Analog Input 7
PC0	T1IN	Timer 1 Input
PC1	T1OUT	Timer 1 Output
PC2	SS	SPI Slave Select
PC3	SCK	SPI Serial Clock
PC4	MOSI	SPI Master Out Slave In
PC5	MISO	SPI Master In Slave Out
PC6	T2IN	Timer 2 In
PC7	T2OUT	Timer 2 Out (not available in 40-pin packages)
	Pin PA0 PA1 PA2 PA3 PA4 PA5 PA6 PA7 PB0 PB1 PB2 PB3 PB6 PB7 PC0 PC1 PC2 PC3 PC4 PC5 PC6 PC7	PinMnemonicPA0TOINPA1TOOUTPA2N/APA3CTS0PA4RXD0 / IRRX0PA5TXD0 / IRTX0PA6SCLPA7SDAPB0ANA0PB1ANA1PB2ANA2PB3ANA3PB4ANA4PB5ANA5PB6ANA6PB7ANA7PC0T1INPC1T1OUTPC2SSPC3SCKPC4MOSIPC5MISOPC6T2INPC7T2OUT

Table 11	. Port	Alternate	Function	Mapping
----------	--------	-----------	----------	---------



I²CI—I²C Interrupt Request

0 = No interrupt request is pending for the I²C.

1 = An interrupt request from the I²C is awaiting service.

SPII-SPI Interrupt Request

- 0 = No interrupt request is pending for the SPI.
- 1 = An interrupt request from the SPI 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 24) 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.

able 24. li	. Interrupt Request 1 Register (IRQ1)							
BITS	7	6	5	4	3	2		
FIELD	PAD7I	PAD6I	PAD5I	PAD4I	PAD3I	PAD2I		
RESET	0	0	0	0	0	0		

Тź

R/W

R/W

R/W

ADDR

PADxI—Port A or Port D Pin x Interrupt Request

R/W

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

R/W

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

FC3H

where x indicates the specific GPIO Port pin number (0 through 7). For each pin, only 1 of either Port A or Port D can be enabled for interrupts at any one time. Port selection (A or D) is determined by the values in the Interrupt Port Select Register.

R/W

R/W

1

PAD1I

0

R/W

0

PAD01

0

R/W



Interrupt Request 2 Register

The Interrupt Request 2 (IRQ2) register (Table 25) 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 1 register to determine if any interrupt requests are pending.

Table 25. Interrupt Request 2 Register (IRQ2)

BITS	7	6	5	4	3	2	1	0
FIELD	T3I	U1RXI	UITXI	DMAI	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						

T3I—Timer 3 Interrupt Request

- 0 = No interrupt request is pending for Timer 3.
- 1 = An interrupt request from Timer 3 is awaiting service.

U1RXI—UART 1 Receive Interrupt Request

- 0 = No interrupt request is pending for the UART1 receiver.
- 1 = An interrupt request from UART1 receiver is awaiting service.
- U1TXI-UART 1 Transmit Interrupt Request
- 0 = No interrupt request is pending for the UART 1 transmitter.
- 1 = An interrupt request from the UART 1 transmitter is awaiting service.

DMAI—DMA Interrupt Request

- 0 = No interrupt request is pending for the DMA.
- 1 = An interrupt request from the DMA is awaiting service.

PCxI—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 through 3).



Watch-Dog Timer

Overview

The Watch-Dog Timer (WDT) helps protect against corrupt or unreliable software, power faults, and other system-level problems which may place the Z8 Encore![®] into unsuitable operating states. The Watch-Dog Timer includes the following features:

- On-chip RC oscillator
- A selectable time-out response: Short 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 Z8F640x family device when the WDT reaches its terminal count. The Watch-Dog Timer uses its own dedicated on-chip RC oscillator as its clock source. The Watch-Dog Timer has only two modes of operation—on and off. Once enabled, it always counts and must be refreshed to prevent a time-out. An enable can be performed by executing the WDT instruction or by setting the WDT_AO Option Bit. The WDT_AO bit enables the Watch-Dog Timer to operate all the time, 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 given by the following equation:

WDT Time-out Period (ms) = $\frac{\text{WDT Reload Value}}{50}$

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 50kHz. The Watch-Dog Timer cannot be refreshed once it reaches 000002H. The WDT Reload Value must not be set to values below 000004H. Table 45 provides



Reserved These bits are reserved and must be 0.

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

The Watch-Dog Timer Reload Upper, High and Low Byte (WDTU, WDTH, WDTL) registers (Tables 47 through 49) 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], WDTH[7:0], WDTL[7:0]. Writing to these registers sets the desired Reload Value. Reading from these registers returns the current Watch-Dog Timer count value.



The 24-bit WDT Reload Value must not be set to a value less than 000004H or unpredictable behavior may result.

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

Table 47. Watch-Dog Timer Reload Upper Byte Register (WDTU)

WDTU-WDT Reload Upper Byte

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

Table 48. 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*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*
ADDR	ADDR FF2H							
R/W* - Read returns the current WDT count value. Write sets the desired Reload Value.								

WDTH—WDT Reload High Byte



92

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 and F4FH								

Table 57. UARTx Baud Rate Low Byte Register (UxBRL)

The UART data rate is calculated using the following equation:

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, the integer baud rate divisor value is calculated using the following equation:

UART Baud Rate Divisor Value (BRG) = Round (System Clock Frequency (Hz)) 16 × UART Data Rate (bits/s)

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

For reliable communication, the UART baud rate error must never exceed 5 percent. Table 58 provides information on data rate errors for popular baud rates and commonly used crystal oscillator frequencies.



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 an 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 highimpedance 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.

132

ZILOG

Analog-to-Digital Converter

Overview

The Analog-to-Digital Converter (ADC) converts an analog input signal to a 10-bit binary number. The features of the sigma-delta ADC include:

- 12 analog input sources are multiplexed with general-purpose I/O ports
- Interrupt upon conversion complete
- Internal voltage reference generator
- Direct Memory Access (DMA) controller can automatically initiate data conversion and transfer of the data from 1 to 12 of the analog inputs.

Architecture

Figure 83 illustrates the three major functional blocks (converter, analog multiplexer, and voltage reference generator) of the ADC. The ADC converts an analog input signal to its digital representation. The 12-input analog multiplexer selects one of the 12 analog input sources. The ADC requires an input reference voltage for the conversion. The voltage reference for the conversion may be input through the external VREF pin or generated internally by the voltage reference generator.





Figure 84. Flash Memory Arrangement

Operation

The Flash Controller programs and erases the Flash memory. The Flash Controller provides the proper Flash controls and timing for byte programming, Page Erase, and Mass Erase of the Flash memory. The Flash Controller contains a protection mechanism, via the Flash Control register (FCTL) to prevent accidental programming or erasure. The Flow Chart in Figure 85 illustrates basic Flash Controller operation. The following subsections provide details on the various operations (Lock, Unlock, Byte Programming, Page Erase, and Mass Erase) listed in Figure 85.



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. With the Flash Controller unlocked, 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 through 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. Mass Erasing the Flash memory sets all bytes to the value FFH. With the Flash Controller unlocked, 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. Typically, the Flash Memory is Mass Erased using the On-Chip Debugger. Via the On-Chip Debugger, poll the Flash Status register to determine when the Mass Erase operation is complete. Although the Flash can be Mass Erased by user program code, when the Mass Erase is complete the user program code is completely erased. 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 programing 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*!TM for more information on bypassing the Flash Controller. This document is available for download at <u>www.zilog.com</u>.



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:.

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

Caution: Flash programming and erasure is not supported for system clock frequencies below 32KHz (32768Hz) or above 20MHz. The Flash Frequency High and Low Byte registers must be loaded with the correct value to insure proper operation of the Z8F640x family device.

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						
ADDR		FFAH						

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

Table 89. Flash Frequency Low Byte Register (FFREQL)

BITS	7	6	5	4	3	2	1	0
FIELD	FFREQL							
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		FFBH						

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





Figure 90. Recommended Crystal Oscillator Configuration (20MHz operation)

Parameter	Value	Units	Comments
Frequency	20	MHz	
Resonance	Parallel		
Mode	Fundamental		
Series Resistance (R _S)	25	Ω	Maximum
Load Capacitance (CL)	20	pF	Maximum
Shunt Capacitance (C ₀)	7	pF	Maximum
Drive Level	1	mW	Maximum

Table 99. Recommended Crystal Oscillator Specifications (20MHz Operation)



		$V_{DD} = 3.0 - 3.6V$ $T_A = -40^{\circ}C \text{ to } 105^{\circ}C$				
Symbol	Parameter	Minimum	Typical	Maximum	Units	Conditions
	DC Offset Error	-50	_	25	mV	40-pin PDIP, 44-pin LQFP, 44-pin PLCC, and 68-pin PLCC packages.
V _{REF}	Internal Reference Voltage	-	2.0	-	V	
	Single-Shot Conversion Time	_	5129	_	cycles	System clock cycles
	Continuous Conversion Time	_	256	-	cycles	System clock cycles
	Sampling Rate	Syst	System Clock / 256		Hz	
	Signal Input Bandwidth	-	-	3.5	kHz	
R _S	Analog Source Impedance	-	-	10 ¹	kΩ	
Zin	Input Impedance		150		kΩ	20MHz system clock. Input impedance increases with lower system clock frequency.
V _{REF}	External Reference Voltage			AVDD	V	AVDD <= VDD. When using an external reference voltage, decoupling capacitance should be placed from VREF to AVSS.

Table 106. Analog-to-Digital Converter Electrical Characteristics and Timing (Continued)

¹ Analog source impedance affects the ADC offset voltage (because of pin leakage) and input settling time.



184

Notation	Description	Operand	Range	
b	Bit	b	b represents a value from 0 to 7 (000B to 111B).	
сс	Condition Code	_	See Condition Codes overview in the eZ8 CPU User Manual.	
DA	Direct Address	Addrs	Addrs. 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	
р	Polarity	р	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	Х	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 115. Notational Shorthand

Table 116 contains additional symbols that are used throughout the Instruction Summary and Instruction Set Description sections.



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
STOP	_	Stop Mode
WDT	_	Watch-Dog Timer Refresh

Table 121. CPU Control Instructions

Table 122. 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



Opcode Maps

Figures 101 and 102 provide information on each of the eZ8 CPU instructions. A description of the opcode map data and the abbreviations are provided in Figure 100 and Table 127.



Figure 100. Opcode Map Cell Description



controller 111 controller signals 13 interrupts 112 operation 111 SDA and SCL signals 111 stop and start conditions 112 I2CBRH register 121 I2CBRL register 121 I2CCTL register 119 I2CDATA register 118 I2CSTAT register 118 IM 184 immediate data 184 immediate operand prefix 185 **INC 187** increment 187 increment word 187 **INCW 187** indexed 184 indirect address prefix 185 indirect register 184 indirect register pair 184 indirect working register 184 indirect working register pair 184 infrared encoder/decoder (IrDA) 95 instruction set, ez8 CPU 182 instructions ADC 187 **ADCX 187** ADD 187 **ADDX 187** AND 190 **ANDX 190** arithmetic 187 **BCLR 188 BIT 188** bit manipulation 188 block transfer 188 **BRK 190 BSET 188** BSWAP 188, 191 **BTJ 190** BTJNZ 190 **BTJZ 190**

CALL 190 CCF 188, 189 **CLR 189** COM 190 CP 187 CPC 187 **CPCX 187** CPU control 189 CPX 187 DA 187 **DEC 187 DECW 187** DI 189 **DJNZ 190** EI 189 **HALT 189 INC 187 INCW 187 IRET 190** JP 190 LD 189 LDC 189 LDCI 188, 189 LDE 189 LDEI 188 LDX 189 LEA 189 load 189 logical 190 **MULT 187** NOP 189 OR 190 ORX 190 POP 189 **POPX 189** program control 190 **PUSH 189** PUSHX 189 RCF 188, 189 **RET 190** RL 191 RLC 191 rotate and shift 191 RR 191



SDA and SCL (IrDA) signals 111 second opcode map after 1FH 205 serial clock 101 serial peripheral interface (SPI) 99 set carry flag 188, 189 set register pointer 189 shift right arithmetic 191 shift right logical 191 signal descriptions 13 single assertion (pulse) interrupt sources 47 single-shot conversion (ADC) 133 SIO 5 slave data transfer formats (I2C) 114 slave select 102 software trap 190 source operand 185 SP 185 SPI architecture 99 baud rate generator 105 baud rate high and low byte register 110 clock phase 102 configured as slave 100 control register 107 control register definitions 106 data register 106 error detection 105 interrupts 105 mode fault error 105 mode register 109 multi-master operation 104 operation 100 overrun error 105 signals 101 single master, multiple slave system 100 single master, single slave system 99 status register 108 timing, PHASE = 0.103timing, PHASE=1 104 SPI controller signals 13 SPI mode (SPIMODE) 109 SPIBRH register 110 SPIBRL register 110 SPICTL register 107

SPIDATA register 106 SPIMODE register 109 SPISTAT register 108 SRA 191 src 185 SRL 191 **SRP 189** SS, SPI signal 101 stack pointer 185 status register, I2C 118 **STOP 189** stop mode 31, 189 stop mode recovery sources 29 using a GPIO port pin transition 30 using watch-dog timer time-out 29 **SUB 188** subtract 188 subtract - extended addressing 188 subtract with carry 188 subtract with carry - extended addressing 188 **SUBX 188 SWAP 191** swap nibbles 191 symbols, additional 185 system and short resets 26

Т

TCM 188 TCMX 188 technical support 213 test complement under mask 188 test under mask 188 timer signals 14 timers 5, 57 architecture 57 block diagram 58 capture mode 62, 71 compare mode 63, 71 compare mode 63, 71 continuous mode 59, 70 counter mode 60 counter modes 70