



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	Active
Core Processor	eZ8
Core Size	8-Bit
Speed	20MHz
Connectivity	I <sup>2</sup> C, IrDA, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	46
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 12x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	64-LQFP
Supplier Device Package	-
Purchase URL	<a href="https://www.e-xfl.com/product-detail/zilog/z8f2422ar020eg">https://www.e-xfl.com/product-detail/zilog/z8f2422ar020eg</a>

Table 70.	SPI Baud Rate Low Byte Register (SPIBRL) . . . . .	127
Table 71.	I <sup>2</sup> C Data Register (I2CDATA) . . . . .	142
Table 72.	I2C Status Register (I2CSTAT) . . . . .	142
Table 73.	I2C Control Register (I2CCTL) . . . . .	144
Table 74.	I <sup>2</sup> C Baud Rate Low Byte Register (I2CBRL) . . . . .	146
Table 75.	I <sup>2</sup> C Baud Rate High Byte Register (I2CBRH) . . . . .	146
Table 76.	I <sup>2</sup> C Diagnostic State Register (I2CDST) . . . . .	147
Table 77.	I <sup>2</sup> C Diagnostic Control Register (I2CDIAG) . . . . .	149
Table 78.	DMAx Control Register (DMAxCTL) . . . . .	153
Table 79.	DMAx I/O Address Register (DMAxIO) . . . . .	154
Table 80.	DMAx Address High Nibble Register (DMAxH) . . . . .	155
Table 81.	DMAx Start/Current Address Low Byte Register (DMAxSTART) . . . . .	156
Table 82.	DMAx End Address Low Byte Register (DMAxEND) . . . . .	156
Table 83.	DMA_ADC Register File Address Example . . . . .	157
Table 84.	DMA_ADC Control Register (DMAACTL) . . . . .	158
Table 85.	DMA_ADC Address Register (DMAA_ADDR) . . . . .	158
Table 86.	DMA_ADC Status Register (DMAA_STAT) . . . . .	159
Table 87.	ADC Control Register (ADCCTL) . . . . .	165
Table 88.	ADC Data High Byte Register (ADCD_H) . . . . .	167
Table 89.	ADC Data Low Bits Register (ADCD_L) . . . . .	168
Table 90.	Flash Memory Configurations . . . . .	169
Table 91.	Flash Memory Sector Addresses . . . . .	169
Table 92.	Z8 Encore! XP F64xx Series Information Area Map . . . . .	171
Table 93.	Flash Control Register (FCTL) . . . . .	176
Table 94.	Flash Status Register (FSTAT) . . . . .	177
Table 95.	Flash Sector Protect Register (FPROT) . . . . .	178
Table 96.	Page Select Register (FPS) . . . . .	178
Table 97.	Flash Frequency High Byte Register (FFREQH) . . . . .	179
Table 98.	Flash Frequency Low Byte Register (FFREQL) . . . . .	179
Table 99.	Flash Option Bits At Flash Memory Address 0000H . . . . .	181
Table 100.	Options Bits at Flash Memory Address 0001H . . . . .	182
Table 101.	OCD Baud-Rate Limits . . . . .	186
Table 102.	On-Chip Debugger Commands . . . . .	189
Table 103.	OCD Control Register (OCDCTL) . . . . .	193
Table 104.	OCD Status Register (OCDSTAT) . . . . .	194
Table 105.	Recommended Crystal Oscillator Specifications (20MHz Operation) . . . . .	197

Table 6. Z8 Encore! XP F64xx Series Information Area Map

Program Memory Address (Hex)	Function
FE00H–FE3FH	Reserved
FE40H–FE53H	Part Number 20-character ASCII alphanumeric code Left-justified and filled with zeros (ASCII Null character)
FE54H–FFFFH	Reserved

**Table 10. Stop Mode Recovery Sources and Resulting Action**

Operating Mode	Stop Mode Recovery Source	Action
STOP Mode	Watchdog Timer time-out when configured for Reset.	Stop Mode Recovery.
	Watchdog 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.

## Stop Mode Recovery Using Watchdog Timer Time-Out

If the Watchdog Timer times out during STOP Mode, the device undergoes a Stop Mode Recovery sequence. In the Watchdog Timer Control Register, the WDT and stop bits are set to 1. If the Watchdog Timer is configured to generate an interrupt upon time-out and the Z8 Encore! XP F64xx Series devices are configured to respond to interrupts, the eZ8 CPU services the Watchdog Timer interrupt request following the normal Stop Mode Recovery sequence.

## Stop Mode Recovery Using a GPIO Port Pin Transition HALT

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. The GPIO Stop Mode Recovery signals are filtered to reject pulses less than 10 ns (typical) in duration. In the Watchdog 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. Thus, 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).

### Port A–H Alternate Function Subregisters

The Port A–H Alternate Function Subregister, shown in Table 17, is accessed through the Port A–H Control Register by writing 02H to the Port A–H Address Register. The Port A–H Alternate Function subregisters select the alternate functions for the selected pins. To determine the alternate function associated with each port pin, see the [GPIO Alternate Functions](#) section on page 37.

---

**! Caution:** Do not enable alternate function for GPIO port pins which do not have an associated alternate function. Failure to follow this guideline may result in unpredictable operation.

---

**Table 17. Port A–H Alternate Function Subregisters**

Bit	7	6	5	4	3	2	1	0
Field	AF7	AF6	AF5	AF4	AF3	AF2	AF1	AF0
RESET	0							
R/W	R/W							
Address	See note.							
Note: If a 02H exists in the Port A–H Address Register, it is accessible through the Port A–H Control Register.								

Bit	Description
[7:0]	<b>Port Alternate Function Enabled</b>
AFx	0 = The port pin is in NORMAL Mode and the DDx bit in the Port A–H Data Direction Subregister determines the direction of the pin. 1 = The alternate function is selected. Port pin operation is controlled by the alternate function.

Note: x indicates register bits in the range [7:0].

**Port A–H Stop Mode Recovery Source Enable Subregisters**

The Port A–H Stop Mode Recovery Source Enable Subregister, shown in Table 20, is accessed through the Port A–H Control Register by writing 05H to the Port A–H Address Register. Setting the bits in the Port A–H Stop Mode Recovery Source Enable subregisters 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 20. Port A–H Stop Mode Recovery Source Enable Subregisters**

Bit	7	6	5	4	3	2	1	0
Field	PSMRE7	PSMRE6	PSMRE5	PSMRE4	PSMRE3	PSMRE2	PSMRE1	PSMRE0
RESET	0							
R/W	R/W							
Address	See note.							
Note: If a 05H exists in the Port A–H Address Register, it is accessible through the Port A–H Control Register.								

Bit	Description
[7:0]	<b>Port Stop Mode Recovery Source Enabled</b>
PSMRE	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.

Note: x indicates register bits in the range [7:0].

it is appropriate to have the timer output make a permanent state change upon a One-Shot time-out, first set the TPOL bit in the Timer Control 1 Register to the start value before beginning ONE-SHOT Mode. Then, after starting the timer, set TPOL to the opposite bit value.

Observe the following procedure for configuring a timer for ONE-SHOT Mode and initiating the count:

1. Write to the Timer Control 1 Register to:
  - Disable the timer
  - Configure the timer for ONE-SHOT 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.
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 1 Register to enable the timer and initiate counting.

In ONE-SHOT Mode, the system clock always provides the timer input. The timer period is calculated using 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) upon timer reload.

Observe the following procedure for configuring a timer for CONTINUOUS Mode and initiating the count:

1. Write to the Timer Control 1 Register to:

If the TPOL bit in the Timer Control 1 Register is set to 1, the timer output signal begins as a High (1) and then 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 1 Register is set to 0, the timer output signal begins as a Low (0) and then 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.

Observe the following procedure for configuring a timer for PWM Mode and initiating the PWM operation:

1. Write to the Timer Control 1 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 1 Register to enable the timer and initiate counting.

The PWM period is calculated using 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 must be used to determine the first PWM time-out period.



## Timer 0–3 Control 0 Registers

The Timer 0–3 Control 0 (TxCTL0) registers, shown in Tables 45 and 46, allow cascading of the timers.

**Table 45. Timer 0–3 Control 0 Register (TxCTL0)**

Bit	7	6	5	4	3	2	1	0
Field	Reserved			CSC	Reserved			
RESET	0							
R/W	R/W							
Address	F06H, F0EH, F16H, F1EH							

Bit	Description
[7:5]	<b>Reserved</b> These bits are reserved and must be programmed to 000.
[4] CSC	<b>Cascade Timers</b> 0 = Timer input signal comes from the pin. 1 = For Timer 0, the input signal is connected to Timer 3 output. For Timer 1, the input signal is connected to the Timer 0 output. For Timer 2, the input signal is connected to the Timer 1 output. For Timer 3, the input signal is connected to the Timer 2 output.
[3:0]	<b>Reserved</b> These bits are reserved and must be programmed to 0000.

6. 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].
7. Return to Step 5 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). Observe the following procedure to configure the UART receiver for interrupt-driven operation:

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 Receiver interrupt and set the appropriate 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 appropriate 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! XP 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.

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, since it does not need to access the UART when it receives data directed to other devices on the multinode 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]. At this point, each new incoming byte interrupts the CPU. The software is then responsible for determining the end of the frame. It checks for 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, then set MPMD[0] 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 is enabled by setting MPMD[1:0] to 10b and writing the UART's address into the UART Address Compare Register. This mode introduces more 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. The first data byte in the frame contains the NEWFRM=1 in the UART Status 1 Register. When the next address byte occurs, the hardware compares it to the UART's address. If there is a match, the interrupts continue and the NEWFRM bit is set for the first byte of the new frame. If there is no match, then the UART ignores all incoming bytes until the next address match.

S	Slave Address	W = 0	A	Data	A	Data	A	Data	A/A	P/S
---	---------------	-------	---	------	---	------	---	------	-----	-----

**Figure 29. 7-Bit Addressed Slave Data Transfer Format**

Observe the following procedure for a transmit operation to a 7-bit addressed slave:

1. Software asserts the IEN bit in the I<sup>2</sup>C Control Register.
2. Software asserts the TXI bit of the I<sup>2</sup>C Control Register to enable transmit interrupts.
3. The I<sup>2</sup>C interrupt asserts, because the I<sup>2</sup>C Data Register is empty
4. Software responds to the TDRE bit by writing a 7-bit slave address plus write bit (=0) to the I<sup>2</sup>C Data Register.
5. Software asserts the start bit of the I<sup>2</sup>C Control Register.
6. The I<sup>2</sup>C Controller sends the start condition to the I<sup>2</sup>C slave.
7. The I<sup>2</sup>C Controller loads the I<sup>2</sup>C Shift Register with the contents of the I<sup>2</sup>C Data Register.
8. After one bit of address has been shifted out by the SDA signal, the transmit interrupt is asserted (TDRE = 1).
9. Software responds by writing the transmit data into the I<sup>2</sup>C Data Register.
10. The I<sup>2</sup>C Controller shifts the rest of the address and write bit out by the SDA signal.
11. If the I<sup>2</sup>C slave sends an acknowledge (by pulling the SDA signal Low) during the next High period of SCL the I<sup>2</sup>C Controller sets the ACK bit in the I<sup>2</sup>C Status Register. Continue with [Step 12](#).

If the slave does not acknowledge, the Not Acknowledge interrupt occurs (NCKI bit is set in the Status Register, ACK bit is cleared). Software responds to the Not Acknowledge interrupt by setting the stop and flush bits and clearing the TXI bit. The I<sup>2</sup>C Controller sends the stop condition on the bus and clears the stop and NCKI bits. The transaction is complete (ignore the following steps).

12. The I<sup>2</sup>C Controller loads the contents of the I<sup>2</sup>C Shift Register with the contents of the I<sup>2</sup>C Data Register.
13. The I<sup>2</sup>C Controller shifts the data out of using the SDA signal. After the first bit is sent, the transmit interrupt is asserted.
14. If more bytes remain to be sent, return to [Step 9](#).
15. Software responds by setting the stop bit of the I<sup>2</sup>C Control Register (or start bit to initiate a new transaction). In the stop case, software clears the TXI bit of the I<sup>2</sup>C Control Register at the same time.

Bit	Description (Continued)
[1] FLUSH	<b>Flush Data</b> Setting this bit to 1 clears the I <sup>2</sup> C Data Register and sets the TDRE bit to 1. This bit allows flushing of the I <sup>2</sup> C Data Register when a Not Acknowledge interrupt is received after the data has been sent to the I <sup>2</sup> C Data Register. Reading this bit always returns 0.
[0] FILTEN	<b>I<sup>2</sup>C Signal Filter Enable</b> This bit enables low-pass digital filters on the SDA and SCL input signals. These filters reject any input pulse with periods less than a full system clock cycle. The filters introduce a 3-system clock cycle latency on the inputs. 1 = low-pass filters are enabled. 0 = low-pass filters are disabled.

## I<sup>2</sup>C Baud Rate High and Low Byte Registers

The I<sup>2</sup>C Baud Rate High and Low Byte registers, shown in Tables 74 and 75, combine to form a 16-bit reload value, BRG[15:0], for the I<sup>2</sup>C Baud Rate Generator.

When the I<sup>2</sup>C is disabled, the Baud Rate Generator can function 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 I<sup>2</sup>C by clearing the IEN bit in the I<sup>2</sup>C Control Register to 0.
2. Load the appropriate 16-bit count value into the I<sup>2</sup>C 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 I<sup>2</sup>C Control Register to 1.

When configured as a general purpose timer, the interrupt interval is calculated using the following equation:

$$\text{Interrupt Interval (s)} = \text{System Clock Period (s)} \times \text{BRG}[15:0]$$

## DMAx Control Register

The DMAx Control Register, shown in Table 78, enables and selects the mode of operation for DMAx.

**Table 78. DMAx Control Register (DMAxCTL)**

Bit	7	6	5	4	3	2	1	0
Field	DEN	DLE	DDIR	IRQEN	WSEL	RSS		
RESET	0							
R/W	R/W							
Address	FB0H, FB8H							

Bit	Description
[7] DEN	<b>DMAx Enable</b> 0 = DMAx is disabled and data transfer requests are disregarded. 1 = DMAx is enabled and initiates a data transfer upon receipt of a request from the trigger source.
[6] DLE	<b>DMAx Loop Enable</b> 0 = DMAx reloads the original Start Address and is then disabled after the End Address data is transferred. 1 = DMAx, after the End Address data is transferred, reloads the original Start Address and continues operating.
[5] DDIR	<b>DMAx Data Transfer Direction</b> 0 = Register File → on-chip peripheral control register. 1 = On-chip peripheral control → Register File.
[4] IRQEN	<b>DMAx Interrupt Enable</b> 0 = DMAx does not generate any interrupts. 1 = DMAx generates an interrupt when the End Address data is transferred.

Bit	Description (Continued)
[4] BRKLOOP	<b>Breakpoint Loop</b> This bit determines what action the OCD takes when a BRK instruction is decoded if breakpoints are enabled (BRKEN is 1). If this bit is 0, then the DBGMODE bit is automatically set to 1 and the OCD entered DEBUG Mode. If BRKLOOP is set to 1, then the eZ8 CPU loops on the BRK instruction. 0 = BRK instruction sets DBGMODE to 1. 1 = eZ8 CPU loops on BRK instruction.
[3:1]	<b>Reserved</b> These bits are reserved and must be programmed to 000.
[0] RST	<b>Reset</b> Setting this bit to 1 resets the Z8 Encore! XP F64xx Series devices. The devices go 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 when the reset finishes. 0 = No effect. 1 = Reset the Z8 Encore! XP F64xx Series device.

## OCD Status Register

The OCD Status Register, shown in Table 104, reports status information about the current state of the debugger and the system.

**Table 104. OCD Status Register (OCDSTAT)**

Bit	7	6	5	4	3	2	1	0
Field	IDLE	HALT	RPEN	Reserved				
RESET	0							
R/W	R							

Bit	Description
[7] IDLE	<b>CPU Idle</b> This bit is set if the part is in DEBUG Mode (DBGMODE is 1), or if a BRK instruction occurred since the last time OCDCTL was written. This can be used to determine if the CPU is running or if it is idling. 0 = The eZ8 CPU is running. 1 = The eZ8 CPU is either stopped or looping on a BRK instruction.
[6] HALT	<b>HALT Mode</b> 0 = The device is not in HALT Mode. 1 = The device is in HALT Mode.

<b>Bit</b>	<b>Description (Continued)</b>
[5] RPN	<b>Read Protect Option Bit Enabled</b> 0 = The Read Protect option bit is disabled (1). 1 = The Read Protect option bit is enabled (0), disabling many OCD commands.
[4:0]	<b>Reserved</b> These bits are reserved and must be programmed to 00000.



## 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 128 through 135 contain the instructions belonging to each group and the number of operands required for each instruction. Some instructions appear in more than one table; these instructions can be considered to be 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 128. 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

**Table 133. Logical Instructions**

<b>Mnemonic</b>	<b>Operands</b>	<b>Instruction</b>
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 134. Program Control Instructions**

<b>Mnemonic</b>	<b>Operands</b>	<b>Instruction</b>
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

**Hex Address: FC5**

**Table 222. IRQ1 Enable Low Bit Register (IRQ1ENL)**

Bit	7	6	5	4	3	2	1	0
Field	PAD7ENL	PAD6ENL	PAD5ENL	PAD4ENL	PAD3ENL	PAD2ENL	PAD1ENL	PAD0ENL
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
Address	FC5H							

**Hex Address: FC6**

**Table 223. Interrupt Request 2 Register (IRQ2)**

Bit	7	6	5	4	3	2	1	0
Field	T3I	U1RXI	U1TXI	DMAI	PC3I	PC2I	PC1I	PC0I
RESET	0							
R/W	R/W							
Address	FC6H							

**Hex Address: FC7**

**Table 224. IRQ2 Enable High Bit Register (IRQ2ENH)**

Bit	7	6	5	4	3	2	1	0
Field	T3ENH	U1RENH	U1TENH	DMAENH	C3ENH	C2ENH	C1ENH	C0ENH
RESET	0							
R/W	R/W							
Address	FC7H							

**Hex Address: FC8**

**Table 225. IRQ2 Enable Low Bit Register (IRQ2ENL)**

Bit	7	6	5	4	3	2	1	0
Field	T3ENL	U1RENL	U1TENL	DMAENL	C3ENL	C2ENL	C1ENL	C0ENL
RESET	0							
R/W	R/W							
Address	FC8H							

- compare with carry 231
- compare with carry - extended addressing 231
- complement 234
- complement carry flag 232, 233
- condition code 228
- continuous conversion (ADC) 165
- continuous mode 79
- control register definition, UART 99
- control register, I2C 145
- counter modes 79
- CP 231
- CPC 231
- CPCX 231
- CPU and peripheral overview 4
- CPU control instructions 233
- CPX 231
- Customer Feedback Form 305
- customer feedback form 294
- Customer Information 305

## **D**

- DA 228, 231
- data register, I2C 142
- DC characteristics 203
- debugger, on-chip 184
- DEC 231
- decimal adjust 231
- decrement 231
- decrement and jump non-zero 234
- decrement word 231
- DECW 231
- destination operand 229
- device, port availability 37
- DI 233
- direct address 228
- direct memory access controller 151
- disable interrupts 233
- DJNZ 234
- DMA
  - address high nibble register 156
  - configuring DMA0-1 data transfer 151
  - configuring for DMA\_ADC data transfer 153
  - control of ADC 166

- control register 154
- control register definitions 153
- controller 6
- DMA\_ADC address register 158
- DMA\_ADC control register 159
- DMA\_ADC operation 152
- end address low byte register 157
- I/O address register 155
- operation 151
- start/current address low byte register 157
- status register 160
- DMAA\_STAT register 160
- DMAACTL register 159
- DMAxCTL register 154, 268, 269
- DMAxEND register 157, 269, 270
- DMAxH register 156, 268, 270
- DMAxI/O address (DMAxIO) 155, 268, 269
- DMAxIO register 155, 268, 269
- DMAxSTART register 157, 268, 270
- dst 229

## **E**

- EI 233
- electrical characteristics 201
  - ADC 215
  - flash memory and timing 214
  - GPIO input data sample timing 218
  - watch-dog timer 214
- enable interrupt 233
- ER 228
- extended addressing register 228
- external pin reset 33
- external RC oscillator 213
- eZ8 CPU features 4
- eZ8 CPU instruction classes 231
- eZ8 CPU instruction notation 228
- eZ8 CPU instruction set 226
- eZ8 CPU instruction summary 235

## **F**

- FCTL register 177, 285
- features, Z8 Encore! 1

first opcode map 247  
 FLAGS 229  
 flags register 229  
 flash  
   controller 5  
   option bit address space 181  
   option bit configuration - reset 181  
   program memory address 0001H 183  
 flash memory  
   arrangement 171  
   byte programming 174  
   code protection 173  
   configurations 170  
   control register definitions 176  
   controller bypass 175  
   electrical characteristics and timing 214  
   flash control register 177, 285  
   flash status register 178  
   frequency high and low byte registers 180  
   mass erase 175  
   operation 172  
   operation timing 172  
   page erase 175  
   page select register 178  
 FPS register 178  
 FSTAT register 178

## **G**

gated mode 79  
 general-purpose I/O 37  
 GPIO 5, 37  
   alternate functions 38  
   architecture 38  
   control register definitions 40  
   input data sample timing 218  
   interrupts 40  
   port A-H address registers 41  
   port A-H alternate function sub-registers 43  
   port A-H control registers 42  
   port A-H data direction sub-registers 42  
   port A-H high drive enable sub-registers 45  
   port A-H input data registers 47  
   port A-H output control sub-registers 44

port A-H output data registers 47  
 port A-H Stop Mode Recovery sub-registers 46  
 port availability by device 37  
 port input timing 218  
 port output timing 219

## **H**

H 229  
 HALT 233  
 halt mode 36, 233  
 hexadecimal number prefix/suffix 229

## **I**

I2C 5  
   10-bit address read transaction 140  
   10-bit address transaction 137  
   10-bit addressed slave data transfer format 137  
   10-bit receive data format 140  
   7-bit address transaction 134  
   7-bit address, reading a transaction 139  
   7-bit addressed slave data transfer format 134, 135, 136  
   7-bit receive data transfer format 139  
   baud high and low byte registers 146, 148, 150  
   C status register 143, 263  
   control register definitions 142  
   controller 129  
   controller signals 15  
   interrupts 131  
   operation 130  
   SDA and SCL signals 131  
   stop and start conditions 133  
 I2CBRH register 147, 148, 150, 263, 264  
 I2CBRL register 147, 263  
 I2CCTL register 145, 263  
 I2CDATA register 143, 262  
 I2CSTAT register 143, 263  
 IM 228  
 immediate data 228  
 immediate operand prefix 229  
 INC 231  
 increment 231