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 "Embedded - Microcontrollers"

Details	
Product Status	Obsolete
Core Processor	eZ8
Core Size	8-Bit
Speed	20MHz
Connectivity	I²C, IrDA, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	46
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 12x10b
Oscillator Type	Internal
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	64-LQFP
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f6402ar020sc

- 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
- 2-9 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!® features seven 8-bit ports (Ports A-G) and one 4-bit port (Port H) for general purpose I/O (GPIO). Each pin is individually programmable.

Flash Controller

The Flash Controller programs and erases the Flash memory.

10-Bit Analog-to-Digital Converter

The Analog-to-Digital Converter (ADC) converts an analog input signal to a 10-bit binary number. The ADC accepts inputs from up to 12 different analog input sources.

UARTs

Each UART is full-duplex and capable of handling asynchronous data transfers. The UARTs support 8- and 9-bit data modes and selectable parity.

I^2C

The inter-integrated circuit (I^2C®) controller makes the Z8 Encore!® compatible with the I^2C protocol. The I^2C controller consists of two bidirectional bus lines, a serial data (SDA) line and a serial clock (SCL) line.

Power-On Reset

The Z8F640x family products contain 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 POR Counter is enabled and counts 514 cycles of the Watch-Dog Timer oscillator. After the POR counter times out, the XTAL Counter is enabled to count a total of 16 system clock pulses. The Z8F640x family device is held in the Reset state until both the POR Counter and XTAL counter have timed out. After the 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 62 illustrates Power-On Reset operation. Refer to the **Electrical Characteristics** chapter for the POR threshold voltage (V_{POR}).

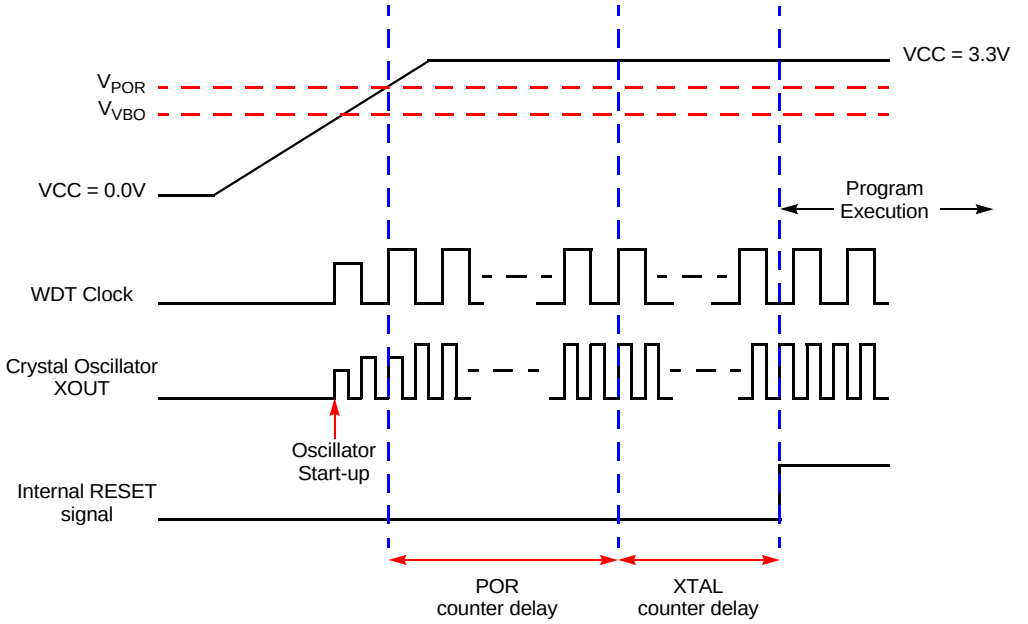

Figure 62. Power-On Reset Operation (not to scale)

Voltage Brown-Out Reset

The devices in the Z8F640x family provide low Voltage Brown-Out (VBO) protection. The VBO circuit senses when the supply voltage drops to an unsafe level (below the VBO

Port A-H Output Data Register

The Port A-H Output Data register (Table 21) writes output data to the pins.

Table 21. Port A-H Output Data Register (P*x*OUT)

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, FDFH, FE3H, FE7H, FEBH, FEFH							

POUT[7:0]—Port Output Data
These bits contain the data to be driven out from 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.

I^2CI— I^2C Interrupt Request
0 = No interrupt request is pending for the I^2C.
1 = An interrupt request from the I^2C 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.

Table 24. Interrupt Request 1 Register (IRQ1)

BITS	7	6	5	4	3	2	1	0
FIELD	PAD7I	PAD6I	PAD5I	PAD4I	PAD3I	PAD2I	PAD1I	PAD0I
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							

PAD*x*I—Port A or Port D Pin *x* Interrupt Request
0 = No interrupt request is pending for GPIO Port A or Port D pin *x*.
1 = An interrupt request from GPIO Port A or Port D pin *x* is awaiting service.

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.

Table 28. IRQ0 Enable Low Bit Register (IRQ0ENL)

BITS	7	6	5	4	3	2	1	0
FIELD	T2ENL	T1ENL	T0ENL	U0RENL	U0TENL	I2CENL	SPIENL	ADCENL
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	FC2H							

T2ENL—Timer 2 Interrupt Request Enable Low Bit
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
I2CENL—I^2C Interrupt Request Enable Low Bit
SPIENL—SPI Interrupt Request Enable Low Bit
ADCENL—ADC Interrupt Request Enable Low Bit

IRQ1 Enable High and Low Bit Registers

The IRQ1 Enable High and Low Bit registers (Tables 30 and 31) form a priority encoded enabling for interrupts in the Interrupt Request 1 register. Priority is generated by setting bits in each register. Table 29 describes the priority control for IRQ1.

Table 29. 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 through 7.

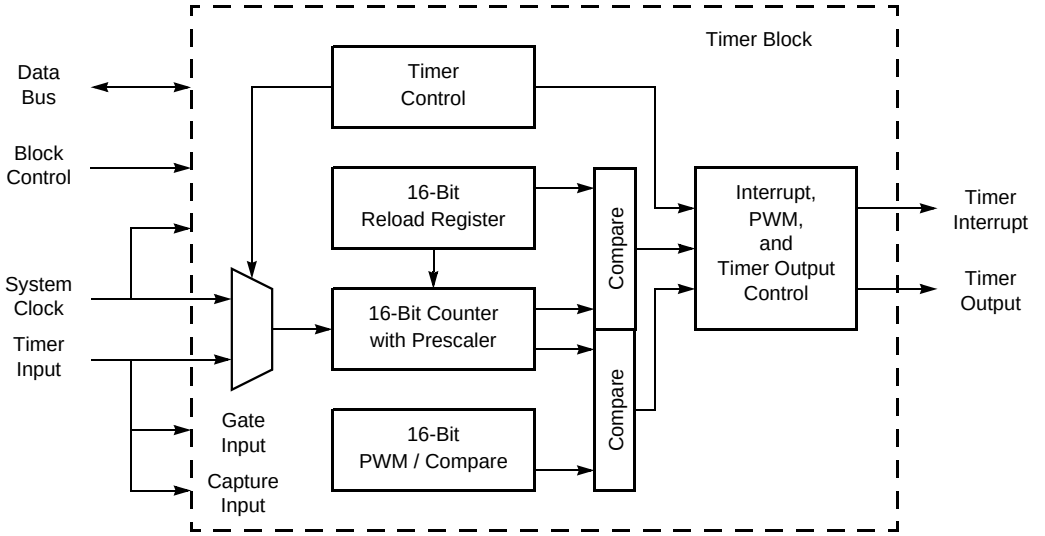

Figure 66. Timer Block Diagram

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. Then, 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 desired to have the Timer Output make a permanent state change upon One-Shot time-

– Select either the rising edge or falling edge of the Timer Input 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 does not have 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. Generally, 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 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. 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:

Counter Mode Timer Input Transitions = Current Count Value – Start Value

PWM Mode

In PWM 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 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 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.

The steps for configuring a timer for PWM 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 desired, 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 given 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 must be used 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 given 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 given 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 desired 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

Table 58. UART Baud Rates (Continued)

10.0 MHz System Clock				5.5296 MHz System Clock			
Desired Rate (kHz)	**BRG Divisor (Decimal)**	**Actual Rate (kHz)**	**Error (%)**	**Desired Rate (kHz)**	**BRG Divisor (Decimal)**	**Actual Rate (kHz)**	**Error (%)**
1250.0	N/A	N/A	N/A	1250.0	N/A	N/A	N/A
625.0	1	625.0	0.00	625.0	N/A	N/A	N/A
250.0	3	208.33	-16.67	250.0	1	345.6	38.24
115.2	5	125.0	8.51	115.2	3	115.2	0.00
57.6	11	56.8	-1.36	57.6	6	57.6	0.00
38.4	16	39.1	1.73	38.4	9	38.4	0.00
19.2	33	18.9	0.16	19.2	18	19.2	0.00
9.60	65	9.62	0.16	9.60	36	9.60	0.00
4.80	130	4.81	0.16	4.80	72	4.80	0.00
2.40	260	2.40	-0.03	2.40	144	2.40	0.00
1.20	521	1.20	-0.03	1.20	288	1.20	0.00
0.60	1042	0.60	-0.03	0.60	576	0.60	0.00
0.30	2083	0.30	0.02	0.30	1152	0.30	0.00

3.579545 MHz System Clock				1.8432 MHz System Clock			
Desired Rate (kHz)	**BRG Divisor (Decimal)**	**Actual Rate (kHz)**	**Error (%)**	**Desired Rate (kHz)**	**BRG Divisor (Decimal)**	**Actual Rate (kHz)**	**Error (%)**
1250.0	N/A	N/A	N/A	1250.0	N/A	N/A	N/A
625.0	N/A	N/A	N/A	625.0	N/A	N/A	N/A
250.0	1	223.72	-10.51	250.0	N/A	N/A	N/A
115.2	2	111.9	-2.90	115.2	1	115.2	0.00
57.6	4	55.9	-2.90	57.6	2	57.6	0.00
38.4	6	37.3	-2.90	38.4	3	38.4	0.00
19.2	12	18.6	-2.90	19.2	6	19.2	0.00
9.60	23	9.73	1.32	9.60	12	9.60	0.00
4.80	47	4.76	-0.83	4.80	24	4.80	0.00
2.40	93	2.41	0.23	2.40	48	2.40	0.00
1.20	186	1.20	0.23	1.20	96	1.20	0.00
0.60	373	0.60	-0.04	0.60	192	0.60	0.00
0.30	746	0.30	-0.04	0.30	384	0.30	0.00

4. The I2C Controller loads the I2C Shift register with the contents of the I2C Data register.
5. After the first bit has been shifted out, a Transmit interrupt is asserted.
6. Software responds by writing eight bits of address to the I2C Data register.
7. The I2C Controller completes shifting of the two address bits and a 0 (write).
8. The I2C slave sends an acknowledge by pulling the SDA signal Low during the next high period of SCL.
9. The I2C Controller loads the I2C Shift register with the contents of the I2C Data register.
10. The I2C Controller shifts out the next eight bits of address. After the first bits are shifted, the I2C Controller generates a Transmit interrupt.
11. Software responds by setting the START bit of the I2C Control register to generate a repeated START.
12. Software responds by writing 11110B followed by the 2-bit slave address and a 1 (read).
13. Software responds by setting the NAK bit of the I2C Control register, so that a Not Acknowledge is sent after the first byte of data has been read. If you want to read only one byte, software responds by setting the NAK bit of the I2C Control register.
14. After the I2C Controller shifts out the address bits mentioned in step 9, the I2C slave sends an acknowledge by pulling the SDA signal Low during the next high period of SCL.
15. The I2C Controller sends the repeated START condition.
16. The I2C Controller loads the I2C Shift register with the contents of the I2C Data register.
17. The I2C Controller sends 11110B followed by the 2-bit slave read and a 1 (read).
18. The I2C slave sends an acknowledge by pulling the SDA signal Low during the next high period of SCL.
19. The I2C slave sends a byte of data.
20. A Receive interrupt is generated.
21. Software responds by reading the I2C Data register.
22. Software responds by setting the STOP bit of the I2C Control register.
23. A NAK condition is sent to the I2C slave.
24. A STOP condition is sent to the I2C slave.

Direct Memory Access Controller

Overview

The Z8F640x family device's Direct Memory Access (DMA) Controller provides three independent Direct Memory Access channels. Two of the channels (DMA0 and DMA1) transfer data between the on-chip peripherals and the Register File. The third channel (DMA_ADC) controls the Analog-to-Digital Converter (ADC) operation and transfers the Single-Shot mode ADC output data to the Register File.

Operation

DMA0 and DMA1 Operation

DMA0 and DMA1, referred to collectively as DMA*x*, transfer data either from the on-chip peripheral control registers to the Register File, or from the Register File to the on-chip peripheral control registers. The sequence of operations in a DMA*x* data transfer is:

1. DMA*x* trigger source requests a DMA data transfer.
2. DMA*x* requests control of the system bus (address and data) from the eZ8 CPU.
3. After the eZ8 CPU acknowledges the bus request, DMA*x* transfers either a single byte or a two-byte word (depending upon configuration) and then returns system bus control back to the eZ8 CPU.
4. If Current Address equals End Address:
   - DMA*x* reloads the original Start Address
   - If configured to generate an interrupt, DMA*x* sends an interrupt request to the Interrupt Controller
   - If configured for single-pass operation, DMA*x* resets the DEN bit in the DMA*x* Control register to 0 and the DMA is disabled.

   If Current Address does not equal End Address, the Current Address increments by 1 (single-byte transfer) or 2 (two-byte word transfer).

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).

Flash Page Select Register

The Flash Page Select register is used to select one of the 128 available Flash memory pages to be erased in a Page Erase operation. 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 erased (all bytes written to FFH).

Table 87. 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
This bit is reserved and must be 0.

PAGE—Page Select
This 7-bit field identifies the Flash memory page for Page Erase operation.
Program Memory Address[15:9] = PAGE[6:0]

On-Chip Debugger

Overview

The Z8F640x family devices have 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
- Execution of eZ8 CPU instructions.

Architecture

The On-Chip Debugger consists of four primary functional blocks: transmitter, receiver, auto-baud generator, and debug controller. Figure 86 illustrates the architecture of the On-Chip Debugger

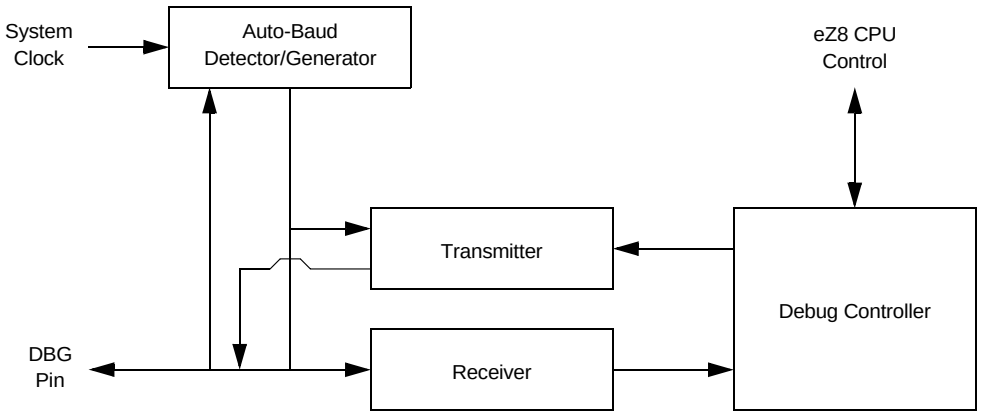

Figure 86. On-Chip Debugger Block Diagram

ister. When the Watchpoint event occurs, the Z8F640x family device enters Debug mode and the DBGMODE bit in the OCDCTL register becomes 1.

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 of the Z8F640x family device, 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 Read Protect Option Bit (RP). The Read Protect Option Bit prevents the code in memory from being read out of the Z8F640x family device. When this option is enabled, several of the OCD commands are disabled. Table 93 contains a summary of the On-Chip Debugger commands. Each OCD command is described in further detail in the bulleted list following Table 93. Table 93 indicates those commands that operate when the Z8F640x family device is not in Debug mode (normal operation) and those commands that are disabled by programming the Read Protect Option Bit.

Table 93. On-Chip Debugger Commands

Debug Command	Command Byte	Enabled when NOT in Debug mode?	Disabled by 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 Watchpoint (20H)**—The Write Watchpoint command sets and configures the debug Watchpoint. If the Z8F640x family device is not in Debug mode or the Read Protect Option Bit is enabled, the WPTCTL bits are all set to zero.

```
DBG <-- 20H
DBG <-- WPTCTL[7:0]
DBG <-- WPTADDR[7:0]
DBG <-- WPTDATA[7:0]
```

- **Read Watchpoint (21H)**—The Read Watchpoint command reads the current Watchpoint registers.

```
DBG <-- 21H
DBG --> WPTCTL[7:0]
DBG --> WPTADDR[7:0]
DBG --> WPTDATA[7:0]
```

On-Chip Debugger Control Register Definitions

OCD Control Register

The OCD Control register controls the state of the On-Chip Debugger. This register enters or exits Debug mode and enables the BRK instruction. It can also reset the Z8F640x family 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 Z8F640x family device is in Debug mode, a “run” function can be implemented by writing 40H to this register.

Table 94. 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
Setting this bit to 1 causes the Z8F640x family device to enter Debug mode. When in Debug mode, the eZ8 CPU stops fetching new instructions. Clearing this bit causes the eZ8 CPU to start running again. This bit is automatically set when a BRK instruction is decoded and Breakpoints are enabled or when a Watchpoint Debug Break is detected. If the Read Protect Option Bit is enabled, this bit can only be cleared by resetting the Z8F640x family device, it cannot be written to 0.
0 = The Z8F640x family device is operating in normal mode.
1 = The Z8F640x family device is in Debug mode.

General Purpose I/O Port Output Timing

Figure 94 and Table 108 provide timing information for GPIO Port pins.

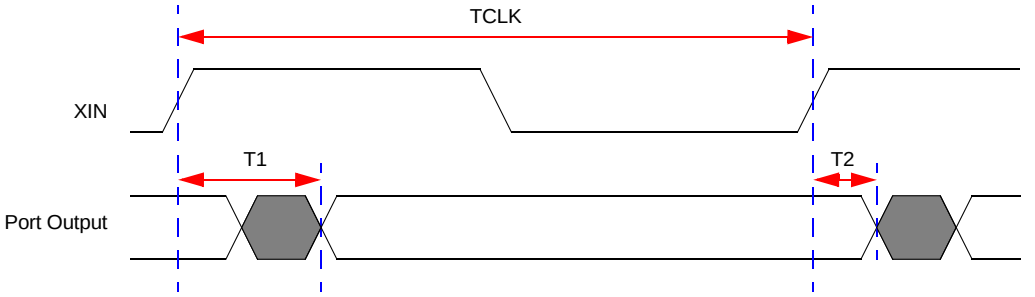

Figure 94. GPIO Port Output Timing

Table 108. GPIO Port Output Timing

		Delay (ns)	
Parameter	**Abbreviation**	**Minimum**	**Maximum**
T_1	XIN Rise to Port Output Valid Delay	–	15
T_2	XIN Rise to Port Output Hold Time	2	–

SPI Master Mode Timing

Figure 96 and Table 110 provide timing information for SPI Master mode pins. Timing is shown with SCK rising edge used to source MOSI output data, SCK falling edge used to sample MISO input data. Timing on the SS output pin(s) is controlled by software.

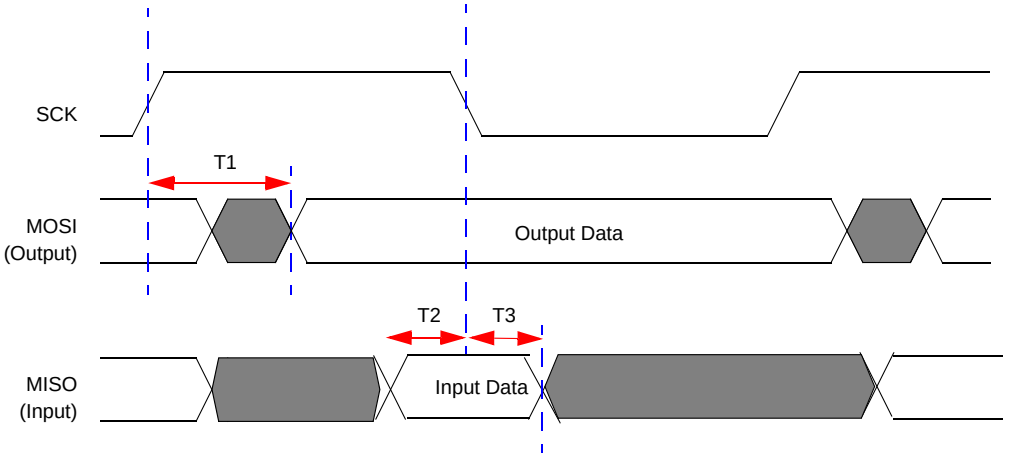

Figure 96. SPI Master Mode Timing

Table 110. SPI Master Mode Timing

		Delay (ns)	
Parameter	**Abbreviation**	**Minimum**	**Maximum**
T_1	SCK Rise to MOSI output Valid Delay	-5	+5
T_2	MISO input to SCK (receive edge) Setup Time	20	
T_3	MISO input to SCK (receive edge) Hold Time	0	

Ordering Information

Table 128. Ordering Information

Part	Flash KB (Bytes)	RAM KB (Bytes)	Max. Speed (MHz)	Temp (^{0}C)	Voltage (V)	Package	Part Number
Z8 Encore!® with 16KB Flash, Standard Temperature							
Z8 Encore!®	16 (16,384)	2 (2048)	20	0 to +70	3.0 - 3.6	PDIP-40	Z8F1601PM020SC
Z8 Encore!®	16 (16,384)	2 (2048)	20	0 to +70	3.0 - 3.6	LQFP-44	Z8F1601AN020SC
Z8 Encore!®	16 (16,384)	2 (2048)	20	0 to +70	3.0 - 3.6	PLCC-44	Z8F1601VN020SC
Z8 Encore!®	16 (16,384)	2 (2048)	20	0 to +70	3.0 - 3.6	LQFP-64	Z8F1602AR020SC
Z8 Encore!®	16 (16,384)	2 (2048)	20	0 to +70	3.0 - 3.6	PLCC-68	Z8F1602VS020SC
Z8 Encore!® with 24KB Flash, Standard Temperature							
Z8 Encore!®	24 (24,576)	2 (2048)	20	0 to +70	3.0 - 3.6	PDIP-40	Z8F2401PM020SC
Z8 Encore!®	24 (24,576)	2 (2048)	20	0 to +70	3.0 - 3.6	LQFP-44	Z8F2401AN020SC
Z8 Encore!®	24 (24,576)	2 (2048)	20	0 to +70	3.0 - 3.6	PLCC-44	Z8F2401VN020SC
Z8 Encore!®	24 (24,576)	2 (2048)	20	0 to +70	3.0 - 3.6	LQFP-64	Z8F2402AR020SC
Z8 Encore!®	24 (24,576)	2 (2048)	20	0 to +70	3.0 - 3.6	PLCC-68	Z8F2402VS020SC
Z8 Encore!® with 32KB Flash, Standard Temperature							
Z8 Encore!®	32 (32,768)	2 (2048)	20	0 to +70	3.0 - 3.6	PDIP-40	Z8F3201PM020SC
Z8 Encore!®	32 (32,768)	2 (2048)	20	0 to +70	3.0 - 3.6	LQFP-44	Z8F3201AN020SC
Z8 Encore!®	32 (32,768)	2 (2048)	20	0 to +70	3.0 - 3.6	PLCC-44	Z8F3201VN020SC
Z8 Encore!®	32 (32,768)	2 (2048)	20	0 to +70	3.0 - 3.6	LQFP-64	Z8F3202AR020SC
Z8 Encore!®	32 (32,768)	2 (2048)	20	0 to +70	3.0 - 3.6	PLCC-68	Z8F3202VS020SC
Z8 Encore!®with 48KB Flash, Standard Temperature							
Z8 Encore!®	48 (49,152)	4 (4096)	20	0 to +70	3.0 - 3.6	PDIP-40	Z8F4801PM020SC
Z8 Encore!®	48 (49,152)	4 (4096)	20	0 to +70	3.0 - 3.6	LQFP-44	Z8F4801AN020SC
Z8 Encore!®	48 (49,152)	4 (4096)	20	0 to +70	3.0 - 3.6	PLCC-44	Z8F4801VN020SC
Z8 Encore!®	48 (49,152)	4 (4096)	20	0 to +70	3.0 - 3.6	LQFP-64	Z8F4802AR020SC
Z8 Encore!®	48 (49,152)	4 (4096)	20	0 to +70	3.0 - 3.6	PLCC-68	Z8F4802VS020SC
Z8 Encore!®	48 (49,152)	4 (4096)	20	0 to +70	3.0 - 3.6	QFP-80	Z8F4803FT020SC

F

G

H

I