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	68-LCC (J-Lead)
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f6422vs020sg

DMA Controller

The Z8 Encore! XP F64xx Series feature three channels of DMA. Two of the channels are for register RAM to and from I/O operations. The third channel automatically controls the transfer of data from the ADC to the memory.

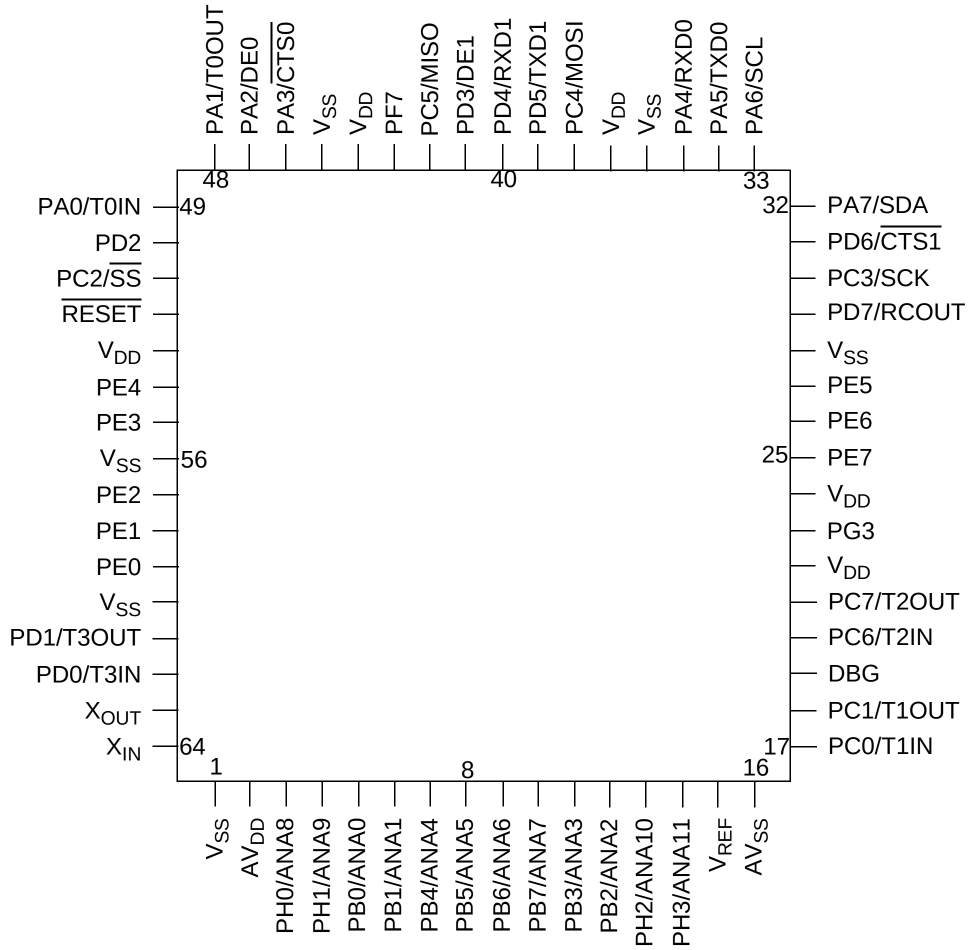

Figure 5. Z8 Encore! XP F64xx Series in 64-Pin Low-Profile Quad Flat Package (LQFP)

Table 5. Z8 Encore! XP F64xx Series Program Memory Maps (Continued)

Program Memory Address (Hex)	Function
0006-0007	Illegal Instruction Trap
0008-0037	Interrupt Vectors*
0038-BFFF	Program Memory
Z8F642x Products	
0000-0001	Option Bits
0002-0003	Reset Vector
0004-0005	WDT Interrupt Vector
0006-0007	Illegal Instruction Trap
0008-0037	Interrupt Vectors*
0038-FFFF	Program Memory

Note: *See Table 23 on page 48 for a list of the interrupt vectors.

Data Memory

The Z8 Encore! XP F64xx Series does not use the eZ8 CPU’s 64KB data memory address space.

Information Area

Table 6 describes the Z8 Encore! XP F64xx Series’ Information Area. This 512-byte Information Area is accessed by setting bit 7 of the Page Select Register to 1. When access is enabled, the Information Area is mapped into program memory and overlays the 512 bytes at addresses FE00H to FFFFH. When the Information Area access is enabled, execution of the LDC and LDCI instructions from these program memory addresses return the Information Area data rather than the program memory data. Reads of these addresses through the On-Chip Debugger also returns the Information Area data. Execution of code from these addresses continues to correctly use program memory. Access to the Information Area is read-only.

Register File Address Map

Table 7 provides the address map for the Register File of the Z8 Encore! XP F64xx Series products. Not all devices and package styles in the Z8 Encore! XP F64xx Series support Timer 3 and all of the GPIO ports. Consider registers for unimplemented peripherals to be reserved.

Table 7. Z8 Encore! XP F64xx Series Register File Address Map

Address (Hex)	Register Description	Mnemonic	Reset (Hex)	Page
General-Purpose RAM				
000–EFF	General-Purpose Register File RAM	—	XX	
Timer 0				
F00	Timer 0 High Byte	T0H	00	72
F01	Timer 0 Low Byte	T0L	01	72
F02	Timer 0 Reload High Byte	T0RH	FF	74
F03	Timer 0 Reload Low Byte	T0RL	FF	74
F04	Timer 0 PWM High Byte	T0PWMH	00	75
F05	Timer 0 PWM Low Byte	T0PWML	00	75
F06	Timer 0 Control 0	T0CTL0	00	76
F07	Timer 0 Control 1	T0CTL1	00	77
Timer 1				
F08	Timer 1 High Byte	T1H	00	72
F09	Timer 1 Low Byte	T1L	01	72
F0A	Timer 1 Reload High Byte	T1RH	FF	74
F0B	Timer 1 Reload Low Byte	T1RL	FF	74
F0C	Timer 1 PWM High Byte	T1PWMH	00	75
F0D	Timer 1 PWM Low Byte	T1PWML	00	75
F0E	Timer 1 Control 0	T1CTL0	00	76
F0F	Timer 1 Control 1	T1CTL1	00	77
Timer 2				
F10	Timer 2 High Byte	T2H	00	72
F11	Timer 2 Low Byte	T2L	01	72
F12	Timer 2 Reload High Byte	T2RH	FF	74
F13	Timer 2 Reload Low Byte	T2RL	FF	74

Note: XX = Undefined.

Table 24. Interrupt Request 0 Register (IRQ0)

Bit	7	6	5	4	3	2	1	0
Field	T2I	T1I	T0I	U0RXI	U0TXI	I2CI	SPII	ADCI
RESET	0							
R/W	R/W							
Address	FC0H							

Bit	Description
[7] T2I	**Timer 2 Interrupt Request** 0 = No interrupt request is pending for Timer 2. 1 = An interrupt request from Timer 2 is awaiting service.
[6] T1I	**Timer 1 Interrupt Request** 0 = No interrupt request is pending for Timer 1. 1 = An interrupt request from Timer 1 is awaiting service.
[5] T0I	**Timer 0 Interrupt Request** 0 = No interrupt request is pending for Timer 0. 1 = An interrupt request from Timer 0 is awaiting service.
[4] U0RXI	**UART 0 Receiver Interrupt Request** 0 = No interrupt request is pending for the UART 0 receiver. 1 = An interrupt request from the UART 0 receiver is awaiting service.
[3] U0TXI	**UART 0 Transmitter Interrupt Request** 0 = No interrupt request is pending for the UART 0 transmitter. 1 = An interrupt request from the UART 0 transmitter is awaiting service.
[2] I2CI	**I^{2}C Interrupt Request** 0 = No interrupt request is pending for the I^{2}C. 1 = An interrupt request from the I^{2}C is awaiting service.
[1] SPII	**SPI Interrupt Request** 0 = No interrupt request is pending for the SPI. 1 = An interrupt request from the SPI is awaiting service.
[0] 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 2 Register

The Interrupt Request 2 (IRQ2) Register, shown in Table 26, 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 26. 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							

Bit	**Description**
[7] T3I	**Timer 3 Interrupt Request** 0 = No interrupt request is pending for Timer 3. 1 = An interrupt request from Timer 3 is awaiting service.
[6] 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.
[5] 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.
[4] DMAI	**DMA Interrupt Request** 0 = No interrupt request is pending for the DMA. 1 = An interrupt request from the DMA is awaiting service.
[3:0] 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.
Note: x indicates the specific GPIO Port C pin in the range [3:0].	

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.

Watchdog Timer

The Watchdog Timer (WDT) helps protect against corrupt or unreliable software, power faults and other system-level problems which can place the Z8 Encore! XP F64xx Series MCU into unsuitable operating states. The features of the Watchdog Timer include:

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

Operation

The Watchdog Timer is a retriggerable one-shot timer that resets or interrupts the Z8 Encore! XP F64xx Series devices when the WDT reaches its terminal count. The Watchdog Timer uses its own dedicated on-chip RC oscillator as its clock source. The Watchdog Timer has only two modes of operation: ON and OFF. After it is 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. This WDT_AO bit enables the Watchdog Timer to operate continuously, even if a WDT instruction has not been executed.

The Watchdog 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 calculated using the following equation:

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

In the above equation, the WDT reload value is the decimal value of the 24-bit value provided by {WDTU[7:0], WDTH[7:0], WDTL[7:0]}; the typical Watchdog Timer RC oscillator frequency is 10kHz. The Watchdog Timer cannot be refreshed after it reaches `000002H`. The WDT reload value must not be set to values below `000004H`.

Table 47 lists approximate time-out delays for the minimum and maximum WDT reload values.

STOP Mode. For more information about Stop Mode Recovery, see the Reset and Stop Mode Recovery chapter on page 28.

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

WDT Reset in Normal Operation

If configured to generate a Reset when a time-out occurs, the Watchdog Timer forces the device into the Reset state. The WDT status bit in the Watchdog Timer Control Register is set to 1. For more information about Reset, see the Reset and Stop Mode Recovery chapter on page 28.

WDT Reset in STOP Mode

If enabled in STOP Mode and configured to generate a Reset when a time-out occurs and the device is in STOP Mode, the Watchdog Timer initiates a Stop Mode Recovery. Both the WDT status bit and the stop bit in the Watchdog Timer Control Register are set to 1 following WDT time-out in STOP Mode. Default operation is for the WDT and its RC oscillator to be enabled during STOP Mode.

WDT RC Disable in STOP Mode

To minimize power consumption in STOP Mode, the WDT and its RC oscillator can be disabled in STOP Mode. The following sequence configures the WDT to be disabled when the Z8 Encore! XP F64xx Series devices enter STOP Mode following execution of a stop instruction:

1. Write `55H` to the Watchdog Timer Control Register (WDTCTL).
2. Write `AAH` to the Watchdog Timer Control Register (WDTCTL).
3. Write `81H` to the Watchdog Timer Control Register (WDTCTL) to configure the WDT and its oscillator to be disabled during STOP Mode. Alternatively, write `00H` to the Watchdog Timer Control Register (WDTCTL) as the third step in this sequence to reconfigure the WDT and its oscillator to be enabled during STOP Mode.

This sequence only affects WDT operation in STOP Mode.

Watchdog Timer Reload Unlock Sequence

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

Receiving IrDA Data

Data received from the infrared transceiver via the **IR_RXD** signal through the RxD pin is decoded by the infrared endec and passed to the UART. The UART's baud rate clock is used by the infrared endec to generate the demodulated signal (RxD) that drives the UART. Each UART/Infrared data bit is 16-clocks wide. Figure 21 displays data reception. When the infrared endec is enabled, the UART's RxD signal is internal to the Z8 Encore! XP F64xx Series products while the IR_RXD signal is received through the RxD pin.

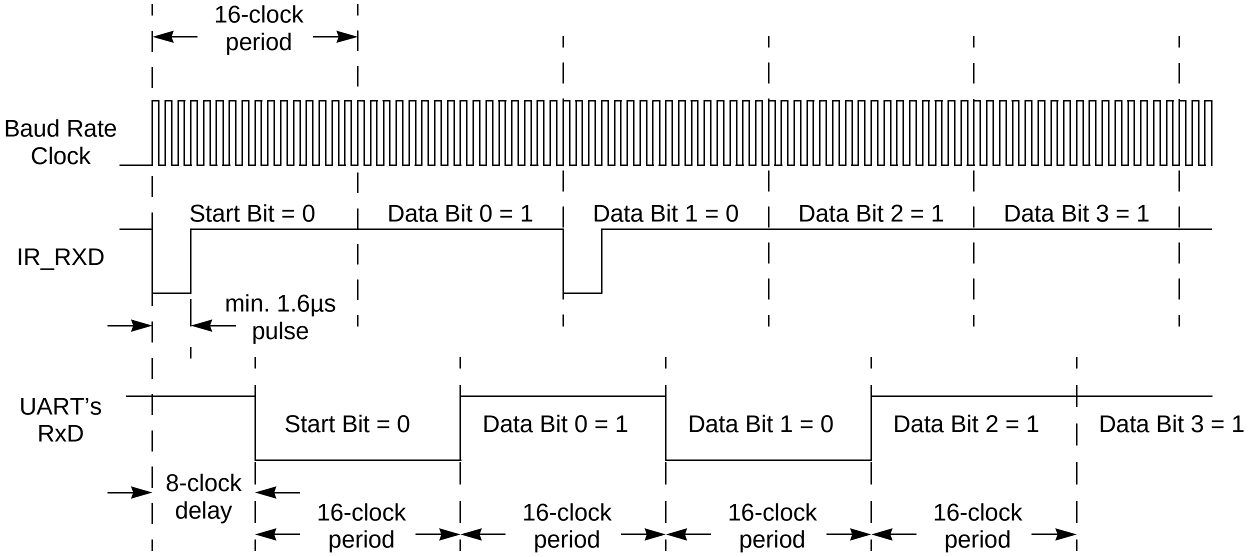

Figure 21. Infrared Data Reception

! **Caution:** The system clock frequency must be at least 1.0 MHz to ensure proper reception of the 1.6 µs minimum width pulses allowed by the IrDA standard.

Endec Receiver Synchronization

The IrDA receiver uses a local baud rate clock counter (0 to 15 clock periods) to generate an input stream for the UART and to create a sampling window for detection of incoming pulses. The generated UART input (UART RxD) is delayed by 8 baud rate clock periods with respect to the incoming IrDA data stream. When a falling edge in the input data stream is detected, the endec counter is reset. When the count reaches a value of 8, the UART RxD value is updated to reflect the value of the decoded data. When the count reaches 12 baud clock periods, the sampling window for the next incoming pulse opens. The window remains open until the count again reaches 8 (i.e., 24 baud clock periods

Bit	Description (Continued)
[5] COL	**Collision** 0 = A multimaster collision (mode fault) has not occurred. 1 = A multimaster collision (mode fault) has been detected.
[4] ABT	**Slave Mode Transaction Abort** This bit is set if the SPI is configured in slave mode, a transaction is occurring and $\overline{\text{SS}}$ deasserts before all bits of a character have been transferred as defined by the NUMBITS field of the SPIMODE Register. The IRQ bit also sets, indicating the transaction has completed. 0 = A slave mode transaction abort has not occurred. 1 = A slave mode transaction abort has been detected.
[3:2]	**Reserved** These bits are reserved and must be programmed to 00.
[1] TXST	**Transmit Status** 0 = No data transmission currently in progress. 1 = Data transmission currently in progress.
[0] SLAS	**Slave Select** If SPI enabled as a Slave, then the following conditions are true: 0 = $\overline{\text{SS}}$ input pin is asserted (Low). 1 = $\overline{\text{SS}}$ input is not asserted (High). If SPI enabled as a Master, this bit is not applicable.

clears the stop and NCKI bits. The transaction is complete (ignore the following steps).

17. The I^{2}C Controller shifts the data out by the SDA signal. After the first bit is sent, the transmit interrupt is asserted.
18. If more bytes remain to be sent, return to Step 14.
19. If the last byte is currently being sent, software sets the stop bit of the I^{2}C Control Register (or start bit to initiate a new transaction). In the stop case, software also clears the TXI bit of the I^{2}C Control Register at the same time.
20. The I^{2}C Controller completes transmission of the last data byte on the SDA signal.
21. The slave may either Acknowledge or Not Acknowledge the last byte. Because either the stop or start bit is already set, the NCKI interrupt does not occur.
22. The I^{2}C Controller sends the stop (or RESTART) condition to the I^{2}C bus and clears the stop (or start) bit.

Read Transaction with a 7-Bit Address

Figure 32 displays the data transfer format for a read operation to a 7-bit addressed slave. The shaded regions indicate data transferred from the I^{2}C Controller to slaves and unshaded regions indicate data transferred from the slaves to the I^{2}C Controller.

S	Slave Address	R = 1	A	Data	A	Data	A	P/S

Figure 32. Receive Data Transfer Format for a 7-Bit Addressed Slave

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

1. Software writes the I^{2}C Data Register with a 7-bit slave address plus the read bit (=1).
2. Software asserts the start bit of the I^{2}C Control Register.
3. If this is a single byte transfer, Software asserts the NAK bit of the I^{2}C Control Register so that after the first byte of data has been read by the I^{2}C Controller, a Not Acknowledge is sent to the I^{2}C slave.
4. The I^{2}C Controller sends the start condition.
5. The I^{2}C Controller shifts the address and read bit out the SDA signal.
6. If the I^{2}C slave acknowledges the address by pulling the SDA signal Low during the next High period of SCL, the I^{2}C Controller sets the ACK bit in the I^{2}C Status Register. Continue with Step 7.

   If the slave does not acknowledge, the Not Acknowledge interrupt occurs (NCKI bit is

Table 71. I^{2}C Data Register (I2CDATA)

Bit	7	6	5	4	3	2	1	0
Field	DATA							
RESET	0							
R/W	R/W							
Address	F50H							

I^{2}C Status Register

The read-only I^{2}C Status Register, shown in Table 72, indicates the status of the I^{2}C Controller.

Table 72. I^{2}C Status Register (I2CSTAT)

Bit	7	6	5	4	3	2	1	0
Field	TDRE	RDRF	ACK	10B	RD	TAS	DSS	NCKI
RESET	1	0						
R/W	R							
Address	F51H							

Bit	Description
[7] TDRE	**Transmit Data Register Empty** When the I^{2}C Controller is enabled, this bit is 1 when the I^{2}C Data Register is empty. When this bit is set, an interrupt is generated if the TXI bit is set, except when the I^{2}C Controller is shifting in data during the reception of a byte or when shifting an address and the RD bit is set. This bit is cleared by writing to the I2CDATA Register.
[6] RDRF	**Receive Data Register Full** This bit is set = 1 when the I^{2}C Controller is enabled and the I^{2}C Controller has received a byte of data. When asserted, this bit causes the I^{2}C Controller to generate an interrupt. This bit is cleared by reading the I^{2}C Data Register (unless the read is performed using execution of the On-Chip Debugger’s Read Register command).

Direct Memory Access Controller

The Z8 Encore! XP F64xx Series 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 ADC operation and transfers SINGLE-SHOT Mode ADC output data to the Register File.

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 to the eZ8 CPU.
4. If the Current Address equals the End Address, then the following conditions are true:
   - 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 the Current Address does not equal the End Address, then the Current Address increments by 1 (single-byte transfer) or 2 (two-byte word transfer).

Configuring DMA0 and DMA1 for Data Transfer

Observe the following procedure to configure and enable DMA0 or DMA1:

1. Write to the DMA*x* I/O Address Register to set the Register File address identifying the on-chip peripheral control register. The upper nibble of the 12-bit address for on-chip peripheral control registers is always `FH`. The full address is {FH, DMA*x*_IO[7:0]}.

DMA*x* Start/Current Address Low Byte Register

The DMA*x* Start/Current Address Low Byte Register, shown in Table 81, in conjunction with the DMA*x* Address High Nibble Register, shown in Table 80, forms a 12-bit Start/Current Address. Writes to this register set the Start Address for DMA operations. Each time the DMA completes a data transfer, the 12-bit Start/Current Address increments by either 1 (single-byte transfer) or 2 (two-byte word transfer). Reads from this register return the low byte of the current address to be used for the next DMA data transfer.

Table 81. DMA*x* Start/Current Address Low Byte Register (DMA*x*START)

Bit	7	6	5	4	3	2	1	0
Field	DMA_START							
RESET	X							
R/W	R/W							
Address	FB3H, FBBH							

Bit	Description
[7:0] DMA_START	**DMA*x* Start/Current Address Low** These bits, with the four lower bits of the DMA*x*_H Register, form the 12-bit Start/Current address. The full 12-bit address is provided by {DMA_START_H[3:0], DMA_START[7:0]}.

DMA*x* End Address Low Byte Register

The DMA*x* End Address Low Byte Register, shown in Table 82, forms a 12-bit End Address.

Table 82. DMA*x* End Address Low Byte Register (DMA*x*END)

Bit	7	6	5	4	3	2	1	0
Field	DMA_END							
RESET	X							
R/W	R/W							
Address	FB4H, FBCH							

Bit	Description
[7] DMA_END	**DMA*x* End Address Low** These bits, with the four upper bits of the DMA*x*_H Register, form a 12-bit address. This address is the ending location of the DMA*x* transfer. The full 12-bit address is provided by {DMA_END_H[3:0], DMA_END[7:0]}.

Table 95. Page Select Register (FPS)

Bit	7	6	5	4	3	2	1	0
Field	INFO_EN	PAGE						
RESET	0							
R/W	R/W							
Address	FF9H							

Bit	Description
[7] INFO_EN	**Information Area Enable** 0 = Information Area is not selected. 1 = Information Area is selected. The Information area is mapped into the Flash memory address space at addresses FE00H through FFFFH.
[6:0] PAGE	**Page Select** This 7-bit field selects the Flash memory page for programming and Page Erase operations. Flash Memory Address[15:9] = PAGE[6:0].

Flash Sector Protect Register

The Flash Sector Protect Register, shown in Table 96, protects Flash memory sectors from being programmed or erased from user code. The Flash Sector Protect Register shares its Register File address with the Page Select Register. The Flash Sector Protect Register can be accessed only after writing the Flash Control Register with 5EH.

User code can only write bits in this register to 1 (bits cannot be cleared to 0 by user code). To determine the appropriate Flash memory sector address range and sector number for your Z8F64xx Series product, please refer to Table 91 on page 169.

Table 96. Flash Sector Protect Register (FPROT)

Bit	7	6	5	4	3	2	1	0
Field	SECT7	SECT6	SECT5	SECT4	SECT3	SECT2	SECT1	SECT0
RESET	0							
R/W	R/W*							
Address	FF9H							
Note: *R/W = This register is accessible for read operations; it can be written to 1 only via user code.								

Bit	Description
[7:0] SECT*n*	**Sector Protect**** 0 = Sector *n* can be programmed or erased from user code. 1 = Sector *n* is protected and cannot be programmed or erased from user code.
Note: **User code can only write bits from 0 to 1.	

Table 102. 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	—
Read OCD Status Register	02H	Yes	—
Read Runtime Counter	03H	—	—
Write OCD Control Register	04H	Yes	Cannot clear DBGMODE bit
Read OCD Control Register	05H	Yes	—
Write Program Counter	06H	—	Disabled
Read Program Counter	07H	—	Disabled
Write Register	08H	—	Only writes of the Flash memory control registers are allowed. Additionally, only the Mass Erase command is allowed to be written to the Flash Control Register.
Read Register	09H	—	Disabled
Write Program Memory	0AH	—	Disabled
Read Program Memory	0BH	—	Disabled
Write Data Memory	0CH	—	Disabled
Read Data Memory	0DH	—	Disabled
Read Program Memory CRC	0EH	—	—
Reserved	0FH	—	—
Step Instruction	10H	—	Disabled
Stuff Instruction	11H	—	Disabled
Execute Instruction	12H	—	Disabled
Reserved	13H–FFH	—	—

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

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

On-Chip Debugger Timing

Figure 52 and Table 117 provide timing information for the DBG pin. The DBG pin timing specifications assume a 4 µs maximum rise and fall time.

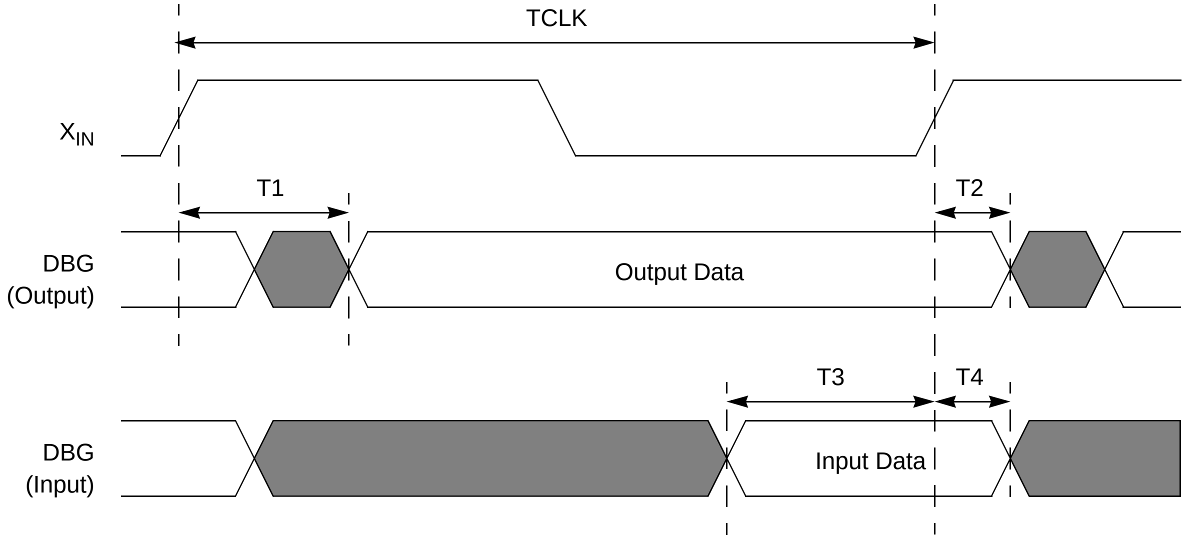

Figure 52. On-Chip Debugger Timing

Table 117. On-Chip Debugger Timing

		Delay (ns)	
Parameter	**Abbreviation**	**Minimum**	**Maximum**
DBG			
T_1	X_{IN} Rise to DBG Valid Delay	–	30
T_2	X_{IN} Rise to DBG Output Hold Time	2	–
T_3	DBG to X_{IN} Rise Input Setup Time	10	–
T_4	DBG to X_{IN} Rise Input Hold Time	5	–
	DBG frequency		System Clock/4

Hex Address: FEB

Table 256. Port A–H Output Data Register (P*x*OUT)

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	POUT7	POUT6	POUT5	POUT4	POUT3	POUT2	POUT1	POUT0
RESET	0							
R/W	R/W							
Address	FD3H, FD7H, FDBH, FDFH, FE3H, FE7H, FEBH, FEFH							

Hex Address: FEC

Table 257. Port A–H GPIO Address Registers (P*x*ADDR)

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	PADDR[7:0]							
RESET	00H							
R/W	R/W							
Address	FD0H, FD4H, FD8H, FDCH, FE0H, FE4H, FE8H, FECH							

Hex Address: FED

Table 258. Port A–H Control Registers (P*x*CTL)

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	PCTL							
RESET	00H							
R/W	R/W							
Address	FD1H, FD5H, FD9H, FDDH, FE1H, FE5H, FE9H, FEDH							

Hex Address: FEE

Table 259. Port A–H Input Data Registers (PxIN)

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	PIN7	PIN6	PIN5	PIN4	PIN3	PIN2	PIN1	PIN0
RESET	X							
R/W	R							
Address	FD2H, FD6H, FDAH, FDEH, FE2H, FE6H, FEAH, FEEH							