
**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 | 60 |
| Program Memory Size | 48KB (48K 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 | -40°C ~ 105°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 80-BQFP |
| Supplier Device Package | - |
| Purchase URL | https://www.e-xfl.com/product-detail/zilog/z8f4823ft020ec |

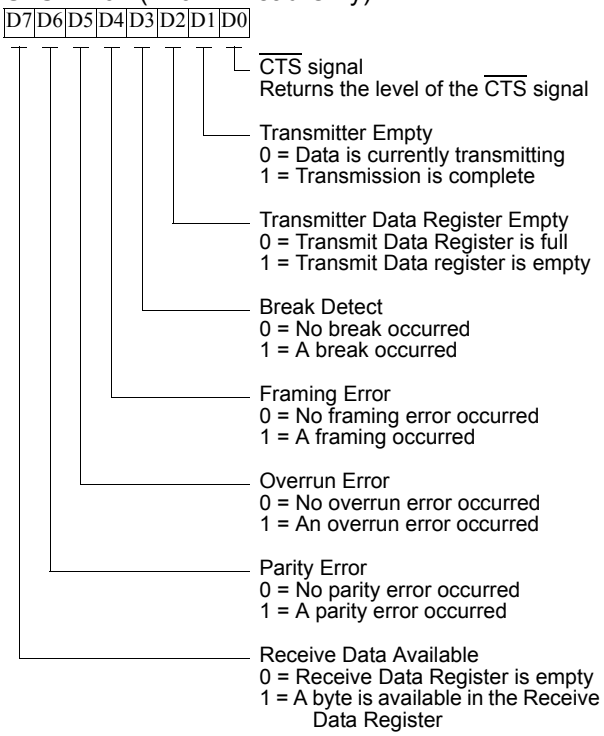
# Register File Address Map

Table 7 provides the address map for the Register File of the 64K Series products. Not all devices and package styles in the 64K Series support Timer 3 and all of the GPIO Ports. Consider registers for unimplemented peripherals as Reserved.

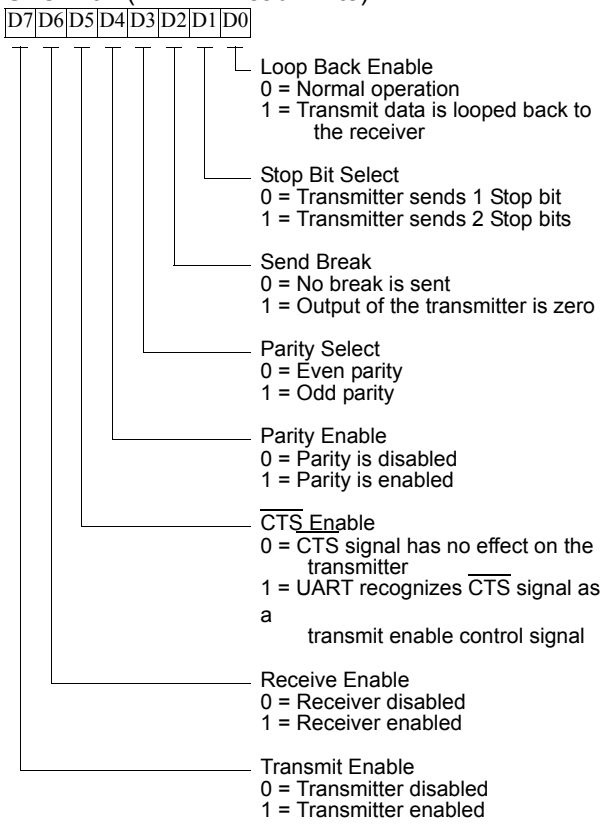
**Table 7. Z8 Encore! XP 64K Series Flash Microcontrollers Register File Address Map**

| Address (Hex) | Register Description | Mnemonic | Reset (Hex) | Page No |
|---|---|---|---|---|
| **General-Purpose RAM** | | | | |
| 000-EFF | General-Purpose Register File RAM | — | XX | |
| **Timer 0** | | | | |
| F00 | Timer 0 High Byte | T0H | 00 | 90 |
| F01 | Timer 0 Low Byte | T0L | 01 | 90 |
| F02 | Timer 0 Reload High Byte | T0RH | FF | 91 |
| F03 | Timer 0 Reload Low Byte | T0RL | FF | 91 |
| F04 | Timer 0 PWM High Byte | T0PWMH | 00 | 92 |
| F05 | Timer 0 PWM Low Byte | T0PWML | 00 | 92 |
| F06 | Timer 0 Control 0 | T0CTL0 | 00 | 93 |
| F07 | Timer 0 Control 1 | T0CTL1 | 00 | 94 |
| **Timer 1** | | | | |
| F08 | Timer 1 High Byte | T1H | 00 | 90 |
| F09 | Timer 1 Low Byte | T1L | 01 | 90 |
| F0A | Timer 1 Reload High Byte | T1RH | FF | 91 |
| F0B | Timer 1 Reload Low Byte | T1RL | FF | 91 |
| F0C | Timer 1 PWM High Byte | T1PWMH | 00 | 92 |
| F0D | Timer 1 PWM Low Byte | T1PWML | 00 | 92 |
| F0E | Timer 1 Control 0 | T1CTL0 | 00 | 93 |
| F0F | Timer 1 Control 1 | T1CTL1 | 00 | 94 |
| **Timer 2** | | | | |
| F10 | Timer 2 High Byte | T2H | 00 | 90 |
| F11 | Timer 2 Low Byte | T2L | 01 | 90 |
| F12 | Timer 2 Reload High Byte | T2RH | FF | 91 |
| F13 | Timer 2 Reload Low Byte | T2RL | FF | 91 |
| F14 | Timer 2 PWM High Byte | T2PWMH | 00 | 92 |
| F15 | Timer 2 PWM Low Byte | T2PWML | 00 | 92 |
| F16 | Timer 2 Control 0 | T2CTL0 | 00 | 93 |
| F17 | Timer 2 Control 1 | T2CTL1 | 00 | 94 |

**UART1 Status 0**
U1STAT0 (F49H - Read Only)




**UART1 Control 0**
U1CTL0 (F4AH - Read/Write)



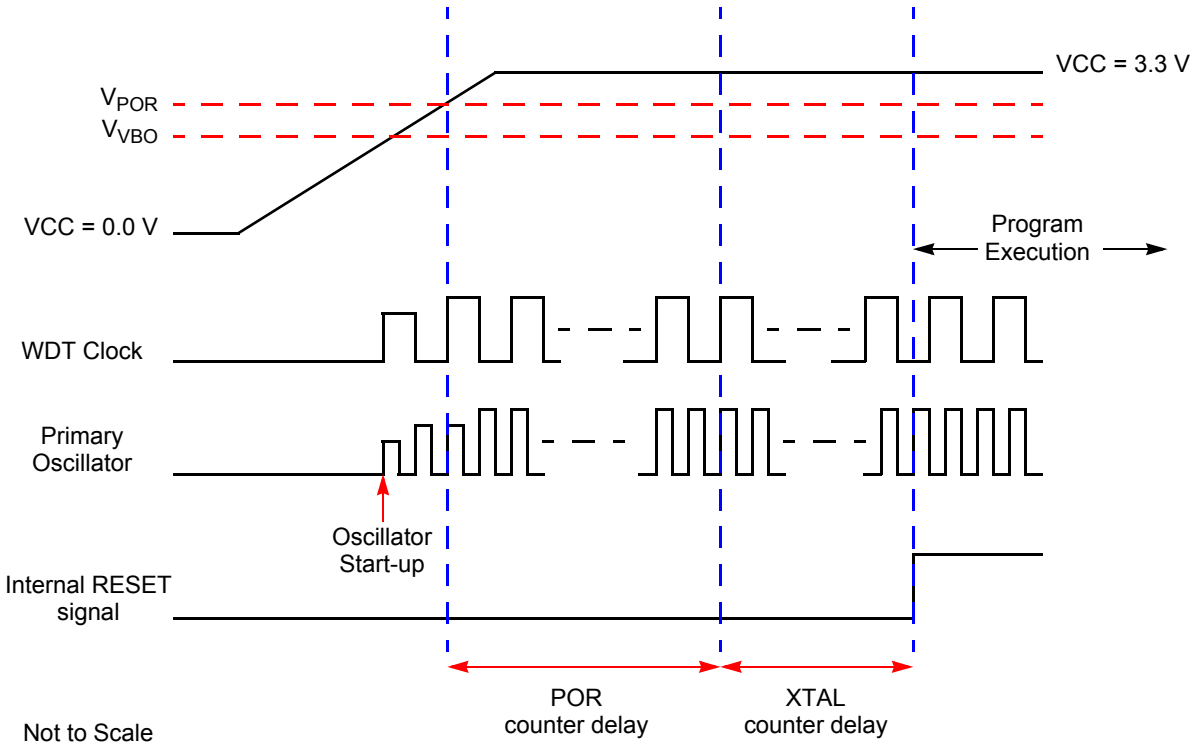



**Figure 8. Power-On Reset Operation**

## Voltage Brownout Reset

The devices in the 64K Series provide low Voltage Brownout protection. The VBO circuit senses when the supply voltage drops to an unsafe level (below the VBO threshold voltage) and forces the device into the Reset state. While the supply voltage remains below the Power-On Reset voltage threshold ($V_{POR}$), the VBO block holds the device in the Reset state.

After the supply voltage again exceeds the Power-On Reset voltage threshold, the devices progress through a full system reset sequence, as described in the Power-On Reset section. Following Power-On Reset, the POR status bit in the Watchdog Timer Control (WDTCTL) register is set to 1. Figure 9 displays Voltage Brownout operation. For the VBO and POR threshold voltages ($V_{VBO}$ and $V_{POR}$), see Electrical Characteristics on page 215.

The Voltage Brownout circuit can be either enabled or disabled during STOP mode. Operation during STOP mode is set by the VBO_A0 Option Bit. For information on configuring VBO_A0, see Option Bits page 195.

### Port A–H Data Direction Sub-Registers

The Port A–H Data Direction sub-register is accessed through the Port A–H Control register by writing 01H to the Port A–H Address register (Table 16).

**Table 16. Port A–H Data Direction Sub-Registers**

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| FIELD | DD7 | DD6 | DD5 | DD4 | DD3 | DD2 | DD1 | DD0 |
| RESET | 1 | | | | | | | |
| R/W | R/W | | | | | | | |
| ADDR | If 01H in Port A–H Address Register, accessible through Port A–H Control Register | | | | | | | |

DD[7:0]—Data Direction
These bits control the direction of the associated port pin. Port Alternate Function operation overrides the Data Direction register setting.
0 = Output. Data in the Port A–H Output Data register is driven onto the port pin.
1 = Input. The port pin is sampled and the value written into the Port A–H Input Data Register. The output driver is tri-stated.

### Port A–H Alternate Function Sub-Registers

The Port A–H Alternate Function sub-register (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 sub-registers select the alternate functions for the selected pins. To determine the alternate function associated with each port pin, see GPIO Alternate Functions on page 59.

**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 Sub-Registers**

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| FIELD | AF7 | AF6 | AF5 | AF4 | AF3 | AF2 | AF1 | AF0 |
| RESET | 0 | | | | | | | |
| R/W | R/W | | | | | | | |
| ADDR | If 02H in Port A–H Address Register, accessible through Port A–H Control Register | | | | | | | |

**Table 39. Timer 0-3 High Byte Register (TxH)**

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| FIELD | TH | | | | | | | |
| RESET | 0 | | | | | | | |
| R/W | R/W | | | | | | | |
| ADDR | F00H, F08H, F10H, F18H | | | | | | | |

**Table 40. Timer 0-3 Low Byte Register (TxL)**

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| FIELD | TL | | | | | | | |
| RESET | 0 | | | | | | | 1 |
| R/W | R/W | | | | | | | |
| ADDR | F01H, F09H, F11H, F19H | | | | | | | |

TH and TL—Timer High and Low Bytes
These 2 bytes, {TMRH[7:0], TMRL[7:0]}, contain the current 16-bit timer count value.

### Timer Reload High and Low Byte Registers

The Timer 0-3 Reload High and Low Byte (TxRH and TxRL) registers (see Table 41and Table 42 on page 92) store a 16-bit reload value, {TRH[7:0], TRL[7:0]}. Values written to the Timer Reload High Byte register are stored in a temporary holding register. When a write to the Timer Reload Low Byte register occurs, the temporary holding register value is written to the Timer High Byte register. This operation allows simultaneous updates of the 16-bit Timer Reload value.

In COMPARE mode, the Timer Reload High and Low Byte registers store the 16-bit Compare value.

**Table 41. Timer 0-3 Reload High Byte Register (TxRH)**

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| FIELD | TRH | | | | | | | |
| RESET | 1 | | | | | | | |
| R/W | R/W | | | | | | | |
| ADDR | F02H, F0AH, F12H, F1AH | | | | | | | |

## Operation

When the Infrared Endec is enabled, the transmit data from the associated on-chip UART is encoded as digital signals in accordance with the IrDA standard and output to the infrared transceiver via the TXD pin. Likewise, data received from the infrared transceiver is passed to the Infrared Endec via the RXD pin, decoded by the Infrared Endec, and then passed to the UART. Communication is half-duplex, which means simultaneous data transmission and reception is not allowed.

The baud rate is set by the UART's Baud Rate Generator and supports IrDA standard baud rates from 9600 baud to 115.2 Kbaud. Higher baud rates are possible, but do not meet IrDA specifications. The UART must be enabled to use the Infrared Endec. The Infrared Endec data rate is calculated using the following equation:

$$\text{Infrared Data Rate (bits/s)} = \frac{\text{System Clock Frequency (Hz)}}{16 \times \text{UART Baud Rate Divisor Value}}$$

### Transmitting IrDA Data

The data to be transmitted using the infrared transceiver is first sent to the UART. The UART's transmit signal (TXD) and baud rate clock are used by the IrDA to generate the modulation signal (IR_TXD) that drives the infrared transceiver. Each UART/Infrared data bit is 16-clock wide. If the data to be transmitted is 1, the IR_TXD signal remains low for the full 16-clock period. If the data to be transmitted is 0, a 3-clock high pulse is output following a 7-clock low period. After the 3-clock high pulse, a 6-clock low pulse is output to complete the full 16-clock data period. Figure 20 displays IrDA data transmission. When the Infrared Endec is enabled, the UART's TXD signal is internal to the 64K Series products while the IR_TXD signal is output through the TXD pin.

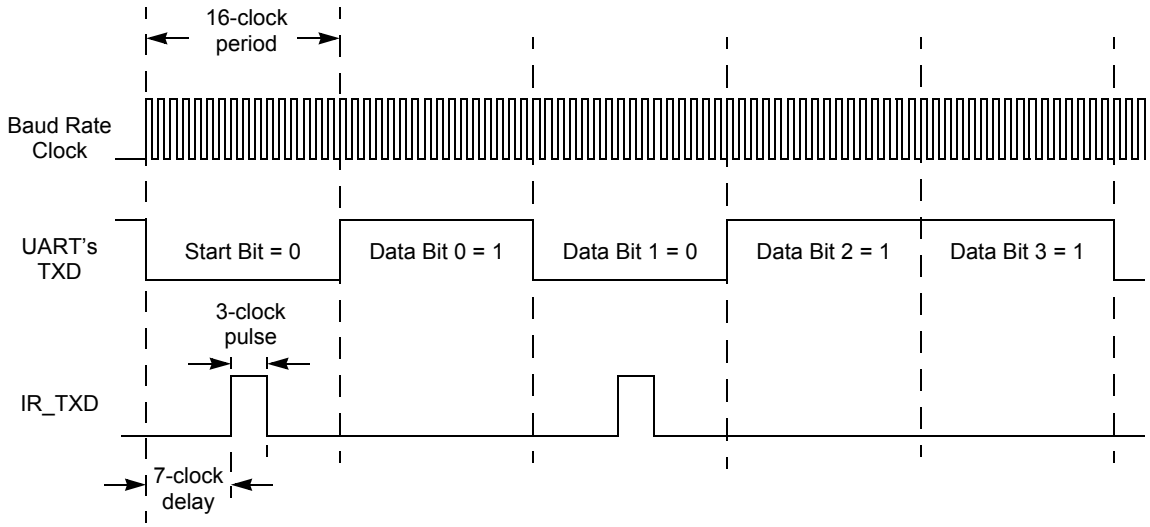



**Figure 20. Infrared Data Transmission**

## SPI Status Register

The SPI Status register (Table 65) indicates the current state of the SPI. All bits revert to their reset state if the SPIEN bit in the SPICTL register = 0.

**Table 65. SPI Status Register (SPISTAT)**

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| FIELD | IRQ | OVR | COL | ABT | Reserved | | TXST | SLAS |
| RESET | 0 | | | | | | | 1 |
| R/W | R/W* | | | | R | | | |
| ADDR | F62H | | | | | | | |
| **Note:** R/W* = Read access. Write a 1 to clear the bit to 0. | | | | | | | | |

IRQ—Interrupt Request
If SPIEN = 1, this bit is set if the STR bit in the SPICTL register is set, or upon completion of an SPI master or slave transaction. This bit does not set if SPIEN = 0 and the SPI Baud Rate Generator is used as a timer to generate the SPI interrupt.
0 = No SPI interrupt request pending.
1 = SPI interrupt request is pending.

OVR—Overrun
0 = An overrun error has not occurred.
1 = An overrun error has been detected.

COL—Collision
0 = A multi-master collision (mode fault) has not occurred.
1 = A multi-master collision (mode fault) has been detected.

ABT—Slave mode transaction abort
This bit is set if the SPI is configured in slave mode, a transaction is occurring and $\overline{SS}$ 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.

Reserved—Must be 0.

TXST—Transmit Status
0 = No data transmission currently in progress.
1 = Data transmission currently in progress.

SLAS—Slave Select
If SPI enabled as a Slave,

The first seven bits transmitted in the first byte are 11110XX. The two bits XX are the two most-significant bits of the 10-bit address. The lowest bit of the first byte transferred is the read/write control bit (=0). The transmit operation is carried out in the same manner as 7-bit addressing.

Follow the steps below for a transmit operation on a 10-bit addressed slave:

1. Software asserts the IEN bit in the $I^2C$ Control register.
2. Software asserts the TXI bit of the $I^2C$ Control register to enable Transmit interrupts.
3. The $I^2C$ interrupt asserts because the $I^2C$ Data register is empty.
4. Software responds to the TDRE interrupt by writing the first slave address byte to the $I^2C$ Data register. The least-significant bit must be 0 for the write operation.
5. Software asserts the START bit of the $I^2C$ Control register.
6. The $I^2C$ Controller sends the START condition to the $I^2C$ slave.
7. The $I^2C$ Controller loads the $I^2C$ Shift register with the contents of the $I^2C$ Data register.
8. After one bit of address is shifted out by the SDA signal, the Transmit interrupt is asserted.
9. Software responds by writing the second byte of address into the contents of the $I^2C$ Data register.
10. The $I^2C$ Controller shifts the rest of the first byte of address and write bit out the SDA signal.
11. If the $I^2C$ slave acknowledges the first address byte by pulling the SDA signal low during the next high period of SCL, the $I^2C$ Controller sets the ACK bit in the $I^2C$ Status register. Continue with step 12.

    If the slave does not acknowledge the first address byte, the $I^2C$ Controller sets the NCKI bit and clears the ACK bit in the $I^2C$ Status register. Software responds to the Not Acknowledge interrupt by setting the STOP and FLUSH bits and clearing the TXI bit. The $I^2C$ 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 $I^2C$ Controller loads the $I^2C$ Shift register with the contents of the $I^2C$ Data register.
13. The $I^2C$ Controller shifts the second address byte out the SDA signal. After the first bit has been sent, the Transmit interrupt is asserted.
14. Software responds by writing a data byte to the $I^2C$ Data register.
15. The $I^2C$ Controller completes shifting the contents of the shift register on the SDA signal.

# Direct Memory Access Controller

## Overview

The 64K 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 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).

## Configuring DMA0 and DMA1 for Data Transfer

Follow the steps below 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]}.
2. Determine the 12-bit Start and End Register File addresses. The 12-bit Start Address is given by {DMA*x*_H[3:0], DMA_START[7:0]}. The 12-bit End Address is given by {DMA*x*_H[7:4], DMA_END[7:0]}.
3. Write the Start and End Register File address high nibbles to the DMA*x* End/Start Address High Nibble register.
4. Write the lower byte of the Start Address to the DMA*x* Start/Current Address register.
5. Write the lower byte of the End Address to the DMA*x* End Address register.
6. Write to the DMA*x* Control register to complete the following:
   – Select loop or single-pass mode operation
   – Select the data transfer direction (either from the Register File RAM to the on-chip peripheral control register; or from the on-chip peripheral control register to the Register File RAM)
   – Enable the DMA*x* interrupt request, if desired
   – Select Word or Byte mode
   – Select the DMA*x* request trigger
   – Enable the DMA*x* channel

## DMA_ADC Operation

DMA_ADC transfers data from the ADC to the Register File. The sequence of operations in a DMA_ADC data transfer is:

1. ADC completes conversion on the current ADC input channel and signals the DMA controller that two-bytes of ADC data are ready for transfer.
2. DMA_ADC requests control of the system bus (address and data) from the eZ8 CPU.
3. After the eZ8 CPU acknowledges the bus request, DMA_ADC transfers the two-byte ADC output value to the Register File and then returns system bus control back to the eZ8 CPU.
4. If the current ADC Analog Input is the highest numbered input to be converted:
   – DMA_ADC resets the ADC Analog Input number to 0 and initiates data conversion on ADC Analog Input 0.
   – If configured to generate an interrupt, DMA_ADC sends an interrupt request to the Interrupt Controller

While the Flash Controller programs the Flash memory, the eZ8 CPU idles but the system clock and on-chip peripherals continue to operate. Interrupts that occur when a Programming operation is in progress are serviced once the Programming operation is complete. To exit Programming mode and lock the Flash Controller, write 00H to the Flash Control register.

User code cannot program Flash Memory on a page that lies in a protected sector. When user code writes memory locations, only addresses located in the unlocked page are programmed. Memory writes outside of the unlocked page are ignored.

**Caution:** *Each memory location must not be programmed more than twice before an erase occurs.*

Follow the steps below to program the Flash from user code:

1. Write 00H to the Flash Control register to reset the Flash Controller.
2. Write the page of memory to be programmed to the Page Select register.
3. Write the first unlock command 73H to the Flash Control register.
4. Write the second unlock command 8CH to the Flash Control register.
5. Re-write the page written in step 2 to the Page Select register.
6. Write Flash Memory using LDC or LDCI instructions to program the Flash.
7. Repeat step 6 to program additional memory locations on the same page.
8. Write 00H to the Flash Control register to lock the Flash Controller.

## 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 Page Select register identifies the page to be erased. 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. Interrupts that occur when the Page Erase operation is in progress are serviced once the Page Erase operation is complete. When the Page Erase operation is complete, the Flash Controller returns to its locked state. Only pages located in unprotected sectors can be erased.

Follow the steps below to perform a Page Erase operation:

1. Write 00H to the Flash Control register to reset the Flash Controller.
2. Write the page to be erased to the Page Select register.
3. Write the first unlock command 73H to the Flash Control register.
4. Write the second unlock command 8CH to the Flash Control register.

Reserved
These bits are reserved and must be 0.

FSTAT—Flash Controller Status
00_0000 = Flash Controller locked
00_0001 = First unlock command received
00_0010 = Second unlock command received
00_0011 = Flash Controller unlocked
00_0100 = Flash Sector Protect register selected
00_1xxx = Program operation in progress
01_0xxx = Page erase operation in progress
10_0xxx = Mass erase operation in progress

## Page Select Register

The Page Select (FPS) register (Table 94) selects one of the 128 available Flash memory pages to be erased or programmed. Each Flash Page contains 512 bytes of Flash memory. During a Page Erase operation, all Flash memory locations with the 7 most significant bits of the address given by the PAGE field are erased to FFH.

The Page Select register shares its Register File address with the Flash Sector Protect Register. The Page Select register cannot be accessed when the Flash Sector Protect register is enabled.

**Table 94. Page Select Register (FPS)**

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| FIELD | INFO_EN | PAGE | | | | | | |
| RESET | 0 | | | | | | | |
| R/W | R/W | | | | | | | |
| ADDR | FF9H | | | | | | | |

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.

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 (Table 95) 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).

**Table 95. Flash Sector Protect Register (FPROT)**

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| FIELD | SECT7 | SECT6 | SECT5 | SECT4 | SECT3 | SECT2 | SECT1 | SECT0 |
| RESET | 0 | | | | | | | |
| R/W | R/W1 | | | | | | | |
| ADDR | FF9H | | | | | | | |

**Note:** R/W1 = Register is accessible for Read operations. Register can be written to 1 only (via user code).

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.

* User code can only write bits from 0 to 1.

## Flash Frequency High and Low Byte Registers

The Flash Frequency High and Low Byte registers (Table 96 and Table 97) combine to form a 16-bit value, FFREQ, to control timing for Flash program and erase operations. The 16-bit Flash Frequency registers must be written with the system clock frequency in kHz for Program and Erase operations. Calculate the Flash Frequency value using the following equation:

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

**Caution:** *Flash programming and erasure is not supported for system clock frequencies below 20 kHz, above 20 MHz, or outside of the valid operating frequency range for the device. The Flash Frequency High and Low Byte registers must be loaded with the correct value to insure proper program and erase times.*

the On-Chip Debugger.
1 = User program code is accessible. All On-Chip Debugger commands are enabled. This setting is the default for unprogrammed (erased) Flash.

Reserved
These Option Bits are reserved for future use and must always be 1.This setting is the default for unprogrammed (erased) Flash.

FWP—Flash Write Protect (Flash version only)

| FWP | Description |
|---|---|
| 0 | Programming, Page Erase, and Mass Erase through User Code is disabled. Mass Erase is available through the On-Chip Debugger. |
| 1 | Programming, and Page Erase are enabled for all of Flash Program Memory. |

## Flash Memory Address 0001H

**Table 99. Options Bits at Flash Memory Address 0001H**

| BITS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| FIELD | Reserved | | | | | | | |
| RESET | U | | | | | | | |
| R/W | R/W | | | | | | | |
| ADDR | Program Memory 0001H | | | | | | | |

**Note:** U = Unchanged by Reset. R = Read-Only. R/W = Read/Write.

Reserved
These Option Bits are reserved for future use and must always be 1. This setting is the default for unprogrammed (erased) Flash.

## DC Characteristics

Table 106 lists the DC characteristics of the 64K Series products. All voltages are referenced to $V_{SS}$, the primary system ground.

### Table 106. DC Characteristics

| Symbol | Parameter | $T_A$ = –40 °C to 125 °C | | | Units | Conditions |
|---|---|---|---|---|---|---|
| | | Minimum | Typical | Maximum | | |
| $V_{DD}$ | Supply Voltage | 3.0 | – | 3.6 | V | |
| $V_{IL1}$ | Low Level Input Voltage | -0.3 | – | 0.3*$V_{DD}$ | V | For all input pins except $\overline{\text{RESET}}$, DBG, XIN |
| $V_{IL2}$ | Low Level Input Voltage | -0.3 | – | 0.2*$V_{DD}$ | V | For $\overline{\text{RESET}}$, DBG, and XIN. |
| $V_{IH1}$ | High Level Input Voltage | 0.7*$V_{DD}$ | – | 5.5 | V | Port A, C, D, E, F, and G pins. |
| $V_{IH2}$ | High Level Input Voltage | 0.7*$V_{DD}$ | – | $V_{DD}$+0.3 | V | Port B and H pins. |
| $V_{IH3}$ | High Level Input Voltage | 0.8*$V_{DD}$ | – | $V_{DD}$+0.3 | V | $\overline{\text{RESET}}$, DBG, and XIN pins |
| $V_{OL1}$ | Low Level Output Voltage Standard Drive | – | – | 0.4 | V | $I_{OL}$ = 2 mA; VDD = 3.0 V High Output Drive disabled. |
| $V_{OH1}$ | High Level Output Voltage Standard Drive | 2.4 | – | – | V | $I_{OH}$ = -2 mA; VDD = 3.0 V High Output Drive disabled. |
| $V_{OL2}$ | Low Level Output Voltage High Drive | – | – | 0.6 | V | $I_{OL}$ = 20 mA; VDD = 3.3 V High Output Drive enabled $T_A$ = -40 °C to +70 °C |
| $V_{OH2}$ | High Level Output Voltage High Drive | 2.4 | – | – | V | $I_{OH}$ = -20 mA; VDD = 3.3 V High Output Drive enabled; $T_A$ = -40 °C to +70 °C |
| $V_{OL3}$ | Low Level Output Voltage High Drive | – | – | 0.6 | V | $I_{OL}$ = 15 mA; VDD = 3.3 V High Output Drive enabled; $T_A$ = +70 °C to +105 °C |
| $V_{OH3}$ | High Level Output Voltage High Drive | 2.4 | – | – | V | $I_{OH}$ = 15 mA; VDD = 3.3 V High Output Drive enabled; $T_A$ = +70 °C to +105 °C |
| $V_{RAM}$ | RAM Data Retention | 0.7 | – | – | V | |
| $I_{IL}$ | Input Leakage Current | -5 | – | +5 | µA | $V_{DD}$ = 3.6 V; $V_{IN}$ = VDD or VSS[1] |
| $I_{TL}$ | Tri-State Leakage Current | -5 | – | +5 | µA | $V_{DD}$ = 3.6 V |

Figure 45 displays the typical current consumption in HALT mode while operating at 25 ºC versus the system clock frequency. All GPIO pins are configured as outputs and driven High.

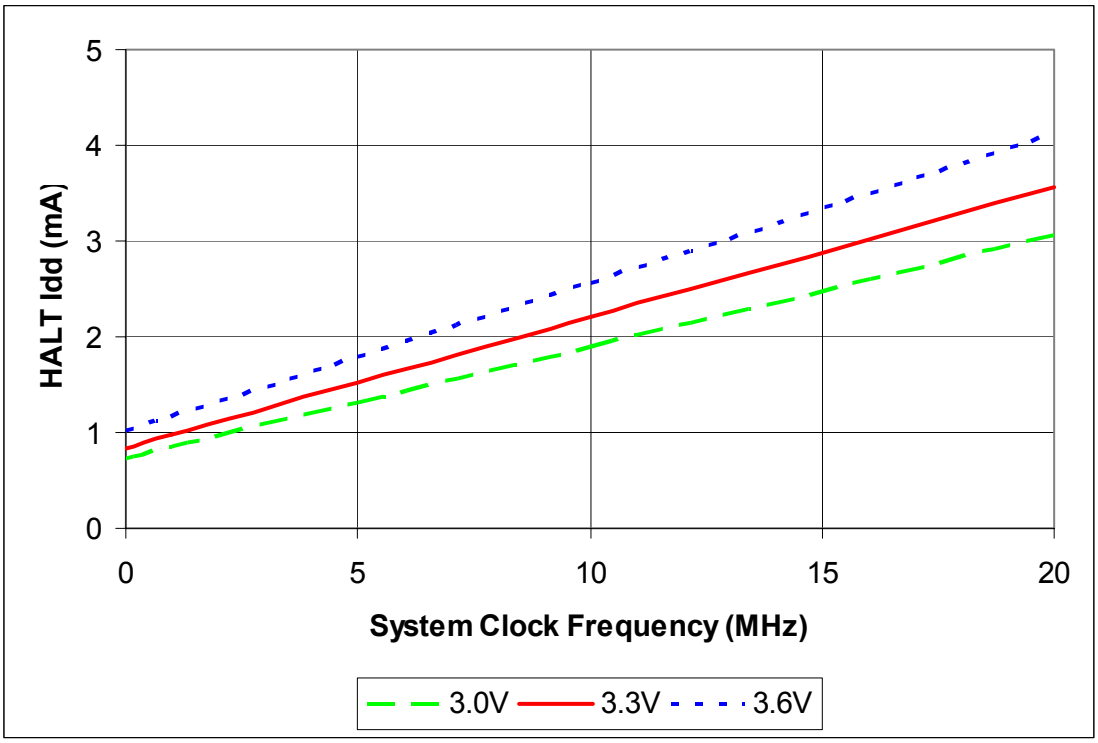



**Figure 45. Typical HALT Mode Idd Versus System Clock Frequency**

Figure 47 displays the maximum current consumption in STOP mode with the VBO and Watchdog Timer enabled versus the power supply voltage. All GPIO pins are configured as outputs and driven High.

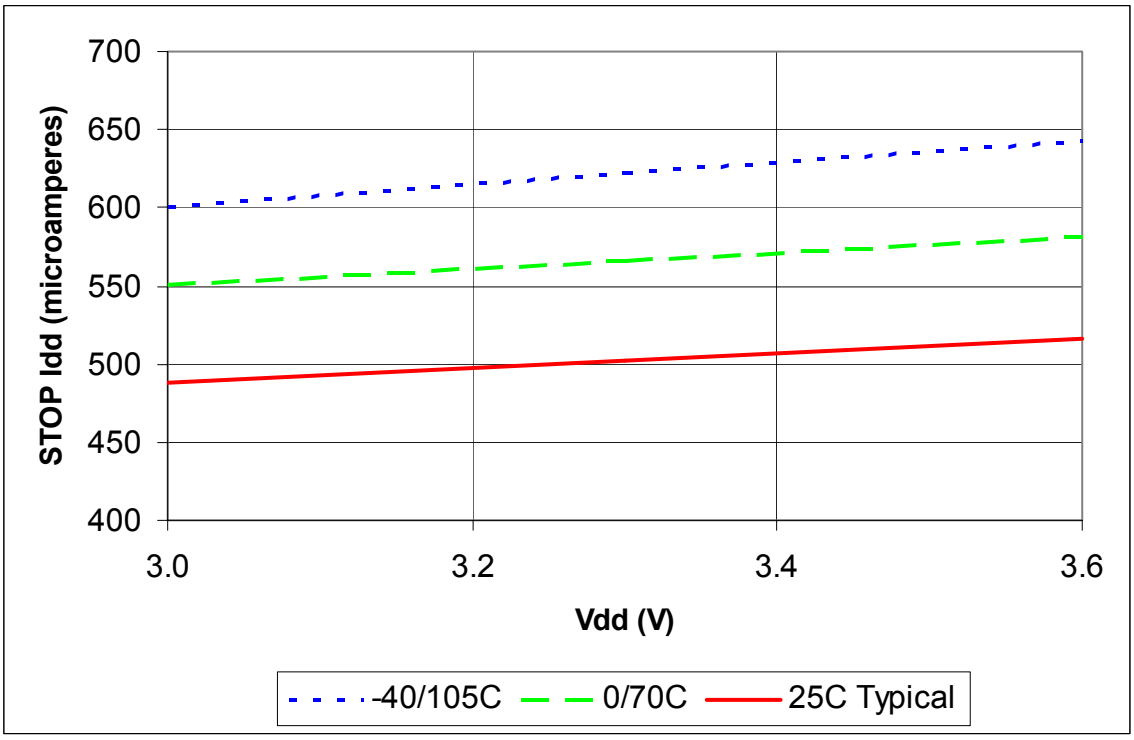



**Figure 47. Maximum STOP Mode Idd with VBO enabled versus Power Supply Voltage**

## On-Chip Peripheral AC and DC Electrical Characteristics

### Table 107. Power-On Reset and Voltage Brownout Electrical Characteristics and Timing

| Symbol | Parameter | $T_A$ = –40 °C to 125 °C | | | Units | Conditions |
|---|---|---|---|---|---|---|
| | | Minimum | Typical[1] | Maximum | | |
| $V_{POR}$ | Power-On Reset Voltage Threshold | 2.40 | 2.70 | 2.90 | V | $V_{DD} = V_{POR}$ |
| $V_{VBO}$ | Voltage Brownout Reset Voltage Threshold | 2.30 | 2.60 | 2.85 | V | $V_{DD} = V_{VBO}$ |
| | $V_{POR}$ to $V_{VBO}$ hysteresis | 50 | 100 | – | mV | |
| | Starting $V_{DD}$ voltage to ensure valid Power-On Reset. | – | $V_{SS}$ | – | V | |
| $T_{ANA}$ | Power-On Reset Analog Delay | – | 50 | – | µs | $V_{DD} > V_{POR}$; $T_{POR}$ Digital Reset delay follows $T_{ANA}$ |
| $T_{POR}$ | Power-On Reset Digital Delay | – | 6.6 | – | ms | 66 WDT Oscillator cycles (10 kHz) + 16 System Clock cycles (20 MHz) |
| $T_{VBO}$ | Voltage Brownout Pulse Rejection Period | – | 10 | – | µs | $V_{DD} < V_{VBO}$ to generate a Reset. |
| $T_{RAMP}$ | Time for VDD to transition from $V_{SS}$ to $V_{POR}$ to ensure valid Reset | 0.10 | – | 100 | ms | |

[1]Data in the typical column is from characterization at 3.3 V and 0 °C. These values are provided for design guidance only and are not tested in production.

**Table 134. Opcode Map Abbreviations**

| Abbreviation | Description | Abbreviation | Description |
|---|---|---|---|
| b | Bit position | IRR | Indirect Register Pair |
| cc | Condition code | p | Polarity (0 or 1) |
| X | 8-bit signed index or displacement | r | 4-bit Working Register |
| DA | Destination address | R | 8-bit register |
| ER | Extended Addressing register | r1, R1, Ir1, Irr1, IR1, rr1, RR1, IRR1, ER1 | Destination address |
| IM | Immediate data value | r2, R2, Ir2, Irr2, IR2, rr2, RR2, IRR2, ER2 | Source address |
| Ir | Indirect Working Register | RA | Relative |
| IR | Indirect register | rr | Working Register Pair |
| Irr | Indirect Working Register Pair | RR | Register Pair |

## M



## N



## O