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	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	https://www.e-xfl.com/product-detail/zilog/z8f2422ar020ec00tr

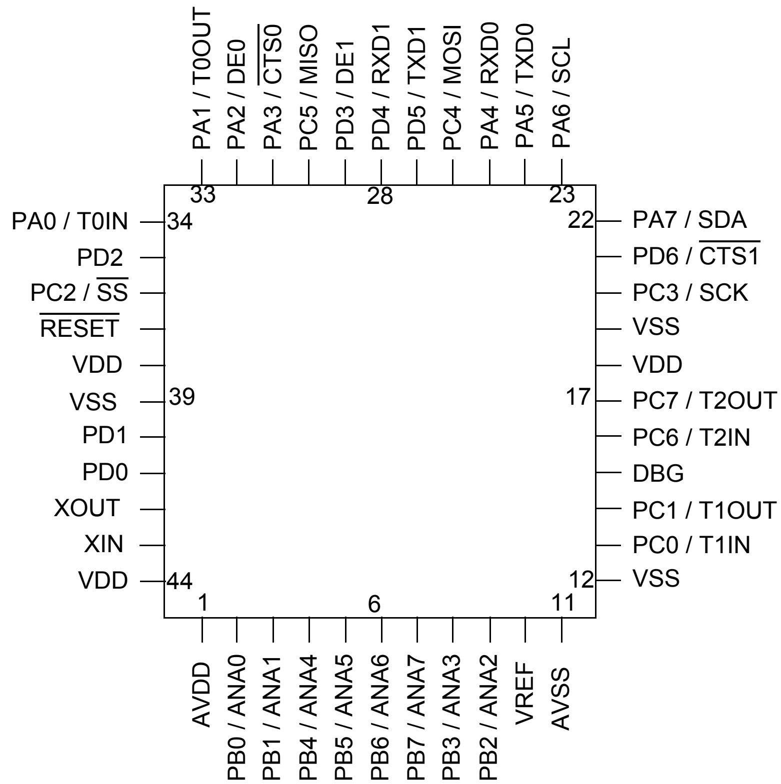

Figure 4. Z8 Encore! XP 64K Series Flash Microcontrollers in 44-Pin Low-Profile Quad Flat Package (LQFP)

Program Memory

The eZ8™ CPU supports 64 KB of Program Memory address space. The Z8 Encore! XP 64K Series Flash Microcontrollers contains 16 KB to 64 KB of on-chip Flash in the Program Memory address space, depending upon the device. Reading from Program Memory addresses outside the available Flash memory addresses returns FFH. Writing to these unimplemented Program Memory addresses produces no effect. Table 5 describes the Program Memory maps for the 64K Series products.

Table 5. Z8 Encore! XP 64K Series Flash Microcontrollers Program Memory Maps

Program Memory Address (Hex)	Function
Z8F162x Products	
0000-0001	Option Bits
0002-0003	Reset Vector
0004-0005	WDT Interrupt Vector
0006-0007	Illegal Instruction Trap
0008-0037	Interrupt Vectors*
0038-3FFF	Program Memory
Z8F242x Products	
0000-0001	Option Bits
0002-0003	Reset Vector
0004-0005	WDT Interrupt Vector
0006-0007	Illegal Instruction Trap
0008-0037	Interrupt Vectors*
0038-5FFF	Program Memory
Z8F322x Products	
0000-0001	Option Bits
0002-0003	Reset Vector
0004-0005	WDT Interrupt Vector
0006-0007	Illegal Instruction Trap
0008-0037	Interrupt Vectors*
0038-7FFF	Program Memory
Z8F482x Products	

I²C Status
I2CSTAT (F51H - Read Only)

D7 D6 D5 D4 D3 D2 D1 D0

- D0: NACK Interrupt
  0 = No action required to service NAK
  1 = START/STOP not set after NAK
- D1: Data Shift State
  0 = Data is not being transferred
  1 = Data is being transferred
- D2: Transmit Address State
  0 = Address is not being transferred
  1 = Address is being transferred
- D3: Read
  0 = Write operation
  1 = Read operation
- D4: 10-Bit Address
  0 = 7-bit address being transmitted
  1 = 10-bit address being transmitted
- D5: Acknowledge
  0 = Acknowledge not transmitted/received
  1 = For last byte, Acknowledge was transmitted/received
- D6: Receive Data Register Full
  0 = I2C has not received data
  1 = Data register contains received data
- D7: Transmit Data Register Empty
  0 = Data register is full
  1 = Data register is empty

I²C Control
I2CCTL (F52H - Read/Write)

D7 D6 D5 D4 D3 D2 D1 D0

- D0: I2C Signal Filter Enable
  0 = Digital filtering disabled
  1 = Low-pass digital filters enabled on SDA and SCL input signals
- D1: Flush Data
  0 = No effect
  1 = Clears I2C Data register
- D2: Send NAK
  0 = Do not send NAK
  1 = Send NAK after next byte received from slave
- D3: Enable TDRE Interrupts
  0 = Do not generate an interrupt when the I2C Data register is empty
  1 = Generate an interrupt when the I2C Transmit Data register is empty
- D4: Baud Rate Generator Interrupt
  0 = Interrupts behave as set by I2C control
  1 = BRG generates an interrupt when it counts down to zero
- D5: Send Stop Condition
  0 = Do not issue Stop condition after data transmission is complete
  1 = Issue Stop condition after data transmission is complete
- D6: Send Start Condition
  0 = Do not send Start Condition
  1 = Send Start Condition
- D7: I2C Enable
  0 = I2C is disabled
  1 = I2C is enabled

I2C Baud Rate Generator High Byte
I2CBRH (F53H - Read/Write)

D7 D6 D5 D4 D3 D2 D1 D0

- D7–D0: I2C Baud Rate divisor [15:8]

I2C Baud Rate Generator Low Byte
I2CBRL (F54H - Read/Write)

D7 D6 D5 D4 D3 D2 D1 D0

- D7–D0: I2C Baud Rate divisor [7:0]

DMA_ADC Control
DMAACTL (FBEH - Read/Write)

D7 D6 D5 D4 D3 D2 D1 D0

- D3–D0: ADC Analog Input Number
  0000 = Analog input 0 updated
  0001 = Analog input 0-1 updated
  0010 = Analog input 0-2 updated
  0011 = Analog input 0-3 updated
  0100 = Analog input 0-4 updated
  0101 = Analog input 0-5 updated
  0100 = Analog input 0-6 updated
  0101 = Analog input 0-7 updated
  1000 = Analog input 0-8 updated
  1001 = Analog input 0-9 updated
  1010 = Analog input 0-10 updated
  1011 = Analog inputs 0-11 updated
  11xx = Reserved
- D5–D4: Reserved
- D6: Interrupt request enable
  0 = DMA_ADC does not generate interrupt requests
  1 = DMA_ADC generates interrupt requests after last analog input
- D7: DMA_ADC Enable
  0 = DMA_ADC is disabled
  1 = DMA_ADC is enabled

DMA Status
DMAA_STAT (FBFH - Read Only)

D7 D6 D5 D4 D3 D2 D1 D0

- D0: DMA0 Interrupt Request Indicator
  0 = DMA0 is not the source of the IRQ
  1 = DMA0 is the source of the IRQ
- D1: DMA1 Interrupt Request Indicator
  0 = DMA1 is not the source of the IRQ
  1 = DMA1 is the source of the IRQ
- D2: DMA_ADC Interrupt Request
  0 = DMA_ADC is not the source of the IRQ
  1 = DMA_ADC is the source of the IRQ
- D3: Reserved
- D7–D4: Current ADC analog input
  Identifies the analog input the ADC is currently converting

Interrupt Request 0
IRQ0 (FC0H - Read/Write)

D7 D6 D5 D4 D3 D2 D1 D0

- D0: ADC Interrupt Request
- D1: SPI Interrupt Request
- D2: I2C Interrupt Request
- D3: UART 0 Transmitter Interrupt
- D4: UART 0 Receiver Interrupt Request
- D5: Timer 0 Interrupt Request
- D6: Timer 1 Interrupt Request
- D7: Timer 2 Interrupt Request

For all of the above peripherals:
0 = Peripheral IRQ is not pending
1 = Peripheral IRQ is awaiting service

IRQ0 Enable High Bit
IRQ0ENH (FC1H - Read/Write)

D7 D6 D5 D4 D3 D2 D1 D0

- D0: ADC IRQ Enable Hit Bit
- D1: SPI IRQ Enable High Bit
- D2: I2C IRQ Enable High Bit
- D3: UART 0 Transmitter IRQ Enable
- D4: UART 0 Receiver IRQ Enable High
- D5: Timer 0 IRQ Enable High Bit
- D6: Timer 1 IRQ Enable High Bit
- D7: Timer 2 IRQ Enable High Bit

where *x* indicates the specific GPIO Port C pin number (0 through 3).

IRQ0 Enable High and Low Bit Registers

The IRQ0 Enable High and Low Bit registers (see Table 28 and Table 29 on page 75) form a priority encoded enabling for interrupts in the Interrupt Request 0 register. Priority is generated by setting bits in each register. Table 27 describes the priority control for IRQ0.

Table 27. IRQ0 Enable and Priority Encoding

IRQ0ENH[*x*]	IRQ0ENL[*x*]	Priority	Description
0	0	Disabled	Disabled
0	1	Level 1	Low
1	0	Level 2	Nominal
1	1	Level 3	High

Note: where *x* indicates the register bits from 0 through 7.

Table 28. IRQ0 Enable High Bit Register (IRQ0ENH)

BITS	7	6	5	4	3	2	1	0
FIELD	T2ENH	T1ENH	T0ENH	U0RENH	U0TENH	I2CENH	SPIENH	ADCENH
RESET	0							
R/W	R/W							
ADDR	FC1H							

T2ENH—Timer 2 Interrupt Request Enable High Bit
T1ENH—Timer 1 Interrupt Request Enable High Bit
T0ENH—Timer 0 Interrupt Request Enable High Bit
U0RENH—UART 0 Receive Interrupt Request Enable High Bit
U0TENH—UART 0 Transmit Interrupt Request Enable High Bit
I2CENH—I^2C Interrupt Request Enable High Bit
SPIENH—SPI Interrupt Request Enable High Bit
ADCENH—ADC Interrupt Request Enable High Bit

Table 33. IRQ2 Enable and Priority Encoding (Continued)

IRQ2ENH[*x*]	IRQ2ENL[*x*]	Priority	Description
1	1	Level 3	High

Note: where *x* indicates the register bits from 0 through 7.

Table 34. IRQ2 Enable High Bit Register (IRQ2ENH)

BITS	7	6	5	4	3	2	1	0
FIELD	T3ENH	U1RENH	U1TENH	DMAENH	C3ENH	C2ENH	C1ENH	C0ENH
RESET	0							
R/W	R/W							
ADDR	FC7H							

T3ENH—Timer 3 Interrupt Request Enable High Bit
U1RENH—UART 1 Receive Interrupt Request Enable High Bit
U1TENH—UART 1 Transmit Interrupt Request Enable High Bit
DMAENH—DMA Interrupt Request Enable High Bit
C3ENH—Port C3 Interrupt Request Enable High Bit
C2ENH—Port C2 Interrupt Request Enable High Bit
C1ENH—Port C1 Interrupt Request Enable High Bit
C0ENH—Port C0 Interrupt Request Enable High Bit

Table 35. IRQ2 Enable Low Bit Register (IRQ2ENL)

BITS	7	6	5	4	3	2	1	0
FIELD	T3ENL	U1RENL	U1TENL	DMAENL	C3ENL	C2ENL	C1ENL	C0ENL
RESET	0							
R/W	R/W							
ADDR	FC8H							

T3ENL—Timer 3 Interrupt Request Enable Low Bit
U1RENL—UART 1 Receive Interrupt Request Enable Low Bit
U1TENL—UART 1 Transmit Interrupt Request Enable Low Bit
DMAENL—DMA Interrupt Request Enable Low Bit
C3ENL—Port C3 Interrupt Request Enable Low Bit
C2ENL—Port C2 Interrupt Request Enable Low Bit

Timers

Overview

The 64K Series products contain up to four 16-bit reloadable timers that can be used for timing, event counting, or generation of pulse width modulated signals. The timers' features include:

- 16-bit reload counter
- Programmable prescaler with prescale values from 1 to 128
- PWM output generation
- Capture and compare capability
- External input pin for timer input, clock gating, or capture signal. External input pin signal frequency is limited to a maximum of one-fourth the system clock frequency.
- Timer output pin
- Timer interrupt

In addition to the timers described in this chapter, the Baud Rate Generators for any unused UART, SPI, or I^2C peripherals may also be used to provide basic timing functionality. For information on using the Baud Rate Generators as timers, see the respective serial communication peripheral. Timer 3 is unavailable in the 44-pin package devices.

Architecture

Figure 12 displays the architecture of the timers.

Follow the steps below 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 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 1 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 1 register determines if the Capture occurs on a rising edge or a falling edge of the Timer Input signal. When the Capture event occurs, an interrupt is generated and the timer continues counting.

Timer 0-3 Control 1 Registers

The Timer 0-3 Control 1 (TxCTL1) registers enable/disable the timers, set the prescaler value, and determine the timer operating mode.

Table 46. Timer 0-3 Control 1 Register (TxCTL1)

BITS	7	6	5	4	3	2	1	0
FIELD	TEN	TPOL	PRES			TMODE		
RESET	0							
R/W	R/W							
ADDR	F07H, F0FH, F17H, F1FH							

TEN—Timer Enable
0 = Timer is disabled.
1 = Timer enabled to count.

TPOL—Timer Input/Output Polarity
Operation of this bit is a function of the current operating mode of the timer.

ONE-SHOT mode

When the timer is disabled, the Timer Output signal is set to the value of this bit. When the timer is enabled, the Timer Output signal is complemented upon timer Reload.

CONTINUOUS mode

When the timer is disabled, the Timer Output signal is set to the value of this bit. When the timer is enabled, the Timer Output signal is complemented upon timer Reload.

COUNTER mode

When the timer is disabled, the Timer Output signal is set to the value of this bit. When the timer is enabled, the Timer Output signal is complemented upon timer Reload.

0 = Count occurs on the rising edge of the Timer Input signal.
1 = Count occurs on the falling edge of the Timer Input signal.

PWM mode

0 = Timer Output is forced Low (0) when the timer is disabled. When enabled, the Timer Output is forced High (1) upon PWM count match and forced Low (0) upon Reload.

UART

Overview

The Universal Asynchronous Receiver/Transmitter (UART) is a full-duplex communication channel capable of handling asynchronous data transfers. The UART uses a single 8-bit data mode with selectable parity. Features of the UART include:

- 8-bit asynchronous data transfer
- Selectable even- and odd-parity generation and checking
- Option of one or two Stop bits
- Separate transmit and receive interrupts
- Framing, parity, overrun and break detection
- Separate transmit and receive enables
- 16-bit Baud Rate Generator (BRG)
- Selectable MULTIPROCESSOR (9-bit) mode with three configurable interrupt schemes
- Baud Rate Generator timer mode
- Driver Enable output for external bus transceivers

Architecture

The UART consists of three primary functional blocks: Transmitter, Receiver, and Baud rate generator. The UART's transmitter and receiver function independently, but employ the same baud rate and data format. Figure 13 on page 104 displays the UART architecture.

The Master and Slave are each capable of exchanging a character of data during a sequence of NUMBITS clock cycles (see NUMBITS field in the SPI Mode Register on page 140). In both Master and Slave SPI devices, data is shifted on one edge of the SCK and is sampled on the opposite edge where data is stable. Edge polarity is determined by the SPI phase and polarity control.

Slave Select

The active Low Slave Select ($\overline{SS}$) input signal selects a Slave SPI device. $\overline{SS}$ must be Low prior to all data communication to and from the Slave device. $\overline{SS}$ must stay Low for the full duration of each character transferred. The $\overline{SS}$ signal may stay Low during the transfer of multiple characters or may deassert between each character.

When the SPI is configured as the only Master in an SPI system, the $\overline{SS}$ pin can be set as either an input or an output. For communication between the Z8F642x family Z8R642x family device's SPI Master and external Slave devices, the $\overline{SS}$ signal, as an output, can assert the $\overline{SS}$ input pin on one of the Slave devices. Other GPIO output pins can also be employed to select external SPI Slave devices.

When the SPI is configured as one Master in a multi-master SPI system, the $\overline{SS}$ pin must be set as an input. The $\overline{SS}$ input signal on the Master must be High. If the $\overline{SS}$ signal goes Low (indicating another Master is driving the SPI bus), a Collision error Flag is set in the SPI Status register.

SPI Clock Phase and Polarity Control

The SPI supports four combinations of serial clock phase and polarity using two bits in the SPI Control register. The clock polarity bit, `CLKPOL`, selects an active high or active Low clock and has no effect on the transfer format. Table 62 lists the SPI Clock Phase and Polarity Operation parameters. The clock phase bit, PHASE, selects one of two fundamentally different transfer formats. For proper data transmission, the clock phase and polarity must be identical for the SPI Master and the SPI Slave. The Master always places data on the MOSI line a half-cycle before the receive clock edge (SCK signal), in order for the Slave to latch the data.

Table 62. SPI Clock Phase (PHASE) and Clock Polarity (CLKPOL) Operation

PHASE	CLKPOL	SCK Transmit Edge	SCK Receive Edge	SCK Idle State
0	0	Falling	Rising	Low
0	1	Rising	Falling	High
1	0	Rising	Falling	Low
1	1	Falling	Rising	High

reading the I^2C Data register. Once the I^2C data register has been read, the I^2C reads the next data byte.

Address Only Transaction with a 7-bit Address

In the situation where software determines if a slave with a 7-bit address is responding without sending or receiving data, a transaction can be done which only consists of an address phase. Figure 28 displays this 'address only' transaction to determine if a slave with a 7-bit address will acknowledge. As an example, this transaction can be used after a 'write' has been done to a EEPROM to determine when the EEPROM completes its internal write operation and is once again responding to I^2C transactions. If the slave does not Acknowledge, the transaction can be repeated until the slave does Acknowledge.

S	Slave Address	W = 0	$A/\overline{A}$	P

Figure 28. 7-Bit Address Only Transaction Format

Follow the steps below for an address only transaction to a 7-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 (TDRE = 1)
4. Software responds to the `TDRE` bit by writing a 7-bit slave address plus write bit (=0) to the I^2C Data register. As an alternative this could be a read operation instead of a write operation.
5. Software sets the START and STOP bits of the I^2C Control register and clears the TXI bit.
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. Software polls the STOP bit of the I^2C Control register. Hardware deasserts the STOP bit when the address only transaction is completed.
9. Software checks the ACK bit of the I^2C Status register. If the slave acknowledged, the ACK bit is = 1. If the slave does not acknowledge, the ACK bit is = 0. The NCKI interrupt does not occur in the not acknowledge case because the STOP bit was set.

IEN—I²C Enable
1 = The I²C transmitter and receiver are enabled.
0 = The I²C transmitter and receiver are disabled.

START—Send Start Condition
This bit sends the Start condition. Once asserted, it is cleared by the I²C Controller after it sends the START condition or if the `IEN` bit is deasserted. If this bit is 1, it cannot be cleared to 0 by writing to the register. After this bit is set, the Start condition is sent if there is data in the I²C Data or I²C Shift register. If there is no data in one of these registers, the I²C Controller waits until the Data register is written. If this bit is set while the I²C Controller is shifting out data, it generates a START condition after the byte shifts and the acknowledge phase completes. If the `STOP` bit is also set, it also waits until the STOP condition is sent before the sending the START condition.

STOP—Send Stop Condition
This bit causes the I²C Controller to issue a Stop condition after the byte in the I²C Shift register has completed transmission or after a byte has been received in a receive operation. Once set, this bit is reset by the I²C Controller after a Stop condition has been sent or by deasserting the `IEN` bit. If this bit is 1, it cannot be cleared to 0 by writing to the register.

BIRQ—Baud Rate Generator Interrupt Request
This bit allows the I²C Controller to be used as an additional timer when the I²C Controller is disabled. This bit is ignored when the I²C Controller is enabled.
1 = An interrupt occurs every time the baud rate generator counts down to one.
0 = No baud rate generator interrupt occurs.

TXI—Enable TDRE interrupts
This bit enables the transmit interrupt when the I²C Data register is empty (TDRE = 1).
1 = Transmit interrupt (and DMA transmit request) is enabled.
0 = Transmit interrupt (and DMA transmit request) is disabled.

NAK—Send NAK
This bit sends a Not Acknowledge condition after the next byte of data has been read from the I²C slave. Once asserted, it is deasserted after a Not Acknowledge is sent or the `IEN` bit is deasserted. If this bit is 1, it cannot be cleared to 0 by writing to the register.

FLUSH—Flush Data
Setting this bit to 1 clears the I²C Data register and sets the TDRE bit to 1. This bit allows flushing of the I²C Data register when a Not Acknowledge interrupt is received after the data has been sent to the I²C Data register. Reading this bit always returns 0.

FILTEN—I²C Signal Filter Enable
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.

If the current ADC Analog Input is not the highest numbered input to be converted, DMA_ADC initiates data conversion in the next higher numbered ADC Analog Input.

Configuring DMA_ADC for Data Transfer

Follow the steps below to configure and enable DMA_ADC:

1. Write the DMA_ADC Address register with the 7 most-significant bits of the Register File address for data transfers.
2. Write to the DMA_ADC Control register to complete the following:
   – Enable the DMA_ADC interrupt request, if desired
   – Select the number of ADC Analog Inputs to convert
   – Enable the DMA_ADC channel

Caution: *When using the DMA_ADC to perform conversions on multiple ADC inputs, the Analog-to-Digital Converter must be configured for SINGLE-SHOT mode. If the* `ADC_IN` *field in the DMA_ADC Control Register is greater than 000b, the ADC must be in SINGLE-SHOT mode.*

CONTINUOUS mode operation of the ADC can only be used in conjunction with DMA_ADC if the `ADC_IN` *field in the DMA_ADC Control Register is reset to 000b to enable conversion on ADC Analog Input 0 only.*

DMA Control Register Definitions

DMA*x* Control Register

The DMA*x* Control register (see Table 77 on page 167) enables and selects the mode of operation for DMA*x*.

Table 77. DMA*x* Control Register (DMA*x*CTL)

BITS	7	6	5	4	3	2	1	0
FIELD	DEN	DLE	DDIR	IRQEN	WSEL	RSS		
RESET	0							
R/W	R/W							
ADDR	FB0H, FB8H							

DEN—DMA*x* Enable
0 = DMA*x* is disabled and data transfer requests are disregarded.

Table 96. Flash Frequency High Byte Register (FFREQH)

BITS	7	6	5	4	3	2	1	0
FIELD	FFREQH							
RESET	0							
R/W	R/W							
ADDR	FFAH							

Table 97. Flash Frequency Low Byte Register (FFREQL)

BITS	7	6	5	4	3	2	1	0
FIELD	FFREQL							
RESET	0							
R/W	R/W							
ADDR	FFBH							

FFREQH and FFREQL—Flash Frequency High and Low Bytes
These 2 bytes, {FFREQH[7:0], FFREQL[7:0]}, contain the 16-bit Flash Frequency value.

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

		T_A = –40 °C to 125 °C				
Symbol	**Parameter**	**Minimum**	**Typical**	**Maximum**	**Units**	**Conditions**
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 RESET, DBG, XIN
V_{IL2}	Low Level Input Voltage	-0.3	–	0.2*V_{DD}	V	For 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	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 43 displays the typical active mode current consumption while operating at 25 ºC versus the system clock frequency. All GPIO pins are configured as outputs and driven High.

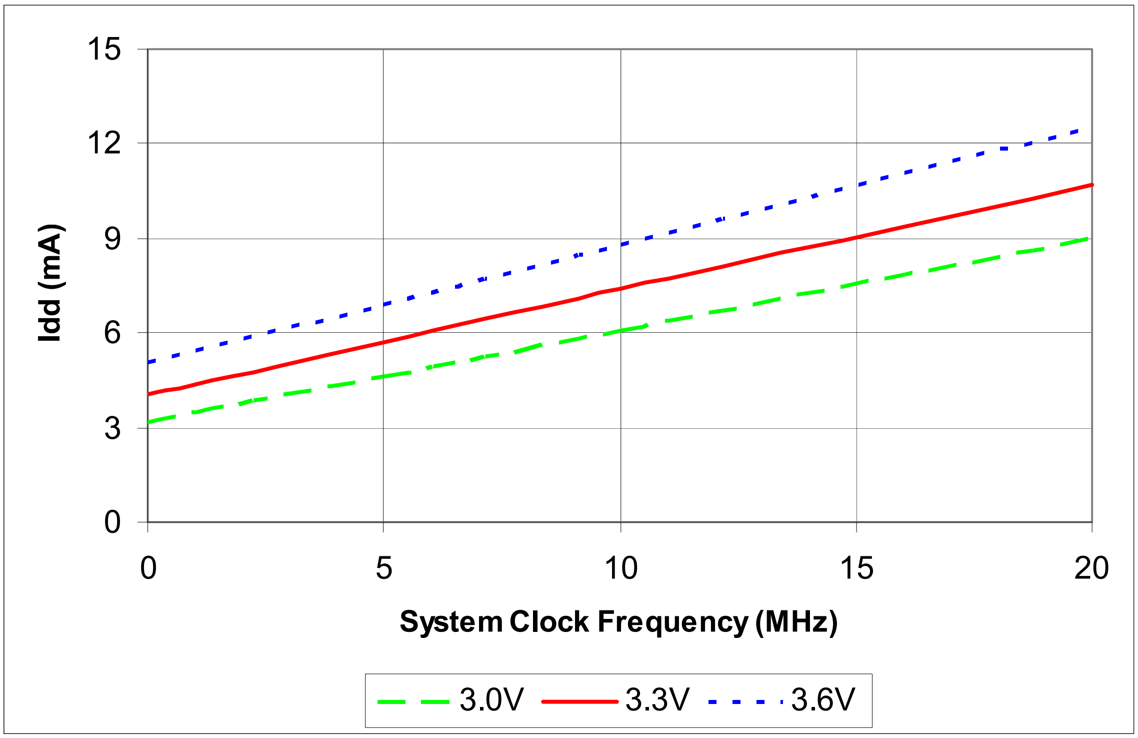

Figure 43. Typical Active Mode Idd Versus System Clock Frequency