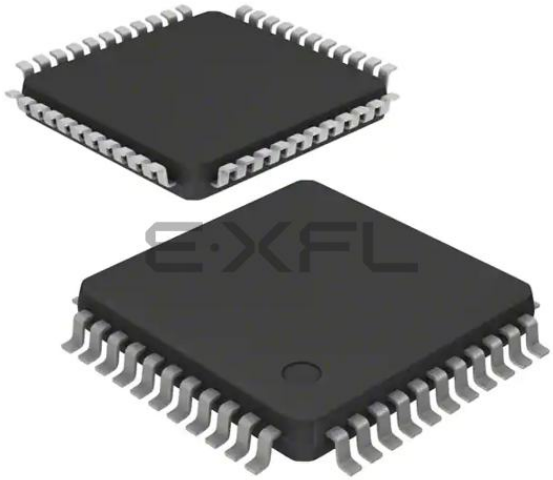

Welcome to E-XFL.COM

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	Active
Core Processor	eZ8
Core Size	8-Bit
Speed	24MHz
Connectivity	DALI, DMX, I²C, LINbus, SPI, UART/USART, USB
Peripherals	DMA, LVD, POR, PWM, WDT
Number of I/O	36
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	3.75K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 10x12b; D/A 1x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-LQFP
Supplier Device Package	44-LQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/zilog/z8f3281an024xk2246

1.4.8. Analog Comparators

The analog comparators compare the signal at an input pin or at other internal signal sources with either an internal programmable voltage reference, an internal fixed reference, the DAC output or a second input pin. The comparator outputs are used to either drive an output pin, the Event System, or to generate an interrupt. The comparators can function in all operating modes including Stop Mode.

1.4.9. Temperature Sensor

The temperature sensor produces an analog output proportional to the device temperature. The signal is sent either to the ADC or to the analog comparators. The temperature sensor can function in all operating modes including Stop Mode.

1.4.10. Low-Voltage Detector

The low-voltage detector generates an interrupt when the supply voltage drops below a user-programmable level.

1.4.11. USB 2.0

The Full-Speed Universal Serial Bus (USB 2.0) device provides eight endpoints supporting bulk, control, and interrupt transfers. It contains an integrated USB-PHY and a PLL for transmit clocking.

1.4.12. Enhanced SPI

The enhanced SPI is a full-duplex, buffered, synchronous character-oriented channel which supports a four-wire interface.

1.4.13. UART with LIN, DALI, and DMX

A full-duplex 9-bit UART provides serial, asynchronous communication, and supports the Local Interconnect Network (LIN) and Digital Addressable Lighting Interface (DALI) serial communications protocols as well as Asynchronous Serial Digital Data Transmission Standard for Controlling Lighting Equipment and Accessories (DMX). The UART supports 8-bit and 9-bit data modes, selectable parity, and an efficient bus transceiver Driver Enable signal for controlling a multi-transceiver bus, such as a RS-485. The LIN bus is a cost-efficient, single-master, multiple-slave organization which supports speed up to 20 kilobits. Manchester encoding is supported for the DALI protocol.

1.4.14. Master/Slave I^2C

The inter-integrated circuit (I^2C) controller makes the F6482 Series products compatible with the I^2C protocol. The I^2C controller consists of two bidirectional bus lines:

- Serial data (SDA) line

7.10.11. Port A–J Input Data Registers

Reading from the Port A–J Input Data registers, shown in Table 33, returns the sampled values from the corresponding port pins. The Port A–J Input Data registers are read-only. The value returned for any unused ports is 0. Unused ports include those missing on the packages other than the 80-pin package.

Table 33. Port A–J Input Data Registers (P*x*IN)

Bit	7	6	5	4	3	2	1	0
Field	PIN7	PIN6	PIN5	PIN4	PIN3	PIN2	PIN1	PIN0
Reset	X	X	X	X	X	X	X	X
R/W	R	R	R	R	R	R	R	R
Address	Port A @ FD2h, Port B @ FD6h, Port C @ FDAh, Port D @ FDEh, Port E @ FE2h, Port F @ FE6h, Port G @ FEAh, Port H @ FEEh, Port J @ FBEh							

Note: *x* = A, B, C, D, E, F, G, H, or J.

Bit	Description
[7:0] P*x*IN	**Port Input Data** Sampled data from the corresponding port pin input. 0: Input data is logical 0 (Low). 1: Input data is logical 1 (High).

Bit	Description (Continued)
[1:0] FLLNDIVL	**Frequency Locked Loop (FLL) N-Divider Low Byte** 00–11: Least significant bits of the FLL N-divider control word, bits 1:0. To meet minimum and maximum FLL operating frequency requirements, {FLLNDIVH, FLLNDIVL} should not be less than 01Fh (1MHz) nor exceed 2DCh (24MHz). {FLLNDIVH, FLLNDIV} function as follows: 000h–01Eh=Reserved, 01Fh=Multiply PCLK by 31. 020h=Multiply PCLK by 32. 021h=Multiply PCLK by 33. • • • 2DAh: Multiply PCLK by 730. 2DBh: Multiply PCLK by 731. 2DCh: Multiply PCLK by 732. 2FBh–3FFh: Reserved.

8.11.6. Clock Control 5 Register

The Clock Control 5 Register (CLKCTL5), shown in Table 43, enables/disables various clock sources and selects controls the DCO and FLL. Entry into Stop Mode and Stop-Mode Recovery (SMR) affects the state of some bits in this register and leaves others unchanged. Before writing CLKCTL5, the clock control registers must be unlocked as described in the Clock System Control Register Unlocking/Locking section on page 103.

Table 43. Clock Control 5 Register (CLKCTL5)

Bit	7	6	5	4	3	2	1	0
Field	Reserved	FLLIRQE	FLLLL	FLLRDY	FLADONE	DCOEN	SEEDSEL	FLLEN
Reset	0	0	0	0	0	1	0	1
STOP*	no change	no change	0	0	no change	0	no change	0
SMR*	no change	no change	0	0	no change	1	no change	0
R/W	R	R/W	R	R	R	R/W	R/W	R/W
Address	F87h							
Notes: *STOP = Effect of entering Stop Mode; SMR = Effect of Stop-Mode Recovery.								

Bit	Description
[7]	**Reserved** This bit is reserved and must be programmed to 0.

Note: Some port interrupts are not available on the Z8Fxx82 64-pin package and the Z8Fxx81 32-pin package. The LCD, USB, SPI 1, Comparator 1, Multi-Channel Timer and UART 1 interrupt sources are unavailable on devices not containing those peripherals.

Table 51. Trap and Interrupt Vectors in Order of Priority

Priority*	Program Memory Vector Address	Interrupt or Trap Source
Highest	0002h	Reset (not an interrupt)
	0004h	Watchdog Timer; see the Watchdog Timer chapter on page 206
	0048h	System Clock Fail Trap (not an interrupt, the Clock System chapter on page 96)
	004Ah	Watchdog Timer Oscillator Fail Trap (not an interrupt, the Clock System chapter on page 96)
	0006h	Illegal Instruction Trap (not an interrupt)
	0008h	Timer 2
	000Ah	Timer 1
	000Ch	Timer 0
	000Eh	UART 0 receiver
	0010h	UART 0 transmitter
	0012h	USB
	0014h	USB Resume
	0016h	I^2C
	0018h	SPI 1
	001Ah	DAC
	001Ch	DMA1
	001Eh	DMA0
	0020h	ADC
	0022h	SPI 0
	0024h	LCD
	0026h	RTC
	0028h	Port A7, selectable rising or falling input edge or LVD. To learn more, see the Reset, Stop-Mode Recovery and Low-Voltage Detection chapter on page 38.

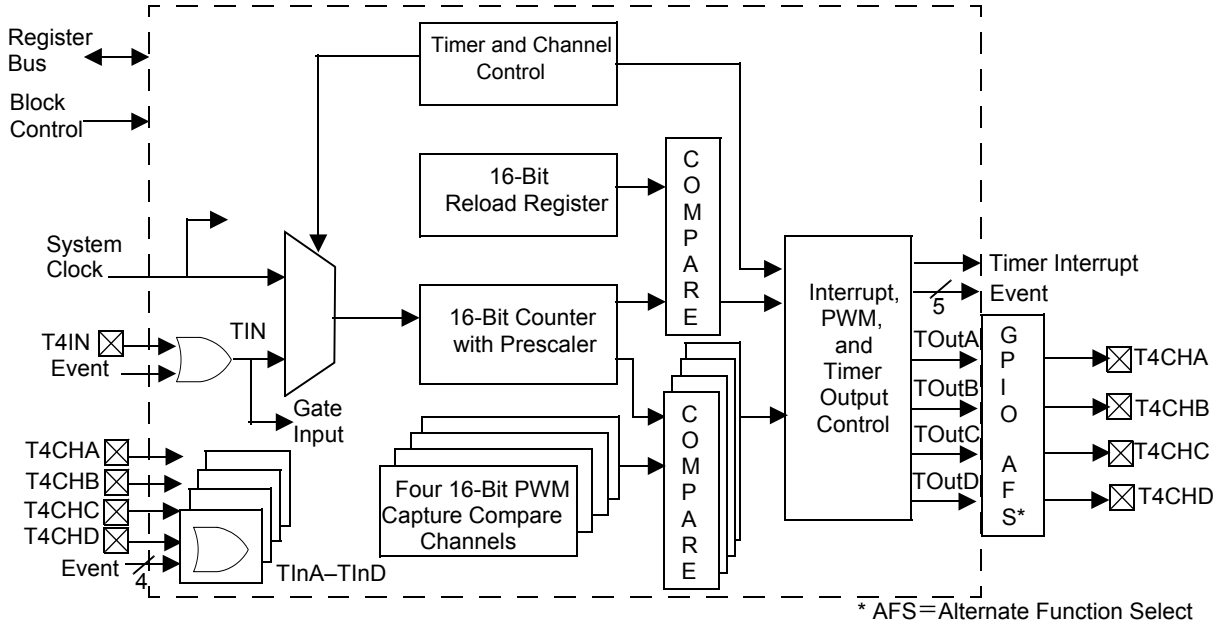

Figure 21. Multi-Channel Timer Block Diagram

11.2. Timer Operation

11.2.1. Multi-Channel Timer Counter

The Multi-Channel Timer is based around a 16-bit up/down counter. The counter, depending on the timer mode, counts up or down with each rising edge of the clock signal. Timer Counter registers MCTH and MCTL can be read/written by software.

11.2.2. Inputs and Outputs

Each GPIO alternate function (T4CHA–T4CHD and T4IN) is logically ORed with a corresponding Event System signal to form an input to the Multi-Channel Timer. Typically, only one of these two signals, either the GPIO input or Event System input, is configured to be active at a given time. Any inactive input is held at logic 0. To learn more about selecting a GPIO input using GPIO alternate function registers, refer to the General-Purpose Input/Output chapter on page 55. For information regarding selecting an Event System input, refer to the Event System chapter on page 410.

Multi-Channel Timer outputs can drive a GPIO and/or be a source to the Event System. To learn more about selecting a GPIO as a Multi-Channel Timer output using GPIO alternate function registers, refer to the General-Purpose Input/Output chapter on page 55. For

Bit	Description (Continued)
[4:2] NUMBITS[2:0]	**Number of Data Bits Per Character to Transfer** This field contains the number of bits to shift for each character transfer. To learn more about valid bit positions when the character length is less than 8 bits, see the description of the ESPI 0-1 Data Registers section on page 295. 000: 8 bits. 001: 1 bit. 010: 2 bits. 011: 3 bits. 100: 4 bits. 101: 5 bits. 110: 6 bits. 111: 7 bits.
[1] SSIO	**Slave Select I/O** This bit controls the direction of the $\overline{SS}$ pin. In single Master Mode, SSIO is set to 1 even if a separate GPIO pin is being used to provide the $\overline{SS}$ output function. In the SPI slave or multi-master configuration, SSIO is set to 0. 0: $\overline{SS}$ pin configured as an input (SPI slave and multi-master modes). 1: $\overline{SS}$ pin configured as an output (SPI single-master mode).
[0] SSPO	**Slave Select Polarity** This bit controls the polarity of the $\overline{SS}$ pin. 0: $\overline{SS}$ is active Low. (SSV=1 corresponds to $\overline{SS}$=0). 1: $\overline{SS}$ is active High. (SSV=1 corresponds to $\overline{SS}$=1).

15.4.5. ESPI 0-1 Status Registers

The ESPI 0-1 Status Registers, shown in Table 148, indicate the current state of the ESPI. All bits revert to their Reset state if the ESPI is disabled.

Table 148. ESPI 0-1 Status Registers (ESPI*x*STAT)

Bit	7	6	5	4	3	2	1	0
Field	TDRE	TUND	COL	ABT	ROVR	RDRNE	TFST	SLAS
Reset	1	0	0	0	0	0	0	1
R/W	R	R/W*	R/W*	R/W*	R/W*	R	R	R
Address	ESPI0STAT @ F64h, ESPI1STAT @ F6Ch							

Note: *R/W=read access; write a 1 to clear the bit to 0; *x* references bits in the range [1:0].

Bit	Description
[7] TDRE	**Transmit Data Register Empty** 0: Transmit Data Register is full or ESPI is disabled. 1: Transmit Data Register is empty. A write to the ESPI (Transmit) Data Register clears this bit.
[6] TUND	**Transmit Underrun** 0: A Transmit Underrun error has not occurred. 1: A Transmit Underrun error has occurred.
[5] COL	**Collision** 0: A multi-master collision (mode fault) has not occurred. 1: A multi-master collision (mode fault) has occurred.
[4] ABT	**Slave Mode Transaction Abort** This bit is set if the ESPI 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 ESPIMODE Register. This bit can also be set in Slave Mode by an SCK monitor time-out (MMEN=0, BRGCTL=1). 0: A Slave Mode transaction abort has not occurred. 1: A Slave Mode transaction abort has occurred.
[3] ROVR	**Receive Overrun** 0: A Receive Overrun error has not occurred. 1: A Receive Overrun error has occurred.
[2] RDRNE	**Receive Data Register Not Empty** 0: Receive Data Register is empty. 1: Receive Data Register is not empty.

16.3.2. I²C Interrupt Status Register

The read-only I²C Interrupt Status Register, shown in Table 155, indicates the cause of any current I²C interrupt and provides the status of the I²C controller. When an interrupt occurs, one or more of the TDRE, RDRF, SAM, ARBLST, SPRS or NCKI bits is set. The GCA and RD bits do not generate an interrupt, but instead provide the status associated with the SAM bit interrupt.

Table 155. I²C Interrupt Status Register (I2CISTAT=F51h)

Bit	7	6	5	4	3	2	1	0
Field	TDRE	RDRF	SAM	GCA	RD	ARBLST	SPRS	NCKI
Reset	1	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R
Address	F51h							

Bit	Description
[7] TDRE	**Transmit Data Register Empty** When the I²C controller is enabled, this bit is 1 when the I²C Data Register is empty. When set, this bit causes the I²C controller to generate an interrupt, except when the I²C controller is shifting in data during the reception of a byte or when shifting an address and the RD bit is set. This bit clears by writing to the I2CDATA Register.
[6] RDRF	**Receive Data Register Full** This bit is set=1 when the I²C controller is enabled and the I²C controller has received a byte of data. When asserted, this bit causes the I²C controller to generate an interrupt. This bit clears by reading the I2CDATA Register.
[5] SAM	**Slave Address Match** This bit is set=1 if the I²C controller is enabled in SLAVE Mode and an address is received that matches the unique slave address or General Call Address (if enabled by the GCE bit in the I²C Mode Register). In 10-bit addressing mode, this bit is not set until a match is achieved on both address bytes. When this bit is set, the RD and GCA bits are also valid. This bit clears by reading the I2CISTAT Register.
[4] GCA	**General Call Address** This bit is set in SLAVE Mode when the General Call Address or start byte is recognized (in either 7 or 10 bit SLAVE Mode). The GCE bit in the I²C Mode Register must be set to enable recognition of the General Call Address and start byte. This bit clears when IEN=0 and is updated following the first address byte of each SLAVE Mode transaction. A General Call Address is distinguished from a start byte by the value of the RD bit (RD=0 for General Call Address, 1 for start byte).
[3] RD	**Read** This bit indicates the direction of transfer of the data. It is set when the master is reading data from the slave. This bit matches the least-significant bit of the address byte after the start condition occurs (for both MASTER and SLAVE modes). This bit clears when IEN=0, and is updated following the first address byte of each transaction.

The I^2C baud rate is calculated using the following equation.

$$I^2C\ Baud\ Rate\ (bits/s) = \frac{System\ Clock\ Frequency\ (Hz)}{4 \times BRG[15:0]}$$

When configured as a general purpose timer, the I^2C baud rate generator interrupt interval is calculated using the following equation.

$$I^2C\ Baud\ Rate\ (bits/s) = \frac{System\ Clock\ Frequency\ (Hz)}{BRG[15:0]}$$

Note: If BRG=0000h, then use 10000h in the equation.

Table 157. I^2C Baud Rate High Byte Register (I2CBRH=53h)

Bit	7	6	5	4	3	2	1	0
Field	BRH							
Reset	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Address	F53h							

Bit	Description
[7:0] BRH	**I^2C Baud Rate High Byte** The most significant byte, BRG[15:8], of the I^2C Baud Rate Generator's reload value.

Note: If the DIAG bit in the I^2C Mode Register is set to 1, a read of the I2CBRH Register returns the current value of the I^2C Baud Rate Counter[15:8].

Table 158. I^2C Baud Rate Low Byte Register (I2CBRL=F54h)

Bit	7	6	5	4	3	2	1	0
Field	BRL							
Reset	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Address	F54h							

6. Wait 1–15 ms, then clear the SIGRSUME bit.

To perform a device-initiated Resume using the software to time the USB Idle state, observe the following procedure:

1. Clear the RIRQE bit in the USBIRQCTL Register so that only a host-initiated Resume generates a USB Resume interrupt.
2. If it is not already running, configure the USB PLL for a 48 MHz output frequency, then enable the PLL and wait until it is stable. See the Clock System chapter on page 96 to learn more.
3. Ensure that the USB bus has been continuously in the Idle state for a minimum of 5 ms, then set the WAKEUP bit in the USBIRQCTL Register to initiate a Resume.
4. Set the SIGRSUME bit in the USB Control and Status Subregister (USBCS) to initiate remote wake-up signaling to the host. To raise the crossover voltage during remote wake-up signalling by initially forcing the Data J bus state, software can first write to the FORCEJ bit in the USB Control and Status Subregister (USBCS), then clear the FORCEJ bit and set the SIGRSUME bit.
5. Wait 1–15 ms, then clear the SIGRSUME bit.
6. Clear the WAKEUP bit in the USBIRQCTL Register.

17.2.8.3. Host-Initiated Resume

While in the Suspend state and not performing a device-initiated Resume, the RIRQE bit in the USBIRQCTL Register should be cleared. When the USB host wishes to wake up a USB device, it drives the Data K bus state on the USB bus for 20 ms. Upon detecting the Data K bus state, the USB Module generates a USB Resume interrupt. Software should perform the following steps to perform a host-initiated Resume:

- If it is not already running, configure the USB PLL for a 48 MHz output frequency, then enable the PLL and wait until it is stable. See the Clock System chapter on page 96 to learn more.
- *Optional*: When the USB Module recognizes PLL_{CLK}, it will clear the DEVRSUME bit in the USB Control and Status Subregister (USBCS), which can be polled. If the RIRQE bit is set, another USB Resume interrupt request is generated; therefore, the RIRQE bit should be cleared during a host-initiated Resume.

17.2.9. Stop Mode Operation

Stop Mode should only be entered when the USB Controller is either:

- Not available on the F6482 Series part number being used
- Unused, as indicated by DISCON=1 in the USB Control and Status Subregister (USBCS)

17.3.8. USB IN Endpoints Start Address Subregister

The USB IN Endpoints Start Address Subregister, shown in Table 177 defines the starting address for IN endpoints and stop address for the uppermost OUT endpoint.

Table 177. USB IN Endpoints Start Address Subregister (USBISTADDR)

Bit	7	6	5	4	3	2	1	0
Field	Reserved	INSTADDR						
Reset	0	0	0	0	0	1	0	0
R/W	R0/W*	R0/W*	R0/W*	R0/W*	R0/W*	R0/W*	R0/W*	R0/W*
Address	If USBSA = 08h in the USB Subaddress Register, accessible through the USB Subdata Register							

Note: *R0/W = Write but reads back as 0

Bit	Description
[7]	**Reserved** This bit is reserved and must be programmed to 0.
[6:0] INSTADDR	**IN Start Address** 00–7F: The starting address for IN endpoints. The first IN starting address (IN endpoint 0) is determined by INSTADDR. {INSTADDR[7:0], 0, 0} maps to endpoint buffer memory address [9:0]; therefore the minimum increment of INSTADDR is 4 bytes of buffer memory. Also, 2*(2*INSTADDR minus the uppermost OUTADDR) determines the size in bytes of the uppermost OUT endpoint buffer memory. See the USB Endpoint Buffer Memory section on page 341 for details.

Bit	Description (Continued)
[1] HSNAK	**EP0 Handshake NA** HSNAK is automatically set when a Setup token arrives. Software clears HSNAK by writing a 1 to it. 0: Do not send a NAK handshake. 1: Send a NAK handshake for every packet in the Status stage.
[0] STALL	**EP0 Stall** STALL is automatically cleared when a Setup token arrives. 0: Do not send a STALL handshake. 1: Send a STALL handshake for any IN or OUT token during the data or handshake phases of the control transfer.

17.3.19. USB IN 0–3 Byte Count Subregisters

The USB IN 0–3 Byte Count subregisters, shown in Table 188, contain the USB IN endpoint byte counts.

Table 188. USB IN 0–3 Byte Count Subregisters (USBIxBC)

Bit	7	6	5	4	3	2	1	0
Field	Reserved	BC						
Reset	0	0	0	0	0	0	0	0
R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R
Address	If USBSA = 35h, 37h, 39h, 3Bh in the USB Subaddress Register, accessible through the USB Subdata Register							

Bit	Description
[7]	**Reserved** This bit is reserved and must be programmed to 0.
[6:0] BC	**IN Byte Count** 00–40: After loading the IN endpoint x buffer memory, software should write BC with the number of bytes loaded. Writing to the USBIxBC Register arms the IN endpoint x by setting BUSY in USBIxCS. When the host sends an IN token for IN endpoint x and BUSY is set, the USB Module will transmit a BC length data packet. 41-7F: Reserved.

18.3.3. DMA Global Control Register

The DMA Global Control Register, shown in Table 206, controls the global DMA operations, including global disable, DMA priority control, burst transfer control and chain control for DMA channel pairs.

Table 206. DMA Global Control Register (DMACTL)

Bit	7	6	5	4	3	2	1	0
Field	GDISABLE	PRIORITY	BURST		Reserved	AUTOINC	CHAIN32	CHAIN10
Reset	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W
Address	FB0h							

Bit	Description
[7] GDISABLE	**DMA Global Disable** Any enabled DMA channel will resume when GDISABLE is cleared to provide an efficient mechanism to pause all DMA channels, thereby quickly allowing the CPU full bandwidth of the Register Bus. 0: DMA requests are enabled for those DMA channels that have ENABLE=1. 1: DMA requests are blocked for all DMA channels.
[6] PRIORITY	**DMA Priority Select** 0: DMA executes channel requests using round robin priority. 1: DMA executes channel requests using fixed priority.
[5:4] BURST	**Burst Transfers** 00: Block. DMA will burst transfer the entire block of data if the DMA request remains asserted. 01: Burst 4. DMA will limit burst transfer length to bursts of 4 transfers if the DMA request remains asserted. 10: Single. DMA will limit burst transfer length to a single transfer even if DMA request remains asserted. 11: Reserved.
[3]	**Reserved** This bit is reserved and must be programmed to 0.
[2] AUTOINC	**Autoincrement Enable** Clear AUTOINC before executing an instruction to manipulate bits in a DMA subregister, such as AND, BIT, and OR. 0: Autoincrement is disabled. 1: Autoincrement is enabled.

Chapter 20. Advanced Encryption Standard (AES) Accelerator

F6482 Series MCUs are equipped with an AES accelerator that implements the *Rijndael cipher encoding and decoding algorithm* in compliance with the NIST Advanced Encryption Standard. It processes 128-bit data blocks with a 128-bit key. In addition to an Electronic Codebook mode, NIST Cipher Block Chaining and Output Feedback modes are also supported. An automatic start feature facilitates use with the DMA Controller, and applications can be configured to be interrupted upon completion or error.

This AES accelerator includes the following features:

- Encrypts and decrypts using the AES Rijndael Block Cipher Algorithm
- Based on Federal Information Processing Standard (FIPS) Publication 197 from the US National Institute of Standards and Technology (NIST)
- Processes 128-bit data blocks with an 8-bit data interface
- Contains a dedicated 128-bit key buffer
- Supports the NIST OFB and CBC confidentiality modes (a *software assist* is required for decryption)
- DMA support for all modes of operation

20.1. AES Architecture

The AES accelerator implements Electronic Codebook (ECB) encryption or decryption on 128-bit data blocks. In addition to ECB encryption and decryption, the AES accelerator also provides a *hardware assist* feature for Output Feedback (OFB) and Cipher Block Chaining (CBC) modes; a software assist is required for decryption.

AES encryption is performed as defined in the FIPS-197 Specification. The Cipher Block Chaining (CBC) and Output Feedback (OFB) modes are described in the SP 800–38A Specification.

A block diagram of the AES accelerator is shown in Figure 62.

Bit	Description (Continued)
[6:0] ANAINH (cont'd.)	**SCAN=1, INMODE=00, 11** xxxxxx1: ANA8 input is selected for ADC scanning. Additional inputs may be selected. xxxxx1x: ANA9 input is selected for ADC scanning. Additional inputs may be selected. xxxx1xx: ANA10 input is selected for ADC scanning. Additional inputs may be selected. xxx1xxx: ANA11 input is selected for ADC scanning. Additional inputs may be selected. xx1xxxx: Op Amp A output is selected for ADC scanning. Additional inputs may be selected. x1xxxxx: Op Amp B output is selected for ADC scanning. Additional inputs may be selected. 1xxxxxx: Temperature Sensor is selected for ADC scanning. Additional inputs may be selected.
	SCAN=1, INMODE=01, 10 All bits are reserved.

21.3.5. ADC Input Select Low Register

The ADC Input Select Low Register, shown in Table 239, selects the ADC input(s) for conversion. If SCAN is set, ADCINSH and ADCINSL are both used to define the ADC inputs to be scanned, otherwise, only ADCINSL is used to select the ADC input(s).

Table 239. ADC Input Select Low Register (ADCINSL)

Bits	7	6	5	4	3	2	1	0
Field	ANAINL							
Reset	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Address	F74h							

Bit	Description
[7:0] ANAINL	**Analog Input Selection Low** ADC Input Selection is a function of SCAN and INMODE.
	SCAN=0, INMODE=00, 11 00000000: ANA0 input is selected for analog-to-digital conversion. 00000001: ANA1 input is selected for analog-to-digital conversion. 00000010: ANA2 input is selected for analog-to-digital conversion. 00000011: ANA3 input is selected for analog-to-digital conversion. 00000100: ANA4 input is selected for analog-to-digital conversion. 00000101: ANA5 input is selected for analog-to-digital conversion. 00000110: ANA6 input is selected for analog-to-digital conversion. 00000111: ANA7 input is selected for analog-to-digital conversion. 00001000: ANA8 input is selected for analog-to-digital conversion. 00001001: ANA9 input is selected for analog-to-digital conversion.

Table 262. Temperature vs. ADC Output, ADC V_{REF} = 1.25V (Continued)

Temperature °C	V_{TEMP}	ADC Output (Hex)
+55	1.048	D69
+60	1.065	D9F
+65	1.081	DD5
+70	1.097	E0A
+75	1.114	E40
+80	1.130	E75
+85	1.146	EAB

25.1.1. Calibration

The Temperature Sensor undergoes calibration during the manufacturing process and is maximally accurate at 30°C. Accuracy decreases as measured temperatures move further from the calibration point.

Table 263. LCD Display Memory Organization (Continued)

Subaddress in LCDSA*	Bit in LCDMEMAx, LCDMEMBx Subregisters							
	7	6	5	4	3	2	1	0
LCDMEMAx	COM3	COM2	COM1	COM0	COM3	COM2	COM1	COM0
12h	SEG05				SEG04			
11h	SEG03				SEG02			
10h	SEG01				SEG00			

Note: *LCDSA in the LCDSA Register contains the subregister address.

26.2.3. LCD Frame Timing

The LCD frame timing is a function of the LCD Controller clock selection, prescaler divide ratio, frame rate divide ratio, duty and contrast control. The LCD Controller clock selection, prescaler divide ratio, and frame rate divide ratio are configured in the LCD-CLK Register. Either PCLK or the WTO can be selected as the LCD Controller input clock source using the CLKSEL bit. The prescaler divide ratio is selected using the PRES-CALE bit, and the prescaler output clocks the frame rate divider and dynamic bias generator, as described in the Bias Generator Selection section on page 507. The frame rate divide ratio is selected using the FDIV bit, and results in the frame clock.

Duty, bias, and waveform type are configured with the LCDMODE bit in the LCD Control 2 Register (LCDCTL2). Contrast control is configured with CONTRAST in the LCD Control 1 Register (LCDCTL1), which selects the number of dead frame clock cycles per frame (Type A waveforms) or per frame pair (Type B waveforms). See the Waveform Generation section on page 508 and the Contrast Control section on page 516 to learn more.

For static, 1/2 and 1/4 duty:

Frame rate = (LCD Controller input clock frequency) ÷ ((8 + DEAD CYCLES) * PRESCALE * FDIV)

In this equation, DEAD CYCLES is selected by the value of CONTRAST.

For 1/3 duty:

Frame rate = (LCD Controller input clock frequency) ÷ ((6 + DEAD CYCLES) * PRESCALE * FDIV

In this equation, DEAD CYCLES is selected by the value of CONTRAST.

For Type A waveforms, the common signals (COMx) repeat each LCD frame, whereas for Type B waveforms, the common signals repeat after every two LCD frames.

Memory Map Example									
COM3	COM2	COM1	COM0		COM3	COM2	COM1	COM0	
x	x	0	1	SEG01	x	x	1	1	SEG00

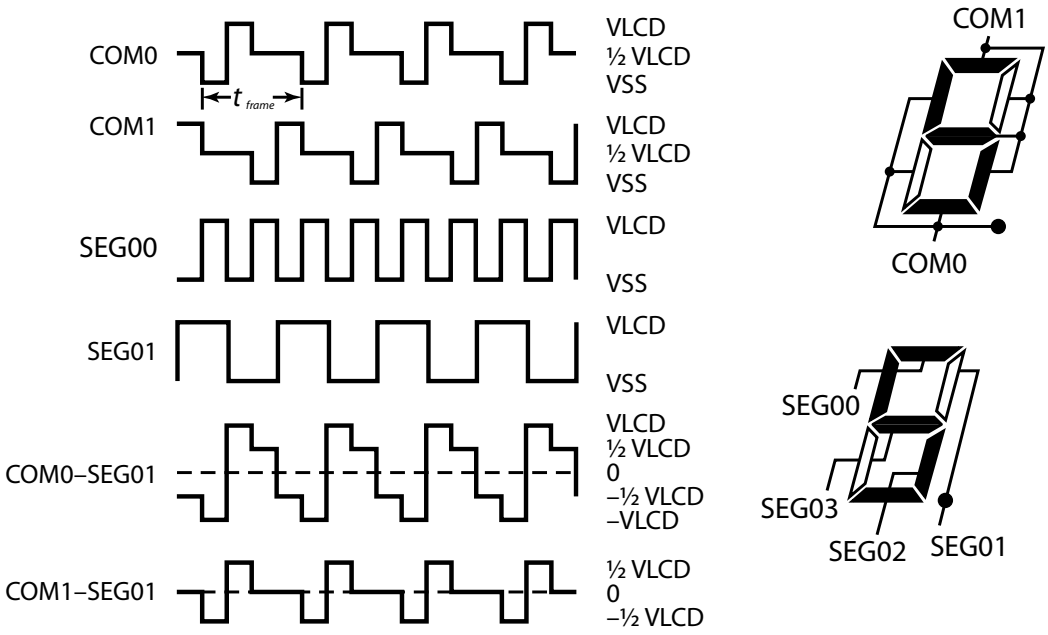

Figure 88. 1/2 Duty Mode with 1/2 Bias Type A Example Waveforms

working register R0. The bit fields of this status byte are defined in Table 303. Additionally, the user code should pop the address and data bytes off the stack.

The write routine uses 16 bytes of stack space in addition to the two bytes of address, and data pushed by the user code. Sufficient memory must be available for this stack usage.

Because of the Flash memory architecture, NVDS writes exhibit a nonuniform execution time. In general, a write takes 136µs (assuming a 20MHz system clock). For every 256 writes, however, a maintenance operation is necessary. In this rare occurrence, the write takes up to 58ms to complete. Slower system clock speeds result in proportionally higher execution times.

NVDS byte writes to illegal addresses (those exceeding the NVDS array size) have no effect. Illegal write operations have a 7µs execution time.

As is the case for writing to Flash memory, before performing a write operation on the NVDS, the FLL must be running at a minimum of 1MHz.

Word write is not available.

Table 303. Write Status Byte

Bit	7	6	5	4	3	2	1	0
Field	Reserved				IGADDR	WRERROR		
Default Value	0	0	0	X	0	0	0	0

Bit	Description
[7:4]	**Reserved** All bits are reserved and must be programmed to 0000.
[3] IGADDR	**Illegal Address** 0: The write attempted was to a legal address (one that is within the NVDS array size). 1: The write attempted was to an illegal address (one that exceeds the NVDS array size).
[2:0] WRERROR	**NVDS Write Error** 0: No NVDS write error occurred. 1–7: An NVDS write error occurred. The memory location accessed may be corrupt. Attempting the write again may succeed because the NVDS routines will attempt to select a different memory location.

29.2.2. Byte Read

To read a byte from the NVDS array, user code must first push the address onto the stack. User code issues a CALL instruction to the address of the byte-read routine (i.e., `0xF000`). At the return from the subroutine, the read byte resides in working register R0, and the read status byte resides in working register R1; the bit fields of this status byte are defined in Table 304. In addition, the user code should pop the address byte off the stack.

31.3. eZ8 CPU Instruction Notation

In the eZ8 CPU Instruction Summary section on page 585, the operands, condition codes, status flags and address modes are represented by the notational shorthand provided in Table 315.

Table 315. Notational Shorthand

Notation	Description	Operand	Range
b	Bit	b	b represents a value from 0 to 7 (000b to 111b)
cc	Condition Code	–	See the Condition Codes overview in the eZ8 CPU Core User Manual (UM0128)
DA	Direct Address	Addrs	Addrs. represents a number in the range of 0000h to FFFFh
ER	Extended Addressing Register	Reg	Reg. represents a number in the range of 000h to FFFh
IM	Immediate Data	#Data	Data is a number between 00h to FFh
Ir	Indirect Working Register	@Rn	n=0 –15
IR	Indirect Register	@Reg	Reg. represents a number in the range of 00h to FFh
Irr	Indirect Working Register Pair	@RRp	p=0, 2, 4, 6, 8, 10, 12 or 14
IRR	Indirect Register Pair	@Reg	Reg. represents an even number in the range 00h to FEh
p	Polarity	p	Polarity is a single bit binary value of either 0b or 1b.
r	Working Register	Rn	n=0–15
R	Register	Reg	Reg. represents a number in the range of 00h to FFh
RA	Relative Address	X	X represents an index in the range of +127 to –128, which is an offset relative to the address of the next instruction
rr	Working Register Pair	RRp	p=0, 2, 4, 6, 8, 10, 12 or 14
RR	Register Pair	Reg	Reg. represents an even number in the range of 00h to FEh
Vector	Vector Address	Vector	Vector represents a number in the range of 00h to FFh
X	Indexed	#Index	The register or register pair to be indexed is offset by the signed Index value (#Index) in a +127 to –128 range.