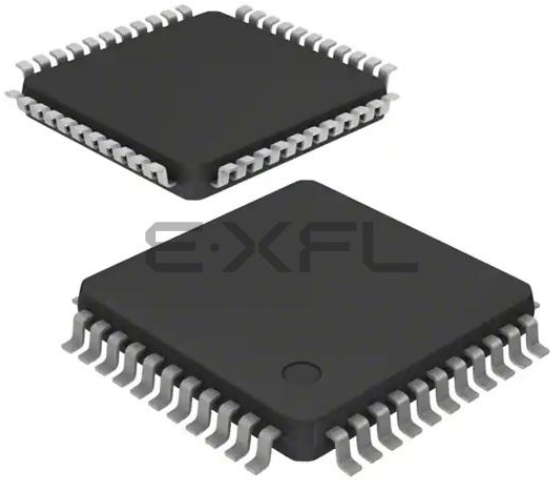

Welcome to E-XFL.COM

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	31
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 3.6V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	0°C ~ 70°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/z8f3221an020sc00tr

Table of Contents

Part Selection Guide

Table 1 identifies the basic features and package styles available for each device within the Z8 Encore! XP product line.

Table 1. Z8 Encore! XP 64K Series Flash Microcontrollers Part Selection Guide

Part Number	Flash (KB)	RAM (KB)	I/O	16-bit Timers with PWM	ADC Inputs	UARTs with IrDA	I^2C	SPI	40/44-pin packages	64/68-pin packages	80-pin package
Z8F1621	16	2	31	3	8	2	1	1	X		
Z8F1622	16	2	46	4	12	2	1	1		X	
Z8F2421	24	2	31	3	8	2	1	1	X		
Z8F2422	24	2	46	4	12	2	1	1		X	
Z8F3221	32	2	31	3	8	2	1	1	X		
Z8F3222	32	2	46	4	12	2	1	1		X	
Z8F4821	48	4	31	3	8	2	1	1	X		
Z8F4822	48	4	46	4	12	2	1	1		X	
Z8F4823	48	4	60	4	12	2	1	1			X
Z8F6421	64	4	31	3	8	2	1	1	X		
Z8F6422	64	4	46	4	12	2	1	1		X	
Z8F6423	64	4	60	4	12	2	1	1			X
Die Form Sales	Contact Zilog®										

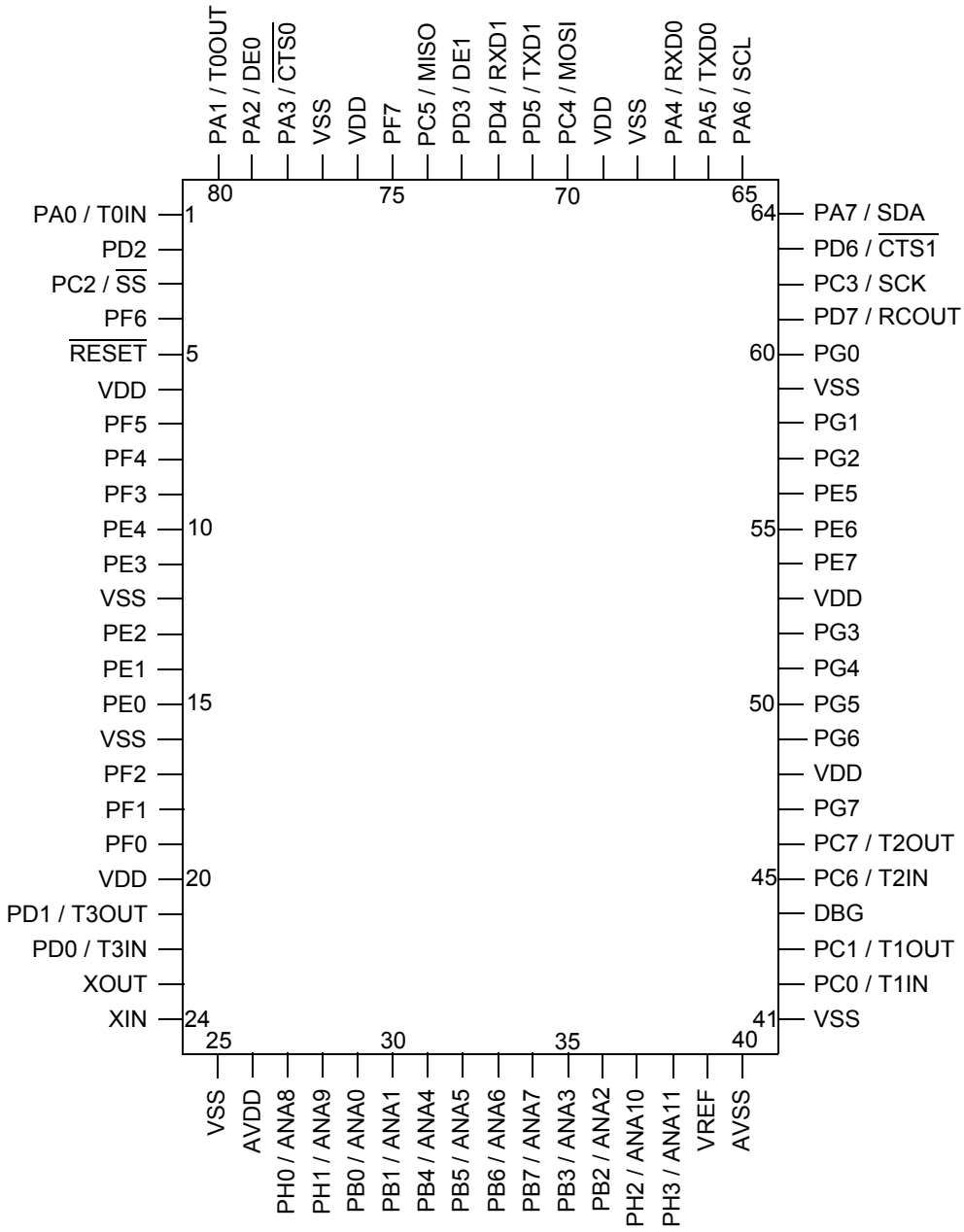

Figure 7. Z8 Encore! XP 64K Series Flash Microcontrollers in 80-Pin Quad Flat Package (QFP)

Control Register Summary

Timer 0 High Byte
T0H (F00H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Timer 0 current count value [15:8]

Timer 0 Low Byte
T0L (F01H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Timer 0 current count value [7:0]

Timer 0 Reload High Byte
T0RH (F02H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Timer 0 reload value [15:8]

Timer 0 Reload Low Byte
T0RL (HF03 - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Timer 0 reload value [7:0]

Timer 0 PWM High Byte
T0PWMH (F04H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Timer 0 PWM value [15:8]

Timer 0 Control 0
T0CTL0 (F06H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Reserved

Cascade Timer
0 = Timer 0 Input signal is GPIO pin
1 = Timer 0 Input signal is Timer 3 out

Reserved

Timer 0 Control 1
T0CTL1 (F07H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Timer Mode
000 = One-Shot mode
001 = CONTINUOUS mode
010 = COUNTER mode
011 = PWM mode
100 = CAPTURE mode
101 = COMPARE mode
110 = GATED mode
111 = Capture/COMPARE mode

Prescale Value
000 = Divide by 1
001 = Divide by 2
010 = Divide by 4
011 = Divide by 8
100 = Divide by 16
101 = Divide by 32
110 = Divide by 64
111 = Divide by 128

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

Timer Enable
0 = Timer is disabled
1 = Timer is enabled

Timer 1 High Byte
T1H (F08H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Timer 1 current count value [15:8]

Timer 1 Low Byte
T1L (F09H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Timer 1 current count value [7:0]

Timer 1 Reload High Byte
T1RH (F0AH - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Timer 1 reload value [15:8]

Timer 1 Reload Low Byte
T1RL (F0BH - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Timer 1 reload value [7:0]

DMA_ADC Control
DMAACTL (FBEH - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

- 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
- D4: Reserved
- D3: 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: 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

Port F Control
PFCTL (FE5H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Port F Control[7:0]
Provides Access to Port Sub-Registers

Port F Input Data
PFIN (FE6H - Read Only)

D7	D6	D5	D4	D3	D2	D1	D0

Port F Input Data [7:0]

Port F Output Data
PFOUT (FE7H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Port F Output Data [7:0]

Port G Address
PGADDR (FE8H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Port G Address[7:0]
Selects Port Sub-Registers:
00H = No function
01H = Data direction
02H = Alternate function
03H = Output control (open-drain)
04H = High drive enable
05H = Stop Mode Recovery enable
06H-FFH = No function

Port G Control
PGCTL (FE9H - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Port G Control[7:0]
Provides Access to Port Sub-Registers

Port G Input Data
PGIN (FEAH - Read Only)

D7	D6	D5	D4	D3	D2	D1	D0

Port G Input Data [7:0]

Port G Output Data
PGOUT (FEBH - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Port G Output Data [7:0]

Port H Address
PHADDR (FECH - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Port H Address[7:0]
Selects Port Sub-Registers:
00H = No function
01H = Data direction
02H = Alternate function
03H = Output control (open-drain)
04H = High drive enable
05H = Stop Mode Recovery enable
06H-FFH = No function

Port H Control
PHCTL (FEDH - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Port H Control [3:0]
Provides Access to Port Sub-Registers

Reserved

Port H Input Data
PHIN (FEEH - Read Only)

D7	D6	D5	D4	D3	D2	D1	D0

Port H Input Data [3:0]

Reserved

Port H Output Data
PHOUT (FEFH - Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0

Port H Output Data [3:0]

Reserved

AF[7:0]—Port Alternate Function enabled
0 = The port pin is in NORMAL mode and the DDx bit in the Port A–H Data Direction sub-register determines the direction of the pin.
1 = The alternate function is selected. Port pin operation is controlled by the alternate function.

Port A–H Output Control Sub-Registers

The Port A–H Output Control sub-register (Table 18) is accessed through the Port A–H Control register by writing 03H to the Port A–H Address register. Setting the bits in the Port A–H Output Control sub-registers to 1 configures the specified port pins for open-drain operation. These sub-registers affect the pins directly and, as a result, alternate functions are also affected.

Table 18. Port A–H Output Control Sub-Registers

BITS	7	6	5	4	3	2	1	0
FIELD	POC7	POC6	POC5	POC4	POC3	POC2	POC1	POC0
RESET	0							
R/W	R/W							
ADDR	If 03H in Port A–H Address Register, accessible through Port A–H Control Register							

POC[7:0]—Port Output Control
These bits function independently of the alternate function bit and disables the drains if set to 1.
0 = The drains are enabled for any output mode.
1 = The drain of the associated pin is disabled (open-drain mode).

Port A–H High Drive Enable Sub-Registers

The Port A–H High Drive Enable sub-register (Table 19) is accessed through the Port A–H Control register by writing 04H to the Port A–H Address register. Setting the bits in the Port A–H High Drive Enable sub-registers to 1 configures the specified port pins for high current output drive operation. The Port A–H High Drive Enable sub-register affects the pins directly and, as a result, alternate functions are also affected.

PAD*x*S—PA*x*/PD*x* Selection
0 = PA*x* is used for the interrupt for PA*x*/PD*x* interrupt request.
1 = PD*x* is used for the interrupt for PA*x*/PD*x* interrupt request.
where *x* indicates the specific GPIO Port pin number (0 through 7).

Interrupt Control Register

The Interrupt Control (IRQCTL) register (Table 38) contains the master enable bit for all interrupts.

Table 38. Interrupt Control Register (IRQCTL)

BITS	7	6	5	4	3	2	1	0
FIELD	IRQE	Reserved						
RESET	0							
R/W	R/W	R						
ADDR	FCFH							

IRQE—Interrupt Request Enable
This bit is set to 1 by execution of an EI or IRET instruction, or by a direct register write of a 1 to this bit. It is reset to 0 by executing a DI instruction, eZ8 CPU acknowledgement of an interrupt request, or Reset.
0 = Interrupts are disabled
1 = Interrupts are enabled

Reserved—Must be 0.

One-Shot time-out, first set the TPOL bit in the Timer Control 1 Register to the start value before beginning ONE-SHOT mode. Then, after starting the timer, set TPOL to the opposite bit value.

Follow the steps below for configuring a timer for ONE-SHOT mode and initiating the count:

1. Write to the Timer Control 1 register to:
   – Disable the timer
   – Configure the timer for ONE-SHOT mode
   – Set the prescale value
   – If using the Timer Output alternate function, set the initial output level (High or Low)
2. Write to the Timer High and Low Byte registers to set the starting count value
3. Write to the Timer Reload High and Low Byte registers to set the Reload value
4. If desired, enable the timer interrupt and set the timer interrupt priority by writing to the relevant interrupt registers
5. If using the Timer Output function, configure the associated GPIO port pin for the Timer Output alternate function
6. Write to the Timer Control 1 register to enable the timer and initiate counting

In ONE-SHOT mode, the system clock always provides the timer input. The timer period is given by the following equation:

$$\text{ONE-SHOT Mode Time-Out Period (s)} = \frac{(\text{Reload Value} - \text{Start Value}) \times \text{Prescale}}{\text{System Clock Frequency (Hz)}}$$

CONTINUOUS Mode

In CONTINUOUS mode, the timer counts up to the 16-bit Reload value stored in the Timer Reload High and Low Byte registers. The timer input is the system clock. Upon reaching the Reload value, the timer generates an interrupt, the count value in the Timer High and Low Byte registers is reset to 0001H and counting resumes. Also, if the Timer Output alternate function is enabled, the Timer Output pin changes state (from Low to High or from High to Low) upon timer Reload.

Follow the steps below for configuring a timer for CONTINUOUS mode and initiating the count:

1. Write to the Timer Control 1 register to:
   – Disable the timer
   – Configure the timer for CONTINUOUS mode
   – Set the prescale value
   – If using the Timer Output alternate function, set the initial output level (High or Low)

Table 42. Timer 0-3 Reload Low Byte Register (TxRL)

BITS	7	6	5	4	3	2	1	0
FIELD	TRL							
RESET	1							
R/W	R/W							
ADDR	F03H, F0BH, F13H, F1BH							

TRH and TRL—Timer Reload Register High and Low
These two bytes form the 16-bit Reload value, {TRH[7:0], TRL[7:0]}. This value sets the maximum count value which initiates a timer reload to 0001H. In COMPARE mode, these two byte form the 16-bit Compare value.

Timer 0-3 PWM High and Low Byte Registers

The Timer 0-3 PWM High and Low Byte (TxPWMH and TxPWML) registers (see Table 43 and Table 44 on page 92) are used for Pulse-Width Modulator (PWM) operations. These registers also store the Capture values for the Capture and Capture/COMPARE modes.

Table 43. Timer 0-3 PWM High Byte Register (TxPWMH)

BITS	7	6	5	4	3	2	1	0
FIELD	PWMH							
RESET	0							
R/W	R/W							
ADDR	F04H, F0CH, F14H, F1CH							

Table 44. Timer 0-3 PWM Low Byte Register (TxPWML)

BITS	7	6	5	4	3	2	1	0
FIELD	PWML							
RESET	0							
R/W	R/W							
ADDR	F05H, F0DH, F15H, F1DH							

01 = The UART generates an interrupt request only on received address bytes.
10 = The UART generates an interrupt request when a received address byte matches the value stored in the Address Compare Register and on all successive data bytes until an address mismatch occurs.
11 = The UART generates an interrupt request on all received data bytes for which the most recent address byte matched the value in the Address Compare Register.

MPEN—MULTIPROCESSOR (9-bit) Enable
This bit is used to enable MULTIPROCESSOR (9-bit) mode.
0 = Disable MULTIPROCESSOR (9-bit) mode.
1 = Enable MULTIPROCESSOR (9-bit) mode.

MPBT—MULTIPROCESSOR Bit Transmit
This bit is applicable only when MULTIPROCESSOR (9-bit) mode is enabled.
0 = Send a 0 in the multiprocessor bit location of the data stream (9^{th} bit).
1 = Send a 1 in the multiprocessor bit location of the data stream (9^{th} bit).

DEPOL—Driver Enable Polarity
0 = DE signal is Active High.
1 = DE signal is Active Low.

BRGCTL—Baud Rate Control
This bit causes different UART behavior depending on whether the UART receiver is enabled (REN = 1 in the UART Control 0 Register).

When the UART receiver is not enabled, this bit determines whether the Baud Rate Generator issues interrupts.
0 = Reads from the Baud Rate High and Low Byte registers return the BRG Reload Value
1 = The Baud Rate Generator generates a receive interrupt when it counts down to 0. Reads from the Baud Rate High and Low Byte registers return the current BRG count value.

When the UART receiver is enabled, this bit allows reads from the Baud Rate Registers to return the BRG count value instead of the Reload Value.
0 = Reads from the Baud Rate High and Low Byte registers return the BRG Reload Value.
1 = Reads from the Baud Rate High and Low Byte registers return the current BRG count value. Unlike the Timers, there is no mechanism to latch the High Byte when the Low Byte is read.

$\overline{\text{RDAIRQ}}$—Receive Data Interrupt $\overline{\text{Enable}}$
0 = Received data and receiver errors generates an interrupt request to the Interrupt Controller.
1 = Received data does not generate an interrupt request to the Interrupt Controller. Only receiver errors generate an interrupt request.

IREN—Infrared Encoder/Decoder Enable
0 = Infrared Encoder/Decoder is disabled. UART operates normally operation.

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.

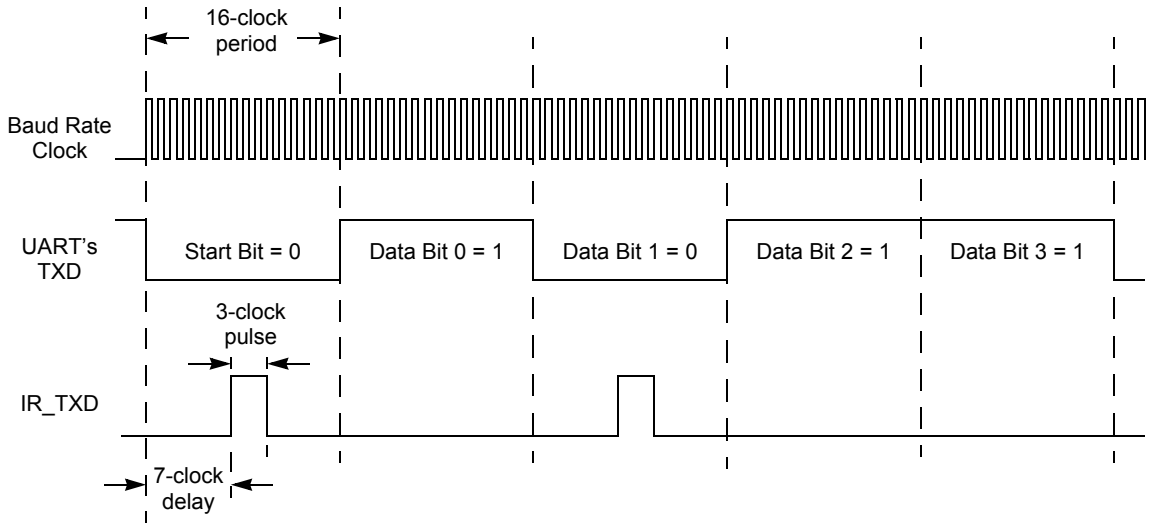

Figure 20. Infrared Data Transmission

repeated. If the incoming data is a logical 1 (no pulse), the Endec returns to the initial state and waits for the next falling edge. As each falling edge is detected, the Endec clock counter is reset, resynchronizing the Endec to the incoming signal. This action allows the Endec to tolerate jitter and baud rate errors in the incoming data stream. Resynchronizing the Endec does not alter the operation of the UART, which ultimately receives the data. The UART is only synchronized to the incoming data stream when a Start bit is received.

Infrared Encoder/Decoder Control Register Definitions

All Infrared Endec configuration and status information is set by the UART control registers as defined in UART Control Register Definitions on page 114.

Caution: *To prevent spurious signals during IrDA data transmission, set the* `IREN` *bit in the UARTx Control 1 register to 1 to enable the Infrared Encoder/Decoder before enabling the GPIO Port alternate function for the corresponding pin.*

register. The IRQE, PHASE, CLKPOL, WOR bits in the SPICTL register and the NUMBITS field in the SPIMODE register must be set to be consistent with the other SPI devices. The STR bit in the SPICTL register may be used if desired to force a "startup" interrupt. The BIRQ bit in the SPICTL register and the SSV bit in the SPIMODE register are not used in SLAVE mode. The SPI baud rate generator is not used in SLAVE mode so the SPIBRH and SPIBRL registers need not be initialized.

If the slave has data to send to the master, the data must be written to the SPIDAT register before the transaction starts (first edge of SCK when $\overline{SS}$ is asserted). If the SPIDAT register is not written prior to the slave transaction, the MISO pin outputs whatever value is currently in the SPIDAT register.

Due to the delay resulting from synchronization of the SPI input signals to the internal system clock, the maximum SPICLK baud rate that can be supported in SLAVE mode is the system clock frequency (XIN) divided by 8. This rate is controlled by the SPI master.

Error Detection

The SPI contains error detection logic to support SPI communication protocols and recognize when communication errors have occurred. The SPI Status register indicates when a data transmission error has been detected.

Overrun (Write Collision)

An overrun error (write collision) indicates a write to the SPI Data register was attempted while a data transfer is in progress (in either MASTER or SLAVE modes). An overrun sets the OVR bit in the SPI Status register to 1. Writing a 1 to OVR clears this error Flag. The data register is not altered when a write occurs while data transfer is in progress.

Mode Fault (Multi-Master Collision)

A mode fault indicates when more than one Master is trying to communicate at the same time (a multi-master collision). The mode fault is detected when the enabled Master's $\overline{SS}$ pin is asserted. A mode fault sets the COL bit in the SPI Status register to 1. Writing a 1 to COL clears this error Flag.

Slave Mode Abort

In SLAVE mode of operation if the $\overline{SS}$ pin deasserts before all bits in a character have been transferred, the transaction is aborted. When this condition occurs the ABT bit is set in the SPISTAT register as well as the IRQ bit (indicating the transaction is complete). The next time $\overline{SS}$ asserts, the MISO pin outputs SPIDAT[7], regardless of where the previous transaction left off. Writing a 1 to ABT clears this error Flag.

SPI Interrupts

When SPI interrupts are enabled, the SPI generates an interrupt after character transmission/reception completes in both MASTER and SLAVE modes. A character can be

15. The I2C Controller sends the repeated START condition.
16. The I2C Controller loads the I2C Shift register with the contents of the I2C Data register (third address transfer).
17. The I2C Controller sends 11110B followed by the two most significant bits of the slave read address and a 1 (read).
18. The I2C slave sends an acknowledge by pulling the SDA signal Low during the next high period of SCL

    If the slave were to Not Acknowledge at this point (this should not happen because the slave did acknowledge the first two address bytes), software would respond by setting the STOP and FLUSH bits and clearing the TXI bit. The I2C Controller sends the STOP condition on the bus and clears the STOP and NCKI bits. The transaction is complete (ignore the following steps).
19. The I2C Controller shifts in a byte of data from the I2C slave on the SDA signal. The I2C Controller sends a Not Acknowledge to the I2C slave if the NAK bit is set (last byte), else it sends an Acknowledge.
20. The I2C Controller asserts the Receive interrupt (RDRF bit set in the Status register).
21. Software responds by reading the I2C Data register which clears the RDRF bit. If there is only one more byte to receive, set the NAK bit of the I2C Control register.
22. If there are one or more bytes to transfer, return to step 19.
23. After the last byte is shifted in, a Not Acknowledge interrupt is generated by the I2C Controller.
24. Software responds by setting the STOP bit of the I2C Control register.
25. A STOP condition is sent to the I2C slave and the STOP and NCKI bits are cleared.

I2C Control Register Definitions

I2C Data Register

The I2C Data register (see Table 70 on page 157) holds the data that is to be loaded into the I2C Shift register during a write to a slave. This register also holds data that is loaded from the I2C Shift register during a read from a slave. The I2C Shift Register is not accessible in the Register File address space, but is used only to buffer incoming and outgoing data.

Table 106. DC Characteristics (Continued)

Symbol	Parameter	T_A = –40 °C to 125 °C			Units	Conditions
		Minimum	Typical	Maximum		
I_{DDS}	Stop Mode Supply Current (See Figure 47 and Figure 48) GPIO pins configured as outputs	–	520	700 650	µA	V_{DD} = 3.6 V, VBO and WDT Enabled V_{DD} = 3.3 V
		–	10	25 20	µA	V_{DD} = 3.6 V, T_A = 0 to 70 ºC VBO Disabled WDT Enabled V_{DD} = 3.3 V
		–		80 70	µA	V_{DD} = 3.6 V, T_A = –40 to +105 ºC VBO Disabled WDT Enabled V_{DD} = 3.3 V
		–		250 150	µA	V_{DD} = 3.6 V, T_A = –40 to +125 ºC VBO Disabled WDT Enabled V_{DD} = 3.3 V

[1]This condition excludes all pins that have on-chip pull-ups, when driven Low.

[2]These values are provided for design guidance only and are not tested in production.

General-Purpose I/O Port Output Timing

Figure 51 and Table 115 provide timing information for GPIO Port pins.

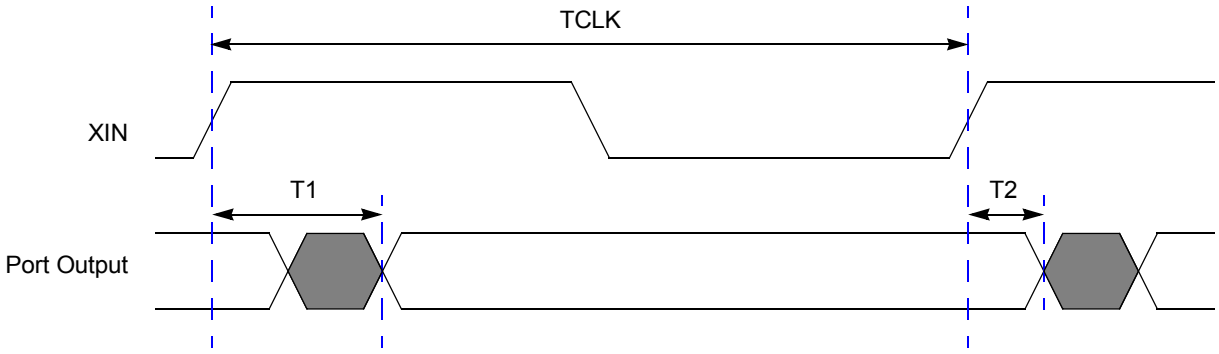

Figure 51. GPIO Port Output Timing

Table 115. GPIO Port Output Timing

		Delay (ns)	
Parameter	**Abbreviation**	**Minimum**	**Maximum**
GPIO Port pins			
T_1	XIN Rise to Port Output Valid Delay	–	20
T_2	XIN Rise to Port Output Hold Time	2	–

I²C Timing

Figure 55 and Table 119 provide timing information for I²C pins.

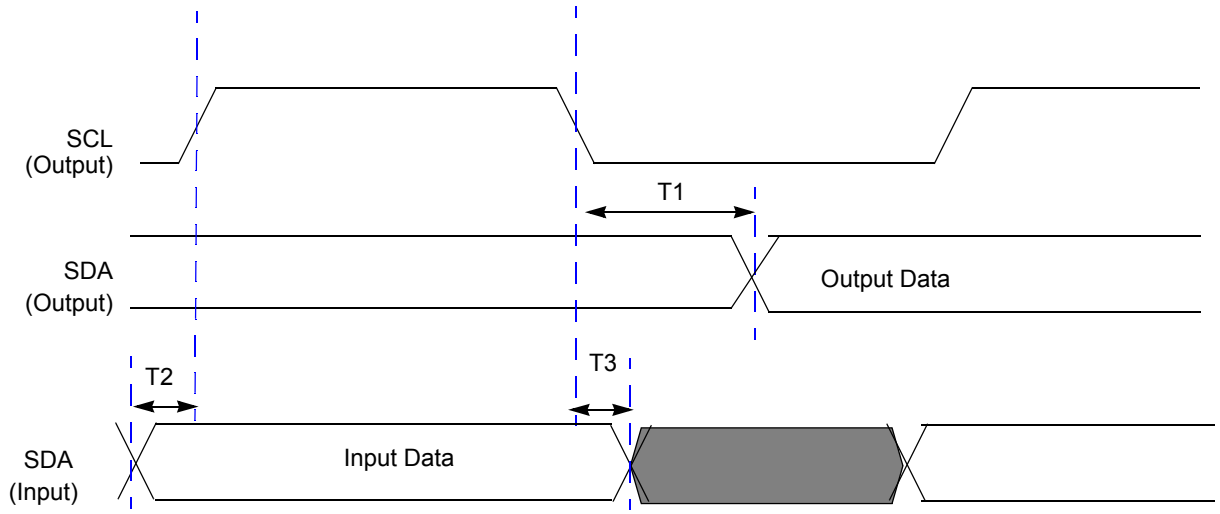

Figure 55. I²C Timing

Table 119. I²C Timing

		Delay (ns)	
Parameter	**Abbreviation**	**Minimum**	**Maximum**
I²C			
T_1	SCL Fall to SDA output delay		SCL period/4
T_2	SDA Input to SCL rising edge Setup Time	0	
T_3	SDA Input to SCL falling edge Hold Time	0	

Table 124. Condition Codes (Continued)

Binary	Hex	Assembly Mnemonic	Definition	Flag Test Operation
0011	3	ULE	Unsigned Less Than or Equal	(C OR Z) = 1
0100	4	OV	Overflow	V = 1
0101	5	MI	Minus	S = 1
0110	6	Z	Zero	Z = 1
0110	6	EQ	Equal	Z = 1
0111	7	C	Carry	C = 1
0111	7	ULT	Unsigned Less Than	C = 1
1000	8	T (or blank)	Always True	–
1001	9	GE	Greater Than or Equal	(S XOR V) = 0
1010	A	GT	Greater Than	(Z OR (S XOR V)) = 0
1011	B	UGT	Unsigned Greater Than	(C = 0 AND Z = 0) = 1
1100	C	NOV	No Overflow	V = 0
1101	D	PL	Plus	S = 0
1110	E	NZ	Non-Zero	Z = 0
1110	E	NE	Not Equal	Z = 0
1111	F	NC	No Carry	C = 0
1111	F	UGE	Unsigned Greater Than or Equal	C = 0

eZ8 CPU Instruction Classes

eZ8 CPU instructions can be divided functionally into the following groups:

- Arithmetic
- Bit Manipulation
- Block Transfer
- CPU Control
- Load
- Logical
- Program Control
- Rotate and Shift

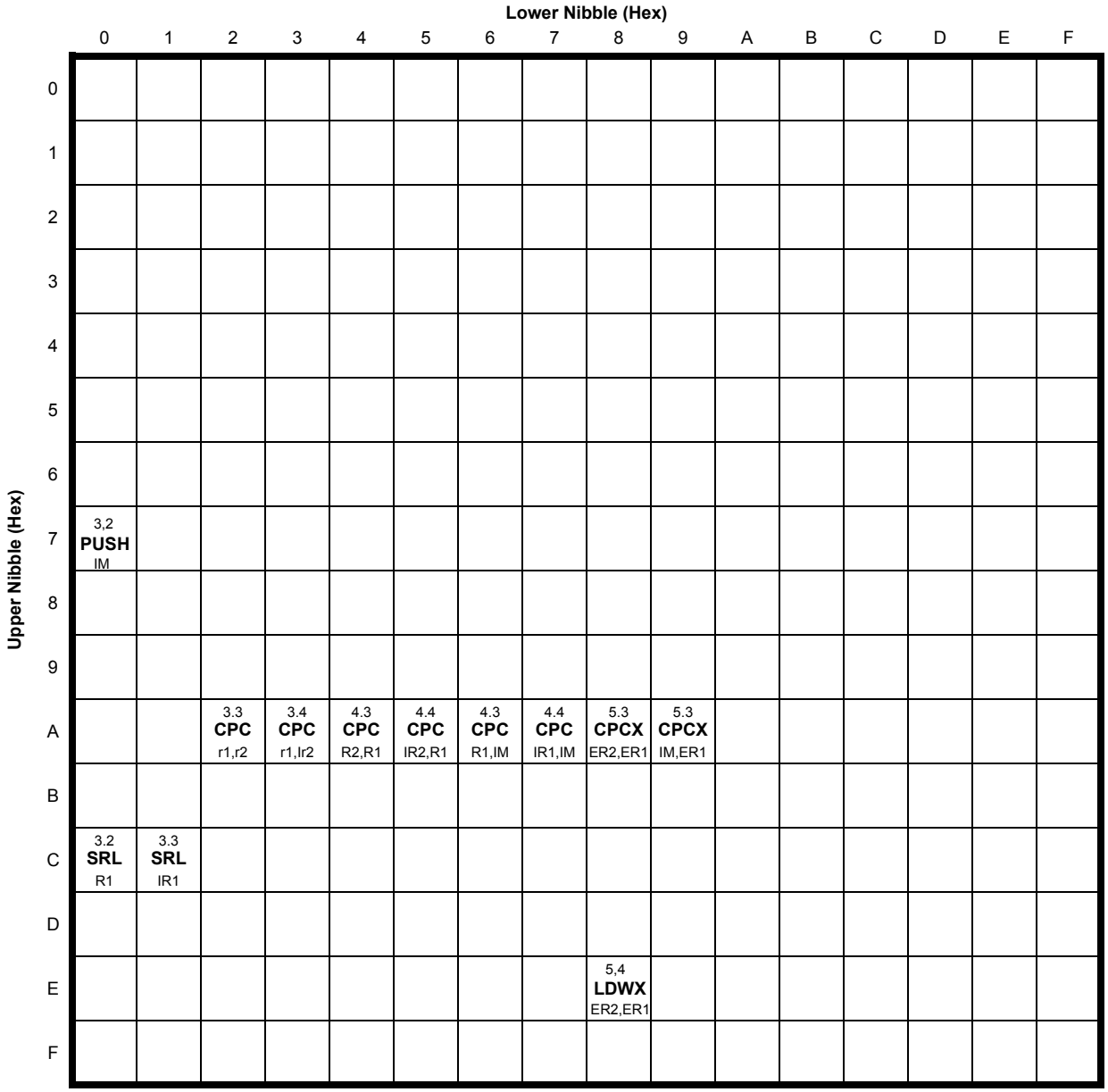

Figure 61. Second Opcode Map after 1FH