E·XFL

Zilog - EZ80190AZ050SC00TR Datasheet



Welcome to E-XFL.COM

Understanding Embedded - Microprocessors

Embedded microprocessors are specialized computing chips designed to perform specific tasks within an embedded system. Unlike general-purpose microprocessors found in personal computers, embedded microprocessors are tailored for dedicated functions within larger systems, offering optimized performance, efficiency, and reliability. These microprocessors are integral to the operation of countless electronic devices, providing the computational power necessary for controlling processes, handling data, and managing communications.

Applications of **Embedded - Microprocessors**

Embedded microprocessors are utilized across a broad spectrum of applications, making them indispensable in

Details

Product Status	Obsolete
Core Processor	eZ80
Number of Cores/Bus Width	1 Core, 8-Bit
Speed	50MHz
Co-Processors/DSP	-
RAM Controllers	-
Graphics Acceleration	No
Display & Interface Controllers	-
Ethernet	-
SATA	-
USB	-
Voltage - I/O	3.3V
Operating Temperature	0°C ~ 70°C (TA)
Security Features	-
Package / Case	100-LQFP
Supplier Device Package	100-VQFP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/zilog/ez80190az050sc00tr

Email: info@E-XFL.COM

Address: Room A, 16/F, Full Win Commercial Centre, 573 Nathan Road, Mongkok, Hong Kong

zilog ,

GPIO Interrupts	46
Level-Triggered Interrupts	46
Edge-Triggered Interrupts	47
GPIO Control Registers	47
Port x Data Registers	47
Port x Data Direction Registers	48
Port x Alternate Registers 1	48
Port x Alternate Registers 2	48
Chip Selects and Wait States	50
Memory and I/O Chip Selects	50
Memory Chip Select Operation	50
Memory Chip Select Priority	51
Reset States	51
Memory Chip Select Example	51
I/O Chip Select Operation	53
I/O Chip Select Precaution	54
Wait States	54
Chip Select Registers	55
Chip Select x Lower Bound Register	55
Chip Select x Upper Bound Register	56
Chip Select x Control Register	57
Random Access Memory	59
RAM Control Registers	60
RAM Control Register	60
RAM Address Upper Byte Register	60
Universal Zilog Interface	62
Baud Rate Generator	63
Baud Rate Generator Functional Description	63
Recommended Usage of the Baud Rate Generator	63
UZI and BRG Control Registers	64
UZI Control Registers	64
BRG Divisor Latch Registers—Low Byte	64
BRG Divisor Latch Registers—High Byte	65
Universal Asynchronous Receiver/Transmitter	67
UART Functional Description	68
UART Functions	68
	68
	69
UART Modem Control	69

zilog ₆

Pin No.	Symbol	Function	Signal Direction	Description
10	ADDR0	Address Bus	Input/Output	The ADDR0 is configured as an output in normal operation. The address bus selects a location in memory or I/O space to be read or written. This pin is configured as an input during bus acknowledge cycles. Drives the Chip Select/Wait State Generator block to generate Chip Selects.
11	ADDR1	Address Bus	Input/Output	The ADDR1 pin is configured as an output in normal operation. The address bus selects a location in memory or I/O space to be read or written. This pin is configured as an input during bus acknowledge cycles. Drives the Chip Select/ Wait State Generator block to generate Chip Selects.
12	ADDR2	Address Bus	Input/Output	The ADDR2 pin is configured as an output in normal operation. The address bus selects a location in memory or I/O space to be read or written. This pin is configured as an input during bus acknowledge cycles. Drives the Chip Select/ Wait State Generator block to generate Chip Selects.
13	ADDR3	Address Bus	Input/Output	The ADDR3 pin is configured as an output in normal operation. The address bus selects a location in memory or I/O space to be read or written. This pin is configured as an input during bus acknowledge cycles. Drives the Chip Select/ Wait State Generator block to generate Chip Selects.
14	ADDR4	Address Bus	Input/Output	The ADDR4 pin is configured as an output in normal operation. The address bus selects a location in memory or I/O space to be read or written. This pin is configured as an input during bus acknowledge cycles. Drives the Chip Select/ Wait State Generator block to generate Chip Selects.
15	ADDR5	Address Bus	Input/Output	The ADDR5 pin is configured as an output in normal operation. The address bus selects a location in memory or I/O space to be read or written. This pin is configured as an input during bus acknowledge cycles. Drives the Chip Select/ Wait State Generator block to generate Chip Selects.

Table 1. 100-Pin LQFP Pin Identification of the eZ80190 Device (Continued)



Bit Position	Value	Description
[7:0] WDT_RR	A5h	The first write value required to reset the WDT prior to a time- out.
	5Ah	The second write value required to reset the WDT prior to a time-out. If an A5h, 5Ah sequence is written to WDT_RR, the WDT timer is reset to its initial count value, and counting resumes.

42

zilog

Table 13. Port x Data Registers (PA_DR = 96h, PB_DR = 9Ah, PC_DR = 9Eh, PD_DR = A2h)

Bit	7	6	5	4	3	2	1	0
Reset	Х	Х	Х	Х	Х	Х	Х	Х
CPU Access	R/W							
Note: X = Undefined; R/W = Read/Write.								

Port x Data Direction Registers

In conjunction with the other GPIO Control Registers, the Port x Data Direction registers, listed in Table 14, control the operating modes of the GPIO port pins. For more information see Table 12 on page 43.

Table 14. Port x Data Direction Registers	(PA_DDR = 97h	, PB_DDR	= 9Bh, PC_	_DDR = 9Fh,
PD_	_DDR = A3h)			

Bit	7	6	5	4	3	2	1	0
Reset	1	1	1	1	1	1	1	1
CPU Access	R/W							
Note: R/W = Read/Write.								

Port x Alternate Registers 1

In conjunction with the other GPIO Control Registers, the Port x Alternate Register 1, listed in Table 15, control the operating modes of the GPIO port pins. For more information see Table 12 on page 43.

Table 15. Port x Alternate Registers 1(PA_ALT1 = 98h, PB_ALT1 = 9Ch, PC_ALT1 = A0h,
PD_ALT1 = A4h)

Bit	7	6	5	4	3	2	1	0
Reset	0	0	0	0	0	0	0	0
CPU Access	R/W							
Note: R/W = Read/Write.								

Port x Alternate Registers 2

In conjunction with the other GPIO Control Registers, the Port x Alternate Register 2, listed in Table 16 on page 49, control the operating modes of the GPIO port pins. For more information see Table 12 on page 43.

zilog ₅₂





Chip Select	CSx_CTL[3] CSx_EN	CSx_CTL[4] CSx_IO	CSx_LBR	CSx_UBR	Description						
CS0	1	0	00h	7Fh	CS0 is enabled as a Memory Chip Select. Valid addresses range from 000000h to 7FFFFFh.						
CS1	1	0	80h	9Fh	CS1 is enabled as a Memory Chip Select. Valid addresses range from 800000h to 9FFFFFh.						

Table 17. Register Values for Memory Chip Select Example



Bit Position	Value	Description
3	0	Chip Select is disabled.
CS_EN	1	Chip Select is enabled.
[2:0]	000	Reserved—must be 000.

58

zilog',

Data Transfers

Transmit—To transmit data, the application enables the transmit interrupt. An interrupt is immediately expected in response to this interrupt. The application reads the UARTx_IIR register and determines that the interrupt occurs because of an empty UARTx_THR register. When the application determines this occurrence, the application writes the transmit data bytes to the UARTx_THR register. The number of bytes that the application writes depends on whether or not the FIFO is enabled. If the FIFO is enabled, the application can write 16 bytes at one time. If the FIFO is not enabled, the application can Write Only one byte at a time. As a result of the first write, the interrupt is deactivated. The processor then waits for the next interrupt. When the interrupt is raised by the UART module, the processor repeats the same process until it exhausts all of the data for transmission.

To control and check the modem status, the application sets up the modem by writing to UARTx_MCTL register and reading from the UARTx_MSR register before starting the above process.

Receive—The receiver is always enabled and continually checks for the start bit on the RXD input signal. When an interrupt is raised by the UART module, the application reads the UARTx_IIR register and determines the cause of the interrupt. If the cause is a line status interrupt, the application reads the UARTx_LSR register, reads the data byte, then discards the byte or takes other action. If the interrupt is caused by a RECEIVE DATA READY condition, the application alternately reads the UARTx_LSR and UARTx_RBR registers and removes all received data bytes. It reads the UARTx_LSR register before reading the UARTx_RBR register to determine if there is a NO ERROR condition in the received data.

To control and check modem status, the application sets up the modem by writing to the UARTx_MCTL register and reading the UARTx_MSR register before starting the above process.

Poll Mode Transfers

When interrupts are disabled, all data transfers are referred to as poll mode transfers. In poll mode transfers, the application must continually poll the UARTx_LSR register to transmit or receive data without enabling the interrupts. This condition is also true for the UARTx_MSR register. If the interrupts are not enabled, the data in the UARTx_IIR register cannot be used to determine the cause of an interrupt.

UART Registers

After a system reset, all UART registers are set to their default values. Any writes to unused registers or register bits are ignored. READs return a value of 0. For compatibility with future versions of this part, unused bits within a register should always be written

zilog | 86

master and slave must operate with the same timing. The master device always places data on the MOSI line a half-cycle before the clock edge (SCK signal), for the slave device to latch the data.



Number of Cycles on the SCK Signal

Figure	14	SPI	Timina
Iguic		U I I	rinnig

	СРНА	CPOL	SCK Transmit Edge	SCK Receive Edge	SCK Idle State	SS High Between Characters?
0		0	Falling	Rising	Low	Yes
0		1	Rising	Falling	High	Yes
1		0	Rising	Falling	Low	No
1		1	Falling	Rising	High	No

Table 38. SPI Clock Phase and Clock Polarity Operation

SPI Functional Description

Figure 15 on page 87 displays a block diagram of the serial peripheral interface circuitry. When a master device transmits to a slave device via the MOSI signal, the slave device



Arbitration

A master may start a transfer only if the bus is free. Two or more masters may generate a START condition within the minimum hold time of the START condition. The result is a defined START condition to the bus. Arbitration occurs on the SDA line while the SCL line is at the High level in such a way that the master, (which transmits a High level while another master is transmitting a Low level), switches off its data output stage. The master switches off its data output stage because the level on the bus does not correspond to its own level.

Arbitration can continue for many bits. Its first stage is a comparison of the address bits. If the masters are each trying to address the same device, arbitration continues with a comparison of the data. Because address and data information on the I^2C bus is used for arbitration, no information is lost during this process. A master which loses the arbitration can generate clock pulses until the end of the byte in which it loses the arbitration.

If a master also incorporates a slave function and it loses arbitration during the addressing stage, it is possible that the winning master is trying to address it. The losing master must switch over immediately to SLAVE RECEIVE mode. Figure 20 displays the arbitration procedure for two masters. Of course, more may be involved (depending on how many masters are connected to the bus). The moment there is a difference between the internal data level of the master generating DATA 1 and the actual level on the SDA line, its data output is switched off, which means that a High output level is then connected to the bus. As a result, the data transfer initiated by the winning master is not affected. Because control of the I²C bus is decided solely on the address and data sent by competing masters, there is no central master, nor any order of priority on the bus.

Special attention must be paid if, during a serial transfer, the arbitration procedure is still in progress at the moment when a repeated START condition or a STOP condition is transmitted to the I^2C bus. If it is possible for such a situation to occur, the masters involved must send this repeated START condition or STOP condition at the same position in the format frame.

eZ80190

Product Specification







Figure 21. Multiply-Accumulator Block Diagram

Multiply-Accumulator Basic Operation

Figure 22 on page 115 displays a simplified view of the state progression of the MACC when performing calculation on a set of data. The progression begins in the upper left corner with a DATA bank containing the value EMPTY. The CPU loads the MACC control registers to define the next MACC calculation.

If the MACC is not busy with an existing calculation (EMPTY or DONE), the DATA and CALC banks are immediately swapped to initiate the new calculation. If the MACC is busy with an existing calculation, the DATA bank status changes to READY and waits for the MACC to complete the existing calculation. Then, the DATA and CALC banks are swapped to initiate the new calculation.

Assuming the DATA bank is EMPTY or READY when the MACC completes the new calculation, the CALC bank is swapped with the DATA bank. The CPU can then retrieve the result of the new calculation from the accumulator.

zilog[®]|₁₁₇

Alternatively, any write operation to any of the MACC registers besides MACC_STAT also changes the DATA bank status from DONE to EMPTY. This state change occurs because write operations generally indicate a requirement to define a new calculation.

Alternatives to OTI2R and INI2R

The INI2R and OTI2R instructions are recommended for CPU input and output access to the MACC control registers. However, it is not required that only these instructions be employed. Other I/O instructions can also be used.



Caution: Care must be taken to ensure that the High byte of the I/O address, ADDR[15:8], is only set to 01h when a state change is required on the MACC DATA and CALC banks.

MACC Dual Bank Operation

The Multiply-Accumulator is divided into two separate operating banks. As a result, one bank of the MACC performs a set of multiply-accumulate operations on a set of data while the eZ80190 device is preparing the other bank for the next set of multiply-accumulate operations. Each bank features a separate accumulator and a separate set of control register values (MACC_xSTART, MACC_xEND, MACC_xRELOAD, MACC_ySTART, MACC_yEND, MACC_VRELOAD, and MACC_LENGTH).

The MACC bank that is currently accessible in the eZ80190 device's I/O space is referred to as the DATA bank. The MACC bank that is currently available for use by the MACC for execution is referred to as the CALC bank. The current state of the DATA bank is provided by the DATA_STAT field (bits [1:0]) of the MACC_STAT register. The current state of the CALC bank is provided by the CALC_STAT field (bits [3:2]) of the MACC_STAT register. An explanation of each bank status code is listed in Table 57 and Table 58 on page 118.

Table 57. MACC DATA Bank Status Codes

DATA Bank Status MACC_STAT[1:0]	Description
00b	The DATA bank is EMPTY. No calculation is set up for execution.

zilog

OUT_SHIFT Function

The OUT_SHIFT field, bits 5:3 of the MACC_CTL register, defines the magnitude of the right-shift that is performed when the CPU reads a result from the MACC Accumulator registers MACC_AC0, MACC_AC1, MACC_AC2, MACC_AC3, and MACC_AC4. The MACC automatically manipulates the shift of the 40-bit value as it is read as a succession of 8-bit values. The READs can be right-shifted 0 to 7 bits depending upon the value of OUT_SHIFT. Because the MACC Accumulator value is a two's-complement value, the upper bits are filled with copies of the sign bit, bit 39, during the right-shift operation.

Example 1—When OUT_SHIFT = 000b, reads from the MACC Accumulator registers are not shifted. If the 40-bit MACC Accumulator value is read using a succession of 8-bit READs, the procedure appears as follows:

1. Read the LSB from the MACC Accumulator

DATA_OUT[7:0] = MACC_AC0[7:0] = MACC Accumulator [7:0]

- Read the second byte from the MACC Accumulator
 DATA OUT[7:0] = MACC AC1[7:0] = MACC Accumulator [15:8]
- Read the third byte from the MACC Accumulator
 DATA_OUT[7:0] = MACC_AC2[7:0] = MACC Accumulator [23:16]
- Read the fourth byte from the MACC Accumulator
 DATA_OUT[7:0] = MACC_AC2[7:0] = MACC Accumulator [31:24]
- 5. Read the MSB from the MACC Accumulator

DATA_OUT[7:0] = MACC_AC2[7:0] = MACC Accumulator [39:32]

Example 2—When OUT_SHIFT = 011b, READs from the MACC Accumulator registers are right-shifted by 3 bits. The 3 msbs are filled with copies of the msb of the 40-bit MACC Accumulator. In this example, assume the MACC Accumulator currently contains a positive number so that the msb is 0. If the 40-bit MACC Accumulator value is read using a succession of 8-bit reads, the procedure appears as follows:

1. Read the LSB from the MACC Accumulator

DATA_OUT[7:0] = MACC_AC0[7:0] = MACC Accumulator [10:3]

- Read the second byte from the MACC Accumulator
 DATA_OUT[7:0] = MACC_AC1[7:0] = MACC Accumulator [18:11]
- Read the third byte from the MACC Accumulator
 DATA_OUT[7:0] = MACC_AC2[7:0] = MACC Accumulator [26:19]

zilog ₁₂₆

- 2. MACC_xEND and MACC_yEND define the end of the *linear* address space for the x and y data, respectively. After either the x or y ending value is reached, the next address is defined by MACC_xRELOAD or MACC_yRELOAD, respectively.
- 3. MACC_xRELOAD and MACC_yRELOAD define the circular address to be used when either the *x* index counter or the *y* index counter reaches the ending value for the linear address space.

An example of address indexing for a MACC calculation is displayed in Figure 24 on page 127. The first value is the address returned by the MACC_xSTART register, taken from the *x* RAM memory location. The address increments linearly until the value is used from the address returned by the MACC_xEND register. Instead of incrementing to the next linear address, the next value is taken from the address returned by the MACC_xRELOAD register. Incrementing recommences until the required number of multiply-accumulate operations is completed, as defined by the value in the MACC_LENGTH register.



To the Mulitply-Accumulator

Figure 23. MACC RAM Block Diagram



1	32
	~

Bit Bosition	Valua	Description
[2:0] IN_SHIFT	000	No left-shift is performed during writes to the MACC Accumulator registers by the CPU. MACC_ACx[39:0] = DATA_IN[39:0]
	001	Writes to the MACC Accumulator registers by the CPU are left- shifted by 1 bit with 1 NOISE bit filling the lsb. MACC_ACx[39:0] = {DATA_IN[38:0], NOISE}
	010	Writes to the MACC Accumulator registers by the CPU are left- shifted by 2 bits with repeated NOISE bits filling the lsbs. MACC_ACx[39:0] = {DATA_IN[37:0], 2{NOISE}}
	011	Writes to the MACC Accumulator registers by the CPU are left- shifted by 3 bits with repeated NOISE bits filling the lsbs. MACC_ACx[39:0] = {DATA_IN[36:0], 3{NOISE}}
	100	Writes to the MACC Accumulator registers by the CPU are left- shifted by 4 bits with repeated NOISE bits filling the lsbs. MACC_ACx[39:0] = {DATA_IN[35:0], 4{NOISE}}
	101	Writes to the MACC Accumulator registers by the CPU are left- shifted by 5 bits with repeated NOISE bits filling the lsbs. MACC_ACx[39:0] = {DATA_IN[34:0], 5{NOISE}}
	110	Writes to the MACC Accumulator registers by the CPU are left- shifted by 6 bits with repeated NOISE bits filling the lsbs. MACC_ACx[39:0] = {DATA_IN[33:0], 6{NOISE}}
	111	Writes to the MACC Accumulator registers by the CPU are left- shifted by 7 bits with repeated NOISE bits filling the lsbs. MACC_ACx[39:0] = {DATA_IN[32:0], 7{NOISE}}

MACC Accumulator Byte 0 Register

The MACC_AC0 register, listed in Table 68 on page 132, contains the LSB (bits 7:0) of the 40-bit MACC Accumulator.

Table 68. MACC Accumulator Byte 0 Register					(MACC_AC0 = E8h)				
Bit	7	6	5	4	3	2	1	0	
Reset	Х	Х	Х	Х	Х	Х	Х	Х	
CPU Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Note: X = Undefined	; R/W = Read/	Write.							

zilog

ZDI Register Name	ZDI Register Function	Reset Value	Page #
ZDI_ID_L	$eZ80^{ earrow earro$	05h	163
ZDI_ID_H	eZ80 Product ID High Byte register	00h	163
ZDI_ID_REV	eZ80 Product ID Revision register	XXh	164
ZDI_STAT	Status register	00h	164
ZDI_RD_L	Read Memory Address Low Byte register	XXh	165
ZDI_RD_H	Read Memory Address High Byte register	XXh	165
ZDI_RD_U	Read Memory Address Upper Byte register	XXh	165
ZDI_RD_MEM	Read Memory Data Value	XXh	166
	ZDI Register NameZDI_ID_LZDI_ID_HZDI_ID_REVZDI_STATZDI_RD_LZDI_RD_HZDI_RD_UZDI_RD_UZDI_RD_MEM	ZDI Register NameZDI Register FunctionZDI_ID_LeZ80® Product ID Low Byte registerZDI_ID_HeZ80 Product ID High Byte registerZDI_ID_REVeZ80 Product ID Revision registerZDI_STATStatus registerZDI_RD_LRead Memory Address Low Byte registerZDI_RD_HRead Memory Address High Byte registerZDI_RD_HRead Memory Address Upper Byte registerZDI_RD_URead Memory Address Upper Byte 	ZDI Register NameZDI Register FunctionReset ValueZDI_ID_LeZ80® Product ID Low Byte register05hZDI_ID_HeZ80 Product ID High Byte register00hZDI_ND_REVeZ80 Product ID Revision registerXXhZDI_STATStatus register00hZDI_RD_LRead Memory Address Low Byte registerXXhZDI_RD_HRead Memory Address High ByteXXhZDI_RD_HRead Memory Address Upper Byte registerXXhZDI_RD_URead Memory Data ValueXXh

Table 82. ZDI Read Only Registers

ZDI Register Definitions

ZDI Address Match Registers

The four sets of address match registers are used for setting the addresses for generating break points. When the accompanying BRK_ADDRX bit is set in the ZDI Break Control register to enable the particular address match, the current eZ80190 device address is compared with the 3-byte address set, {ZDI_ADDRx_U, ZDI_ADDRx_H, ZDI_ADDR_x_L}. If the CPU is operating in ADL mode, the address is provided by ADDR[23:0]. If the CPU is operating in Z80[®] mode, the address is provided by {MBASE[7:0], ADDR[15:0]}. If a match is found, ZDI issues a break to the eZ80190 device placing the processor in ZDI mode pending further instructions from the ZDI interface block. If the address is not the first op-code fetch, the ZDI break is executed at the end of the instruction in which it is executed. There are four sets of address match registers. They can be used in conjunction with each other to break on branching instructions.

Note: Due to pipelining functions within the CPU, if the ZDI match address is placed 1 or 2 bytes after completion of a repeating instruction (such as LDIR), the break is issued following completion of only a single cycle of the repeat. When execution is resumed, the repeating instruction completes as required.

zilog[°]

157

Bit Position	Value	Description
6	0	The ZDI break upon matching break address 3 is disabled
BRK_ADDR3	1	The ZDI break, upon matching break address 3, is enabled. ZDI asserts a break when the CPU address, ADDR[23:0], matches the value in the ZDI Address Match 3 registers, {ZDI_ADDR3_U, ZDI_ADDR3_H, ZDI_ADDR3_L}. Breaks can only occur on an instruction boundary. If the address is not the beginning of an instruction, then the break occurs at the end of the current instruction. The break is implemented by setting the BRK_NEXT bit to 1.The BRK_NEXT bit must be reset to 0 to release the break.
5	0	The ZDI break, upon matching break address 2, is disabled.
BRK_ADDR2	1	The ZDI break, upon matching break address 2, is enabled. ZDI asserts a break when the CPU address, ADDR[23:0], matches the value in the ZDI Address Match 2 registers, {ZDI_ADDR2_U, ZDI_ADDR2_H, ZDI_ADDR2_L}. Breaks can only occur on an instruction boundary. If the address is not the beginning of an instruction, then the break occurs at the end of the current instruction. The break is implemented by setting the BRK_NEXT bit to 1. The BRK_NEXT bit must be reset to 0 to release the break.
4	0	The ZDI break, upon matching break address 1, is disabled.
BKK_ADDR1	1	The ZDI break, upon matching break address 1, is enabled. ZDI asserts a break when the CPU address, ADDR[23:0], matches the value in the ZDI Address Match 1 registers, {ZDI_ADDR1_U, ZDI_ADDR1_H, ZDI_ADDR1_L}. If the IGN_LOW_1 bit is set to 1, ZDI asserts a break with the upper two bytes of the CPU address, ADDR[23:8], and matches the value in the ZDI Address Match 1 High and Low Byte registers, {ZDI_ADDR1_U, ZDI_ADDR1_LH}. The lower byte of the address is ignored. Breaks can only occur on an instruction boundary. If the address is not the beginning of an instruction, then the break occurs at the end of the current instruction. The break is implemented by setting the BRK_NEXT bit to 1. The BRK_NEXT bit must be reset to 0 to release the break.

zilog ₁₆₆

register is accessed. As a result, the ZDI master can read any number of data bytes from this address one time, then continue to any number of 8-bit data bytes.

Table 94. ZDI Read Memory Data Value Register (ZDI_RD_MEM = 20h in ZDI Register Read Only Address Space)

Bit	7	6	5	4	3	2	1	0	
Reset	0	0	0	0	0	0	0	0	
CPU Access	R	R R	R	R	R	R	R	R	
Note: R = Read Only.									
Bit Position	Value	Descri	ption						
[7:0] ZDI_RD_MEM	00h– FFh	8-bit data read from the memory address indicated by the CPU's program counter. In Z80 mode, 8-bit data transferred out \leftarrow ({MBASE, SPS}). In ADL mode, 8-bit data transferred out \leftarrow (SPL).							



GPIO Interrupts 46 GPIO Mode 1 44 GPIO Mode 2 44, 45 GPIO Mode 3 44 GPIO Mode 3 44 GPIO Mode 5 44 GPIO Mode 6 45, 47 GPIO Mode 7 45 GPIO Mode 8 45 GPIO Mode 9 45, 47 GPIO Mode 9 45, 47 GPIO mode control registers 43 GPIO Operation 43 GPIO Operation 43 GPIO ports 43

Η

HALT 12 hand-shake capability 67 high-impedance output 44, 45

I

I/O Chip Select Operation 53 I/O Chip Select Precaution 54 I/O Chip Selects 50, 53, 55, 56 I/O Chip Selects, external 23 I/O space 6, 7, 8, 9, 10, 12, 50, 53 I/O space, eZ80 CPU 113 I/O space, eZ80 Webserver 117 I2C Acknowledge 94 I2C Acknowledge bit 102, 107 **I2C** bus 92 I2C bus protocol 93 I2C Clock Control Register 110 **I2C Control Register** 106 I2C Data Register 106 I2C Extended Slave Address Register 105 **I2C General Characteristics** 92 I2C Registers 104 I2C Serial Clock 14

I2C Serial I/O Interface 92 I2C Slave Address Register 105 I2C Software Reset Register 112 I2C Status Register 108 **I2Cx** 98 I2Cx CTL register 92, 97, 100, 102, 103, 104, 106, 109, 110 I2Cx DR register 97, 100, 103, 106, 107 I2Cx SAR byte 105 I2Cx SR register 97, 98, 99, 100, 102, 103, 104, 108, 109 I2Cx xSAR register 105 IDLE state 88, 92, 98, 99, 100, 101, 102, 103, 104, 107, 110 Idle State, SCK 86 IEN 106, 107, 108 IFLG bit 92, 97, 100, 102, 103, 104, 107, 109, 110 IM 0, Op Code Map 175 IM 1, Op Code Map 175 IM 2, Op Code Map 175 IN_SHIFT and OUT_SHIFT 121 **IN SHIFT Function 121** in-circuit emulation 147 INI2R block 115, 116 INI2R instruction 116, 120, 125 initial count value 32, 42 Input/Output Instructions 168 Input/Output Request 12 **INSTRD 5, 12** Instruction READ 12 instruction READ operation 54 Instruction Store 4 0 Registers 161 Inter-Integrated Circuit serial bus 62 internal DMA controllers 22 internal pull-up 12, 44 internal RAM 51, 60 Interrupt Controller 136, 137 interrupt enable 12, 130 interrupt enable bit 70, 142

PC1 15

PC2 15 PC3 16 PC4 16 PC5 16 PC6 16 PC7 17 PD0 17 PD1 18 PD2 18 PD3 19 PD4 19 PD5 19 PD6 20 PD7 20 pen 77 PE-see parity error 80, 172 **PHI 178** Pin Description 4 Poll Mode Transfers 72 POP, Op Code Map 172, 174, 176 port pin value 45, 46 Port x Alternate Register 1 48 Port x Alternate Register 2 48 Port x Data Direction Registers 48 Port x Data Registers 47 Power connections 2 Processor Control Instructions 169 **Program Control Instructions** 170 Programmable Reload Counter/Timers 23 Programmable Reload Timer duration 32 Programmable Reload Timer Operation 32 Programmable Reload Timers 153 Programmable Reload Timers Overview 31 protocol, asynchronous communications 67, 68, 69 protocol, bidirectional serial 148 protocol, I2C bus 93 protocol, reception 68 prt en bit 32 prt irq 34

prt_irq bit 34 prt_irq—see timer interrupt flag 34, 35 pull-down resistor, external 44 pull-up resistor, external 44, 92 pull-up resistor, internal 12, 44 PUSH, Op Code Map 172, 174, 176

R

RAM Address Upper Byte Register 60 RAM Address Upper Byte register 59 **RAM Control Register 60 RAM Control Registers** 60 RC rise times 12 **RD** 5 Reading the Current Count Value 34 receive data ready condition 72 receive FIFO 68, 70, 73, 76 receive FIFO pointer 76 receive FIFO trigger level 76 receiver data ready interrupt event 70 receiver data ready logic 69 receiver interrupt 70 **Receiver Interrupt, UART 70** receiver line error 69 Receiver overrun detection 67 receiver shift register 69, 81 **Recommended Operation 124** Recommended Usage of the Baud Rate Generator 63 Register Map 23 Reload Register 23, 28 reload register, 16-bit 31 **Reload Value 32** Reload Value, PRT 33, 34 Reload Value, timer 32 reload value, timer 35, 37, 38 Request to Send 15, 18, 79 **RESET** 12, 63 **RESET Or NMI Generation 40** Reset States 51



206



Timer Reload Low Byte Register 37 timer reload value 35, 37, 38 Trailing Edge Change on Ring Indicator 82 Transferring Data 93 transmit FIFO 73, 74, 76, 77 Transmit Holding Register bit 80 Transmit Holding Register FIFO bit 80 Transmit Shift Register 80 transmit shift register 68, 69, 77 Transmit Shift Register, SPI 26, 90, 91 Transmit Shift register, SPI 89 trigger-level detection logic 68 **TXD** 73 TXD output 68, 69, 77 TXD pin 68 **TXD** signal 69 **TxD0** 17 TxD1 14

U

UART FIFO Control Register 76 UART Functional Description 68 **UART Functions** 68 UART Interrupt Enable Register 74 **UART Interrupt Identification Register** 75 **UART Interrupts** 70 **UART** interrupts 74 **UART Line Control Register 77 UART Line Status Register 79 UART Modem Control 69** UART Modem Control Register 78 **UART Modem Status Interrupt** 71 UART Modem Status Register 81 **UART Receive Buffer 73** UART Receive Buffer Register 73 **UART Receiver** 69 **UART Receiver Interrupts** 70 **UART Recommended Usage** 71 **UART Registers** 72 **UART Scratch Pad Register 82**

UART Transmit Holding Register 73 UART Transmitter 68 UART Transmitter Interrupt 70 UARTx_IER register 74 UARTx IIR register 72, 75, 76 UARTx LCTL 63, 64, 65, 69, 71 UARTx_LSR register 68, 69, 70, 71, 72, 77, 80, 81 UARTx MSR register 71, 72, 82 UARTx_RBR register 69, 72, 81 UARTx THR register 65, 68, 70, 72 Universal Asynchronous Receiver/ Transmitter 67 Universal Zilog Interface 1, 62, 84 Universal Zilog Interface Blocks, Register Map 26 UZI 62, 84 UZI and BRG Control Registers 64 UZI Baud Rate Generator 63, 85 **UZI Control register 85** UZI control register 63 **UZI Control Registers** 64

UZI interface 14, 15, 16, 17, 18, 19, 20

V

VCC₂ vectored interrupt function 137

W

WAIT state 57, 94, 97, 137 Wait State Timing for Read Operations 190 Wait State Timing for Write Operations 191 Wait States 54 Watchdog Timer 38, 39 Watchdog Timer Overview 39 Watchdog Timer Reset Register 41 Watchdog Timer, Register Map 24 WCOL—see write collision 87, 88, 90 WDT time-out 40, 41