zilog

eZ80190

Product Specification

PS006614-1208

Copyright ©2008 by Zilog[®], Inc. All rights reserved. www.zilog.com

zilog[°]



LIFE SUPPORT POLICY

ZILOG'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS PRIOR WRITTEN APPROVAL OF THE PRESIDENT AND GENERAL COUNSEL OF ZILOG CORPORATION.

As used herein

Life support devices or systems are devices which (a) are intended for surgical implant into the body, or (b) support or sustain life and whose failure to perform when properly used in accordance with instructions for use provided in the labeling can be reasonably expected to result in a significant injury to the user. A critical component is any component in a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system or to affect its safety or effectiveness.

Document Disclaimer

©2008 by Zilog, Inc. All rights reserved. Information in this publication concerning the devices, applications, or technology described is intended to suggest possible uses and may be superseded. ZILOG, INC. DOES NOT ASSUME LIABILITY FOR OR PROVIDE A REPRESENTATION OF ACCURACY OF THE INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED IN THIS DOCUMENT. ZILOG ALSO DOES NOT ASSUME LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT RELATED IN ANY MANNER TO USE OF INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED HEREIN OR OTHERWISE. The information contained within this document has been verified according to the general principles of electrical and mechanical engineering.

eZ80 and Z80 are registered trademarks of Zilog Inc. All other product or service names are the property of their respective owners.



ISO 9001:2000 FS 507510 Zilog products are designed and manufactured under an ISO registered 9001:2000 Quality Management System. For more details, please visit www.zilog.com/quality.

zilogi

Revision History

Each instance in Revision History reflects a change to this document from its previous revision. For more details, refer to the corresponding pages and appropriate links in the table below.

Date	Revision Level	Description	Page Number
December 2008	14	Updated as per latest template and style guide changes. Removed preliminary.	All
March 2006	13	Added registered trademark to eZ80 and eZ80Acclaim	All
February 2004	12	Revised BRG divisor language, UART section	57
July 2003	11	Correction to Timer Reload Register in Programmable Reload Timer Registers section	29
December 2002	10	Characterization data revision, removed web server from title/headers	All

zilog[°] _{iv}

Table of Contents

Architectural Overview	1
General Description	1
Features	1
Block Diagram	2
Pin Description	4
Register Map	. 23
eZ80 [®] CPU Core	. 30
eZ80 CPU Core Overview	. 30
eZ80 CPU Core Features	. 30
Programmable Reload Timers	. 31
Programmable Reload Timers Overview	. 31
Programmable Reload Timer Operation	. 32
Setting Timer Duration	. 32
Single Pass Mode	. 32
Continuous Mode	. 33
Reading the Current Count Value	. 34
Timer Interrupts	. 34
Programmable Reload Timer Registers	. 34
Timer Control Registers	. 35
Timer Data Low Byte Register	. 36
Timer Data High Byte Register	. 36
Timer Reload Low Byte Registers	. 37
	. 38
	. 39
WDT Overview	. 39
	. 40
Enabling And Disabling The WDT	. 40
	. 40
	. 40
WDT Operated Deviator	. 40
	. 40
	. 41
	. 43
	. 43
GPIO Operation	. 43

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



UART Interrupts	70
UART Transmitter Interrupt	70
UART Receiver Interrupts	70
UART Modem Status Interrupt	71
UART Recommended Usage	71
Module Reset	71
Control Transfers	71
Data Transfers	72
Poll Mode Transfers	72
UART Registers	72
UART Transmit Holding Register	73
UART Receive Buffer Register	73
UART Interrupt Enable Register	74
UART Interrupt Identification Register	75
UART FIFO Control Registers	76
UART Line Control Register	77
UART Modem Control Registers	78
UART Line Status Registers	79
UART Modem Status Registers	81
UART Scratch Pad Registers	82
Serial Peripheral Interface	84
SPI Signals	84
Master In Slave Out	84
Master Out Slave In	85
Slave Select	85
Serial Clock	85
SPI Functional Description	86
SPI Flags	88
Mode Fault	88
Write Collision	88
SPI Registers	89
SPI Control Register	89
SPI Status Register	90
SPI Transmit Shift Register	90
SPI Receive Buffer Register	91
I ² C Serial I/O Interface	92
I ² C General Characteristics	92
Clocking Overview	~~
	92
Bus Arbitration Overview	92 92

zilog[°] _{vii}

START and STOP Conditions
Transferring Data
Byte Format
Acknowledge
Clock Synchronization
Arbitration
Clock Synchronization for Handshake
Operating Modes
Master Transmit
Master Receive
Slave Receive 103
I^2 C Registers
Addressing 104
Resetting the I ² C Registers
I ² C Slave Address Register 104
I ² C Extended Slave Address Register
I ² C Data Register 106
I ² C Control Register 106
I ² C Status Register 108
I ² C Clock Control Register 110
Bus Clock Speed
Multiply Accumulator
Multisly Assessmentator Desis On pretion
Software Control of the MACC
Defining the DATA Bank as EMPTY 116
Alternatives to OTI2R and INI2R 117
MACC Dual Bank Operation 117
IN SHIFT and OUT SHIFT 121
IN SHIFT Function
OUT SHIFT Function
Recommended Operation 124
Setting Up A New Calculation
Retrieve A Calculation 124
MACC RAM
MACC RAM Address Indexing 125

zilog[®] _{viii}

Multiply-Accumulator Control And Data Registers AACC x DATA Starting Address Register MACC x DATA Ending Address Register MACC x DATA Reload Address Register MACC Length Register MACC y DATA Starting Address Register MACC y DATA Starting Address Register MACC y DATA Ending Address Register MACC y DATA Reload Address Register MACC y DATA Ending Address Register MACC y DATA Ending Address Register MACC Control Register MACC Control Register MACC Accumulator Byte 0 Register MACC Accumulator Byte 1 Register MACC Accumulator Byte 2 Register MACC Accumulator Byte 3 Register MACC Accumulator Byte 3 Register	127 128 128 129 129 130 130 130 132 133 133
MACC Accumulator Byte 4 Register	134
MACC Status Register	135
Interrupt Controller	137
Direct Memory Access Controller	140
DMA Programming	140
DMA Transfer Modes	141
DMA Channel Priorities	142
DMA Interrupts	142
DMA Control Registers	142
DMA Source Address Registers	143
DMA Destination Address Registers	144
DMA Byte Count Registers	144
	145
	147
	147
	148
2DI Clock and Data Conventions	148
ZDI Start Condition ZDI Single-Bit Byte Separator	149 150
ZDI Register Addressing	150
ZDI Write Operations	151
ZDI Single-Byte Write	151
ZDI Block Write	151
ZDI Read Operations	152
ZDI Single-Byte Read	152
ZDI Block Read	153

zilog _{ix}

Operation Of The eZ80190 Device During ZDI Breakpoints	153
ZDI Write Only Registers	153
ZDI Read Only Registers	154
ZDI Register Definitions	155
ZDI Address Match Registers	155
ZDI Break Control Register	156
ZDI Write Data Registers	159
ZDI Read/Write Control Register	159
Instruction Store 4:0 Registers	161
	162
eZ80 [®] Product ID Low Byte Register	162
eZ80 Product ID High Byte Register	163
eZ80° Product ID Revision Register	163
ZDI Status Register Low High and Upper	104
ZDI Read Memory Data Value Register	165
e780 [®] CPU Instruction Set	167
On-Code Man	172
Crystal Oscillator	179
	180
	100
	100
	101
	184 495
	100
External I/O Read Timing	100
External I/O Write Timing	189
Wait State Timing for Read Operations	190
Wait State Timing for Write Operations	191
General Purpose I/O Port Input Sample Timing	192
General Purpose I/O Port Output Timing	193
External Bus Acknowledge Timing	194
External System Clock Driver Timing	194
Packaging	195
Ordering Information	196
Part Number Description	196

PS006614-1208

zilog _x

Index	198
Customer Support	211



Architectural Overview

General Description

Zilog's eZ80190 microprocessor is a high-speed single-cycle instruction-fetch microprocessor with a clock speed of up to 50 MHz. It is the first of a new set of products based upon Zilog's eZ80[®] CPU.

The eZ80 CPU is one of the fastest 8-bit CPUs available today, executing code up to four times faster with zero wait-state memory than a standard $Z80^{\text{(R)}}$ operating at the same frequency. This increased processing efficiency can be used to improve available bandwidth or to decrease power consumption.

Considering both the high clock speed and instruction pipeline efficiency, the eZ80 CPU's processing power rivals the performance of 16-bit microprocessors.

Features

The key features of eZ80190 microprocessor are as follows:

- Single-cycle instruction fetch, high-performance eZ80 CPU core¹
- 16 x 16-bit Multiply and 40-bit Accumulate with 1 KB dual-port SRAM
- Four Chip Selects with individual Wait State generators
- Six Counter/Timers with prescalers
- Watchdog Timer (WDT)
- 2-channel Direct Memory Access (DMA) controller
- 8 KB high-speed data SRAM
- 2 Universal Zilog Interface (UZI) channels (I²C, SPI, UART) with built-in Baud Rate Generator
- Fixed-priority vectored interrupts (32 external, 11 internal)
- 32 bits of General-Purpose Input/Output (GPIO)
- On-chip oscillator
- 3.0 V to 3.6 V supply voltage with 5 V tolerant inputs
- 100-pin LQFP package

^{1.} For simplicity, the term *eZ80 CPU* is referred to as *CPU* for the bulk of this document.



- Up to 50 MHz clock speed
- Operating Temperature:
 - Standard Temperature Range: 0 °C to +70 °C
 - Extended Temperature Range: -40 °C to +105 °C
- Zilog Debug Interface (ZDI)

Note: All signals with an overline are active Low. For example, B/\overline{W} , for which WORD is active Low, and \overline{B}/W , for which BYTE is active Low.

Power connections follow these conventional descriptions:

Connection	Circuit	Device	
Power	$V_{\rm CC}$	V _{DD}	
Ground	GND	V _{SS}	

Block Diagram

Figure 1 on page 3 displays a block diagram of the eZ80190 processor.





Figure 1. eZ80190 Block Diagram



Pin Description

Figure 2 displays the pin layout of the eZ80190 device in the 100-pin LQFP package. Table 1 on page 5 lists the pins and their functions.



Figure 2. 100-Pin LQFP Configuration of the eZ80190 Device

zilog ₅

Pin No.	Symbol	Function	Signal Direction	Description
1	MREQ	Memory Request	Input/Output, Active Low	MREQ indicates the <u>CPU</u> is accessing a location in memory. The RD, WR, and INSTRD signals indicate the type of access. The eZ80190 device does not drive this line during Reset. It is an input in bus acknowledge cycles.
2	WR	Write	Output, Active Low	WR indicates the CPU is writing to the current address location. The device accessed is determined by the IORQ and MREQ pins. The WR pin is tristated during bus acknowledge cycles.
3	RD	Read	Output, Active Low	$\overline{\text{RD}}$ indicates the eZ80190 device is reading from the current address location. This pin is tristated during bus acknowledge cycles.
4	CS0	Chip Select 0	Output, Active Low	$\overline{\text{CS0}}$ indicates access in the defined $\overline{\text{CS0}}$ memory or I/O address space. This signal is still driven during bus acknowledge cycles and is generated from the address and control provided on the external pins.
5	CS1	Chip Select 1	Output, Active Low	$\overline{\text{CS1}}$ indicates access in the defined $\overline{\text{CS1}}$ memory or I/O address space. This signal is still driven during bus acknowledge cycles and is generated from the address and control provided on the external pins.
6	CS2	Chip Select 2	Output, Active Low	$\overline{\text{CS2}}$ indicates access in the defined $\overline{\text{CS2}}$ memory or I/O address space. This signal is still driven during bus acknowledge cycles and is generated from the address and control provided on the external pins.
7	CS3	Chip Select 3	Output, Active Low	$\overline{\text{CS3}}$ indicates access in the defined $\overline{\text{CS3}}$ memory or I/O address space. This signal is still driven during bus acknowledge cycles and is generated from the address and control provided on the external pins.
8	V _{DD}	Power Supply		Power Supply
9	GND	Ground		Ground

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.

zilog,

Pin No.	Symbol	Function	Signal Direction	Description
16	ADDR6	Address Bus	Input/Output	The ADDR6 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.
17	ADDR7	Address Bus	Input/Output	The ADDR7 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.
18	V_{DD}	Power Supply		Power Supply
19	GND	Ground		Ground
20	ADDR8	Address Bus	Input/Output	The ADDR8 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.
21	ADDR9	Address Bus	Input/Output	The ADDR9 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.
22	ADDR10	Address Bus	Input/Output	The ADDR10 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.

zilog ₈

Pin No.	Symbol	Function	Signal Direction	Description
23	ADDR11	Address Bus	Input/Output	The ADDR11 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.
24	ADDR12	Address Bus	Input/Output	The ADDR12 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.
25	ADDR13	Address Bus	Input/Output	The ADDR13 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.
26	ADDR14	Address Bus	Input/Output	The ADDR14 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.
27	ADDR15	Address Bus	Input/Output	The ADDR15 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.
28	V _{DD}	Power Supply		Power Supply
29	GND	Ground		Ground

zilog ₉

Pin No.	Symbol	Function	Signal Direction	Description
30	ADDR16	Address Bus	Input/Output	The ADDR16 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.
31	ADDR17	Address Bus	Input/Output	The ADDR17 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.
32	ADDR18	Address Bus	Input/Output	The ADDR18 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.
33	ADDR19	Address Bus	Input/Output	The ADDR19 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.
34	ADDR20	Address Bus	Input/Output	The ADDR20 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.

zilog ₁₀

Pin No.	Symbol	Function	Signal Direction	Description
35	ADDR21	Address Bus	Input/Output	The ADDR21 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.
36	ADDR22	Address Bus	Input/Output	The ADDR22 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.
37	ADDR23	Address Bus	Input/Output	The ADDR23 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.
38	V _{DD}	Power Supply		Power Supply
39	GND	Ground		Ground
40	DATA0	Data Bus	Bidirectional, tristate	The data bus transfers data to and from I/O and memory devices. The eZ80190 device drives these lines only during write cycles when the eZ80190 device is the bus master. The data bus is configured as an output in normal operation and as an input during bus acknowledge cycles.
41	DATA1	Data Bus	Bidirectional, tristate	The data bus transfers data to and from I/O and memory devices. The eZ80190 device drives these lines only during write cycles when the eZ80190 device is the bus master. The data bus is configured as an output in normal operation and as an input during bus acknowledge cycles.

zilog ₁₁

Pin No.	Symbol	Function	Signal Direction	Description
42	DATA2	Data Bus	Bidirectional, tristate	The data bus transfers data to and from I/O and memory devices. The eZ80190 device drives these lines only during write cycles when the eZ80190 device is the bus master. The data bus is configured as an output in normal operation and as an input during bus acknowledge cycles.
43	DATA3	Data Bus	Bidirectional, tristate	The data bus transfers data to and from I/O and memory devices. The eZ80190 device drives these lines only during write cycles when the eZ80190 device is the bus master. The data bus is configured as an output in normal operation and as an input during bus acknowledge cycles.
44	DATA4	Data Bus	Bidirectional, tristate	The data bus transfers data to and from I/O and memory devices. The eZ80190 device drives these lines only during write cycles when the eZ80190 device is the bus master. The data bus is configured as an output in normal operation and as an input during bus acknowledge cycles.
45	DATA5	Data Bus	Bidirectional, tristate	The data bus transfers data to and from I/O and memory devices. The eZ80190 device drives these lines only during write cycles when the eZ80190 device is the bus master. The data bus is configured as an output in normal operation and as an input during bus acknowledge cycles.
46	DATA6	Data Bus	Bidirectional, tristate	The data bus transfers data to and from I/O and memory devices. The eZ80190 device drives these lines only during write cycles when the eZ80190 device is the bus master. The data bus is configured as an output in normal operation and as an input during bus acknowledge cycles.
47	DATA7	Data Bus	Bidirectional, tristate	The data bus transfers data to and from I/O and memory devices. The eZ80190 device drives these lines only during write cycles when the eZ80190 device is the bus master. The data bus is configured as an output in normal operation and as an input during bus acknowledge cycles.
48	V _{DD}	Power Supply		Power Supply
49	GND	Ground		Ground

zilog ₁₂

Pin No.	Symbol	Function	Signal Direction	Description
50	NMI	Nonmaskable Interrupt	Schmitt Trigger Input, Active Low	The $\overline{\text{NMI}}$ input is prioritized higher than the maskable interrupts. It is always recognized at the end of an instruction, regardless of the state of the interrupt enable control bits. This input includes a Schmitt trigger to allow RC rise times. This ext <u>ernal NMI signal is combined with an</u> internal NMI signal generated from the WDT block before being connected to the NMI input of the CPU.
51	HALT	Halt	Output, Active Low	A Low on this pin indicates the CPU has stopped because a HALT instruction is executed.
52	INSTRD	Instruction READ	Output, Active Low, tristate	INSTRD (with MREQ and RD) indicates the eZ80190 device is fetching an instruction from code memory. The eZ80190 device does not drive this line during Reset or bus acknowledge cycles.
53	IORQ	Input/Output Request	Input/Output, Active Low	IORQ indicates the CPU is accessing a location in I/O space. RD and WR indicate the type of access. The eZ80190 device does not drive this line during Reset and is an input in bus acknowledge cycles.
54	RESET	Reset	Schmitt Trigger Input, Active Low	This signal is used to initialize the eZ80190 device. This input must be Low for a minimum of 3 system clock cycles, and must be held Low until the clock is stable. This input includes a Schmitt trigger to allow RC rise times.
55	ZCL	ZDI Clock	Input with Pull-up	The ZCL pin is used to clock the data between the Zilog Debug Interface and the eZ80190 device. This pin features an internal pull-up.
56	ZDA	ZDI Data	Input/Output, Open-Drain with Pull-up	The ZDA pin is used to transfer data between the Zilog Debug Interface and the eZ80190 device. This pin is open-drain and features an internal pull-up.
57	PB0	GPIO Port B	Input/Output	The PB0 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port B pin, when programmed as an output, can be selected to be an open-drain or open-source output.

zilog ₁₃

Pin No.	Symbol	Function	Signal Direction	Description
58	PB1	GPIO Port B	Input/Output	The PB1 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port B pin, when programmed as an output, can be selected to be an open-drain or open-source output.
59	PB2	GPIO Port B	Input/Output	The PB2 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port B pin, when programmed as an output, can be selected to be an open-drain or open-source output.
60	PB3	GPIO Port B	Input/Output	The PB3 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port B pin, when programmed as an output, can be selected to be an open-drain or open-source output.
61	PB4	GPIO Port B	Input/Output	The PB4 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port B pin, when programmed as an output, can be selected to be an open-drain or open-source output.
62	PB5	GPIO Port B	Input/Output	The PB5 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port B pin, when programmed as an output, can be selected to be an open-drain or open-source output.
63	PB6	GPIO Port B	Input/Output	The PB6 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port B pin, when programmed as an output, can be selected to be an open-drain or open-source output.

zilog[®] ₁₄

Pin				
No.	Symbol	Function	Signal Direction	Description
64	PB7	GPIO Port B	Input/Output	The PB7 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port B pin, when programmed as an output, can be selected to be an open-drain or open-source output.
65	V_{DD}	Power Supply		Power Supply
66	GND	Ground		Ground
67	PC0	GPIO Port C	Input/Output	The PC0 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port C pin, when programmed as an output, can be selected to be an open-drain or open-source output. Port C is multiplexed with one channel of the UZI interface.
	MISO1	Master In Slave Out	Input/Output	The MISO line is configured as an input when the eZ80190 device is an SPI master device and as an output when eZ80190 device is an SPI slave device. This signal is multiplexed with PC0.
	SCL1	l ² C Serial Clock	Input/Output	The SCL1 pin is used to receive and transmit the I^2C clock. This signal is multiplexed with PC0.
	TxD1	Transmit Data	Output	The TxD1 pin is used by the UART to transmit asynchronous serial data. This signal is multiplexed with PC0.

zilog ₁₅

Pin No.	Symbol	Function	Signal Direction	Description
68	PC1	GPIO Port C	Input/Output	The PC1 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port C pin, when programmed as an output, can be selected to be an open-drain or open-source output. Port C is multiplexed with one channel of the UZI interface.
	MOSI1	Master Out Slave In	Input/Output	The MOSI line is configured as an output when the eZ80190 device is an SPI master device and as an input when the eZ80190 device is an SPI slave device. This signal is multiplexed with PC1.
	RxD1	Receive Data	Input	The RxD1 pin is used by the UART to receive asynchronous serial data. This signal is multiplexed with PC1.
	SDA1	I ² C Serial Data	Input/Output	The SDA1 pin carries the I^2C data signal. This signal is multiplexed with PC1.
69	PC2	GPIO Port C	Input/Output	The PC2 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port C pin, when programmed as an output, can be selected to be an open-drain or open-source output. Port C is multiplexed with one channel of the UZI interface.
	SCK1	SPI Serial Clock	Input/Output	SPI serial clock. This signal is multiplexed with PC2.
	RTS1	Request to Send	Output, Active Low	The RTS1 pin carries the modem-control signal from the UART. This signal is multiplexed with PC2.

zilog ₁₆

Pin No.	Symbol	Function	Signal Direction	Description
70	PC3	GPIO Port C	Input/Output	The PC3 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port C pin, when programmed as an output, can be selected to be an open-drain or open-source output. Port C is multiplexed with one channel of the UZI interface.
	SS1	Slave Select	Input, Active Low	The slave select input line is used to select a slave device in SPI mode. This signal is multiplexed with PC3.
	CTS1	Clear to Send	Input, Active Low	The $\overline{\text{CTS1}}$ pin carries the modem status signal to the UART. This signal is multiplexed with PC3.
71	PC4	GPIO Port C	Input/Output	The PC4 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port C pin, when programmed as an output, can be selected to be an open-drain or open-source output. Port C is multiplexed with one channel of the UZI interface.
	DTR1	Data Terminal Ready	Output, Active Low	The DTR1 pin carries the modem-control signal to the UART. This signal is multiplexed with PC4.
72	PC5	GPIO Port C	Input/Output	The PC5 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port C pin, when programmed as an output, can be selected to be an open-drain or open-source output. Port C is multiplexed with one channel of the UZI interface.
	DSR1	Data Set Ready	Input, Active Low	The $\overline{\text{DSR1}}$ pin carries the modem status signal to the UART. This signal is multiplexed with PC5.
73	PC6	GPIO Port C	Input/Output	The PC6 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port C pin, when programmed as an output, can be selected to be an open-drain or open-source output. Port C is multiplexed with one channel of the UZI interface.
	DCD1	Data Carrier Detect	Input, Active Low	The $\overline{\text{DCD1}}$ pin carries the modem status signal to the UART. This signal is multiplexed with PC6.

zilog[®] ₁₇

Pin No.	Symbol	Function	Signal Direction	Description
74	PC7	GPIO Port C	Input/Output	The PC7 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port C pin, when programmed as an output, can be selected to be an open-drain or open-source output. Port C is multiplexed with one channel of the UZI interface.
	RI1	Ring Indicator	Input, Active Low	The $\overline{\text{RI1}}$ pin carries the modem status signal to the UART. This signal is multiplexed with PC7.
75	TEST	Test	Input, Active High	The TEST pin places the chip in TEST mode. It is used only for factory testing. This signal should be tied Low for normal operation.
76	PD0	GPIO Port D	Input/Output	The PD0 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port D pin, when programmed as an output, can be selected to be an open-drain or open-source output. Port D is multiplexed with one channel of the UZI interface.
	MISO0	Master In Slave Out	Input/Output	The MISO line is configured as an input when the eZ80190 device is an SPI master device and as an output when eZ80190 device is an SPI slave device. This signal is multiplexed with PD0.
	SCL0	l ² C Serial Clock	Input/Output	This pin is used to receive and transmit the I^2C clock. This signal is multiplexed with PD0.
	TxD0	Transmit Data	Output	The TxD0 pin is used by the UART to transmit asynchronous serial data. This signal is multiplexed with PD0.

zilog ₁₈

Pin No.	Symbol	Function	Signal Direction	Description
77	PD1	GPIO Port D	Input/Output	The PD1 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port D pin, when programmed as an output, can be selected to be an open-drain or open-source output. Port D is multiplexed with one channel of the UZI interface.
	MOSI0	Master Out Slave In	Input/Output	The MOSI line is configured as an output when the eZ80190 device is an SPI master device and as an input when the eZ80190 device is an SPI slave device. This signal is multiplexed with PD1.
	RxD0	Receive Data	Input	The RxD0 pin is used by the UART to receive asynchronous serial data. This signal is multiplexed with PD1.
	SDA0	I ² C Serial Data	Input/Output	The SDA0 pin carries the I^2C data signal. This signal is multiplexed with PD1.
78	PD2	GPIO Port D	Input/Output	The PD2 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port D pin, when programmed as an output, can be selected to be an open-drain or open-source output. Port D is multiplexed with one channel of the UZI interface.
	SCK0	SPI Serial Clock	Input/Output	The SPI serial clock signal is multiplexed with PD2.
	RTS0	Request to Send	Output, Active Low	The RTS0 pin carries the modem-control signal from the UART. This signal is multiplexed with PD2.

zilog ₁₉

Pin No.	Symbol	Function	Signal Direction	Description
79	PD3	GPIO Port D	Input/Output	The PD3 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port D pin, when programmed as an output, can be selected to be an open-drain or open-source output. Port D is multiplexed with one channel of the UZI interface.
	SS0	Slave Select	Input, Active Low	The slave select input line is used to select a slave device in SPI mode. This signal is multiplexed with PD3.
	CTS0	Clear to Send	Input, Active Low	The $\overline{\text{CTS0}}$ pin carries the modem status signal to the UART. This signal is multiplexed with PD3.
80	PD4	GPIO Port D	Input/Output	The PD4 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port D pin, when programmed as an output, can be selected to be an open-drain or open-source output. Port D is multiplexed with one channel of the UZI interface.
	DTR0	Data Terminal Ready	Output, Active Low	The DTR0 pin carries the modem control signal to the UART. This signal is multiplexed with PD4.
81	PD5	GPIO Port D	Input/Output	The PD5 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port D pin, when programmed as an output, can be selected to be an open-drain or open-source output. Port D is multiplexed with one channel of the UZI interface.
	DSR0	Data Set Ready	Input, Active Low	The DSR0 pin carries the modem status signal to the UART. This signal is multiplexed with PC5 and PD5.

zilog ₂₀

Pin No.	Symbol	Function	Signal Direction	Description	
82	PD6	GPIO Port D	Input/Output	The PD6 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port D pin, when programmed as an output, can be selected to be an open-drain or open-source output. Port D is multiplexed with one channel of the UZI interface.	
	DCD0	Data Carrier Detect	Input, Active Low	The DCD0 pin carries the modem status signal to the UART. This signal is multiplexed with PC6 and PD6.	
83	PD7	GPIO Port D	Input/Output	The PD7 pin can be used for GPIO. It can be individually programmed as an input or output and can also be used individually as an interrupt input. Each Port D pin, when programmed as an output, can be selected to be an open-drain or open-source output. Port D is multiplexed with one channel of the UZI interface.	
	RI0	Ring Indicator	Input, Active Low	The $\overline{\text{RI0}}$ pin carries the modem status signal to the UART. This signal is multiplexed with PC7 and PD7.	
84	V _{DD}	Power Supply		Power Supply	
85	GND	Ground		Ground	
86	X _{OUT}	Oscillator Output	Output	The X_{OUT} pin is the output of the onboard crystal oscillator. When used, a crystal oscillator should be connected between X_{IN} and X_{OUT} .	
87	X _{IN}	Oscillator Input	Input	The X_{IN} pin is the input to the onboard crystal oscillator. If an external oscillator is used, its clock output should be connected to this pin. When a crystal oscillator is used, it should be connected between X_{IN} and X_{OUT} .	
88	BUSACK	Bus Acknowledge	Output, Active Low	The eZ80190 device responds to a Low on the BUSREQ, by tristating the address, data, and control signals, and by driving the BUSACK line Low. During bus acknowledge cycles A23:0, IORQ, and MREQ are inputs.	

zilog 21

Pin No.	Symbol	Function	Signal Direction	Description
89	PA0	GPIO Port A	Input/Output	The PA0 pin can be used for GPIO. It can be individually programmed as an input or an output and can also be used individually as an interrupt input. Each Port A pin, when programmed as an output, can be selected to be an open-drain or open-source output.
90	PA1	GPIO Port A	Input/Output	The PA1 pin can be used for GPIO. It can be individually programmed as an input or an output and can also be used individually as an interrupt input. Each Port A pin, when programmed as an output, can be selected to be an open-drain or open-source output.
91	PA2	GPIO Port A	Input/Output	The PA2 pin can be used for GPIO. It can be individually programmed as an input or an output and can also be used individually as an interrupt input. Each Port A pin, when programmed as an output, can be selected to be an open-drain or open-source output.
92	PA3	GPIO Port A	Input/Output	The PA3 pin can be used for GPIO. It can be individually programmed as an input or an output and can also be used individually as an interrupt input. Each Port A pin, when programmed as an output, can be selected to be an open-drain or open-source output.
93	PA4	GPIO Port A	Input/Output	The PA4 pin can be used for GPIO. It can be individually programmed as an input or an output and can also be used individually as an interrupt input. Each Port A pin, when programmed as an output, can be selected to be an open-drain or open-source output.
94	PA5	GPIO Port A	Input/Output	The PA5 pin can be used for GPIO. It can be individually programmed as an input or an output and can also be used individually as an interrupt input. Each Port A pin, when programmed as an output, can be selected to be an open-drain or open-source output.

<mark>z</mark>ilog[°]

22

Pin No.	Symbol	Function	Signal Direction	Description	
95	PA6	GPIO Port A	Input/Output	The PA6 pin can be used for GPIO. It can be individually programmed as an input or an output and can also be used individually as an interrupt input. Each Port A pin, when programmed as an output, can be selected to be an open-drain or open-source output.	
96	PA7	GPIO Port A	Input/Output	The PA7 pin can be used for GPIO. It can be individually programmed as an input or an output and can also be used individually as an interrup input. Each Port A pin, when programmed as an output, can be selected to be an open-drain or open-source output.	
97	V _{DD}	Power Supply		Power Supply	
98	GND	Ground		Ground	
99	BUSREQ	Bus Request	Input, Active Low	External devices can force the eZ80190 device to release the bus for their use by driving this line Low. To the CPU, the bus request signal can also originate from internal DMA controllers. In such cases, bus requests from the DMA controllers have a higher priority than a request from an external bus master.	
100	PHI	System Clock	Output	The PHI pin is an output driven by the internal system clock. It can be used by the system for synchronization with the eZ80190 device.	

zilog[°]

Register Map

All on-chip peripheral registers are accessed in the I/O address space. All I/O operations employ 16-bit addresses. The upper byte of the 24-bit address bus is forced to 00h (ADDR[23:16] = 00h) during all I/O operations. All I/O operations using 16-bit addresses within the range of 80h to FFh are routed to the on-chip peripherals; where xx is any value from 00h to FFh. External I/O Chip Selects are not generated if the address space programmed for the I/O Chip Selects overlap the 80h to FFh address range.

Note: Registers at unused addresses within the 80h to FFh range assigned to on-chip peripherals are not implemented. READ access to such addresses return unpredictable values and WRITE access produces no effect. Table 2 lists the register map for the eZ80190 device.

Address (hex)	Mnemonic	Name	Reset (hex)	CPU Access	Page No.		
Program	Programmable Reload Counter/Timers						
80	TMR0_CTL	Timer 0 Control Register	00	R/W	35		
81	TMR0_DR_L	Timer 0 Data Register—Low Byte	00	R	36		
	TMR0_RR_L	Timer 0 Reload Register—Low Byte	00	W	37		
82	TMR0_DR_H	Timer 0 Data Register—High Byte	00	R	36		
	TMR0_RR_H	Timer 0 Reload Register—High Byte	00	W	38		
83	TMR1_CTL	Timer 1 Control Register	00	R/W	35		
84	TMR1_DR_L	Timer 1 Data Register—Low Byte	00	R	36		
	TMR1_RR_L	Timer 1 Reload Register—Low Byte	00	W	37		
85	TMR1_DR_H	Timer 1 Data Register—High Byte	00	R	36		
	TMR1_RR_H	Timer 1 Reload Register—High Byte	00	W	38		
86	TMR2_CTL	Timer 2 Control Register	00	R/W	35		
87	TMR2_DR_L	Timer 2 Data Register—Low Byte	00	R	36		
	TMR2_RR_L	Timer 2 Reload Register—Low Byte	00	W	37		
88	TMR2_DR_H	Timer 2 Data Register—High Byte	00	R	36		
	TMR2_RR_H	Timer 2 Reload Register—High Byte	00	W	38		
89	TMR3_CTL	Timer 3 Control Register	00	R/W	35		

Table 2. Register Map

zilog[°]

Address (hex)	Mnemonic	Name	Reset (hex)	CPU Access	Page No.
8A	TMR3_DR_L	Timer 3 Data Register—Low Byte	00	R	36
	TMR3_RR_L	Timer 3 Reload Register—Low Byte	00	W	37
8B	TMR3_DR_H	Timer 3 Data Register—High Byte	00	R	36
	TMR3_RR_H	Timer 3 Reload Register—High Byte	00	W	38
8C	TMR4_CTL	Timer 4 Control Register	00	R/W	35
8D	TMR4_DR_L	Timer 4 Data Register—Low Byte	00	R	36
	TMR4_RR_L	Timer 4 Reload Register—Low Byte	00	W	37
8E	TMR4_DR_H	Timer 4 Data Register—High Byte	00	R	36
	TMR4_RR_H	Timer 4 Reload Register—High Byte	00	W	38
8F	TMR5_CTL	Timer 5 Control Register	00	R/W	35
90	TMR5_DR_L	Timer 5 Data Register—Low Byte	00	R	36
	TMR5_RR_L	Timer 5 Reload Register—Low Byte	00	W	37
91	TMR5_DR_H	Timer 5 Data Register—High Byte	00	R	36
	TMR5_RR_H	Timer 5 Reload Register—High Byte	00	W	38
92	Not Accessible				
Watchdo	g Timer				
93	WDT_CTL	Watchdog Timer Control Register	00/20 ¹	R/W	41
94	WDT_RR	Watchdog Timer Reset Register	XX	W	41
95	Not Accessible				
General-I	Purpose Input/Outp	out Ports			
96	PA_DR	Port A Data Register	XX	R/W ²	48
97	PA_DDR	Port A Data Direction Register	FF	R/W	48
98	PA_ALT1	Port A Alternate Register 1	00	R/W	48
99	PA_ALT2	Port A Alternate Register 2	00	R/W	49
9A	PB_DR	Port B Data Register	XX	R/W ²	48
9B	PB_DDR	Port B Data Direction Register	FF	R/W	48
9C	PB_ALT1	Port B Alternate Register 1	00	R/W	48

Table 2. Register Map (Continued)

24

zilog[°]

Address (hex)	Mnemonic	Name	Reset (hex)	CPU Access	Page No.			
9D	PB_ALT2	Port B Alternate Register 2	00	R/W	49			
9E	PC_DR Port C Data Register		XX	R/W ²	48			
9F	PC_DDR Port C Data Direction Register		FF	R/W	48			
A0	PC_ALT1 Port C Alternate Register 1		00	R/W	48			
A1	PC_ALT2	Port C Alternate Register 2	00	R/W	49			
A2	PD_DR	Port D Data Register	XX	R/W ²	48			
A3	PD_DDR	Port D Data Direction Register	FF	R/W	48			
A4	PD_ALT1	Port D Alternate Register 1	00	R/W	48			
A5	PD_ALT2	Port D Alternate Register 2	00	R/W	49			
A6	Not Accessible							
A7	Not Accessible							
Chip Sele	ect/Wait State Gene	rator						
A8	CS0_LBR	Chip Select 0 Lower Bound Register	00	R/W	56			
A9	CS0_UBR	Chip Select 0 Upper Bound Register	FF	R/W	56			
AA	CS0_CTL	Chip Select 0 Control Register	E8	R/W	57			
AB	CS1_LBR	Chip Select 1 Lower Bound Register	00	R/W	56			
AC	CS1_UBR	Chip Select 1 Upper Bound Register	00	R/W	56			
AD	CS1_CTL	Chip Select 1 Control Register	00	R/W	57			
AE	CS2_LBR	Chip Select 2 Lower Bound Register	00	R/W	56			
AF	CS2_UBR	Chip Select 2 Upper Bound Register	00	R/W	56			
B0	CS2_CTL	Chip Select 2 Control Register	00	R/W	57			
B1	CS3_LBR	Chip Select 3 Lower Bound Register	00	R/W	56			
B2	CS3_UBR	Chip Select 3 Upper Bound Register	00	R/W	56			
B3	CS3_CTL	Chip Select 3 Control Register	00	R/W	57			
On-Chip	On-Chip RAM Control							
B4	RAM_CTL	RAM Control Register	00	R/W	60			
B5	RAM_ADDR_U	RAM Address Upper Byte	00	R/W	60			

Table 2. Register Map (Continued)

25

zilog[°]

Address (hex)	Mnemonic	Name	Reset (hex)	CPU Access	Page No.			
Universal Zilog Interface Blocks								
B6	SPI0_CTL	SPI 0 Control Register	04	R/W	106			
B7	SPI0_SR	SPI 0 Status Register	00	R	108			
B8	SPI0_RBR	SPI 0 Receive Buffer Register	XX	R	109			
B8	SPI0_TSR	SPI 0 Transmit Shift Register	XX	W	108			
B9	Not Accessible							
BA	SPI1_CTL	SPI 1 Control Register	04	R/W	106			
BB	SPI1_SR	SPI 1 Status Register	00	R	108			
BC	SPI1_RBR	SPI1 Receive Buffer Register	XX	R	109			
BC	SPI1_TSR	SPI1 Transmit Shift Register	XX	W	108			
BD	Not Accessible							
BE	Not Accessible							
BF	Not Accessible							
C0	UART0_RBR	UART 0 Receive Buffer Register	XX	R	89			
	UART0_THR	UART 0 Transmit Holding Register	XX	W	86			
	BRG0_DLR_L	BRG 0 Divisor Latch Register—Low Byte	02	R/W	65			
C1	BRG0_DLR_H	BRG 0 Divisor Latch Register—High Byte	00	R/W	65			
	UART0_IER	UART 0 Interrupt Enable Register	00	R/W	90			
C2	UART0_IIR	UART 0 Interrupt Identification Register	01	R	91			
	UART0_FCTL	UART 0 FIFO Control Register	00	W	98			
C3	UART0_LCTL	UART 0 Line Control Register	00	R/W	99			
C4	UART0_MCTL	UART 0 Modem Control Register	00	R/W	101			
C5	UART0_LSR	UART 0 Line Status Register	60	R	102			
C6	UART0_MSR	UART 0 Modem Status Register	X0	R	104			
C7	UART0_SPR	UART 0 Scratch Pad Register	00	R/W	105			
C8	I2C0_SAR	I ² C 0 Slave Address Register	00	R/W	128			
C9	I2C0_xSAR	I ² C 0 Extended Slave Address Register	00	R/W	128			

Table 2. Register Map (Continued)

26
zilog

Address (hex)	Mnemonic	Name	Reset (hex)	CPU Access	Page No.
CA	I2C0_DR	I ² C 0 Data Register	00	R/W	129
СВ	I2C0_CTL	I ² C 0 Control Register	00	R/W	129
CC	I2C0_SR	I ² C 0 Status Register	F8	R	130
	I2C0_CCR	I ² C 0 Clock Control Register	00	W	131
CD	I2C0_SRR	I ² C 0 Software Reset Register	XX	W	132
CE	Not Accessible				
CF	UZI0_CTL	UZI 0 Control Register	00	R/W	64
D0	UART1_RBR	UART 1 Receive Buffer Register	XX	R	89
	UART1_THR	UART 1 Transmit Holding Register	XX	W	86
	BRG1_DLR_L	BRG 1 Divisor Latch Register—Low Byte	02	R/W	65
D1	BRG1_DLR_H	BRG 1 Divisor Latch Register—High Byte	00	R/W	65
	UART1_IER	UART 1 Interrupt Enable Register	00	R/W	90
D2	UART1_IIR	UART 1 Interrupt Identification Register	01	R	91
	UART1_FCTL	UART 1 FIFO Control Register	00	W	98
D3	UART1_LCTL	UART 1 Line Control Register	00	R/W	99
D4	UART1_MCTL	UART 1 Modem Control Register	00	R/W	101
D5	UART1_LSR	UART 1 Line Status Register	60	R/W	102
D6	UART1_MSR	UART 1 Modem Status Register	XX	R/W	104
D7	UART1_SPR	UART 1 Scratch Pad Register	00	R/W	105
D8	I2C1_SAR	I ² C 1 Slave Address Register	00	R/W	128
D9	I2C1_xSAR	I ² C 1 Extended Slave Address Register	00	R/W	128
DA	I2C1_DR	I ² C 1 Data Register	00	R/W	129
DB	I2C1_CTL	I ² C 1 Control Register	00	R/W	129
DC	I2C1_SR	I ² C 1 Status Register	F8	R	130
	I2C1_CCR	I ² C 1 Clock Control Register	00	W	131
DD	I2C1_SRR	I ² C 1 Software Reset Register	XX	W	132
DE	Not Accessible				

Table 2. Register Map (Continued)

zilog[°]

Address	•••••	News	Reset	CPU	Page
(hex)	Mnemonic	Name	(hex)	Access	No.
DF	UZI1_CTL	UZI 1 Control Register	00	R/W	64
Multiply-	Accumulator				
E0	MACC_xSTART	Multiply-Accumulator <i>x</i> Starting Address Register	00	R/W	134
E1	MACC_xEND	Multiply-Accumulator <i>x</i> Ending Address Register	00	R/W	135
E2	MACC_xRELOAD	Multiply-Accumulator x Reload Register	00	R/W	137
E3	MACC_LENGTH	Multiply-Accumulator Length Register	00	R/W	138
E4	MACC_ySTART	Multiply-Accumulator y Starting Address Register	00	R/W	142
E5	MACC_yEND	Multiply-Accumulator <i>y</i> Ending Address Register	00	R/W	143
E6	MACC_yRELOAD	Multiply-Accumulator y Reload Register	00	R/W	144
E7	MACC_CTL	Multiply-Accumulator Control Register	00	R/W	145
E8	MACC_AC0	Multiply-Accumulator Byte 0 Register	XX	R/W	145
E9	MACC_AC1	Multiply-Accumulator Byte 1 Register	XX	R/W	154
EA	MACC_AC2	Multiply-Accumulator Byte 2 Register	XX	R/W	155
EB	MACC_AC3	Multiply-Accumulator Byte 3 Register	XX	R/W	156
EC	MACC_AC4	Multiply-Accumulator Byte 4 Register	XX	R/W	156
ED	MACC_STAT	Multiply-Accumulator Status Register	XX	R/W	159
DMA Con	trollers				
EE	DMA0_SAR_L	DMA0 Source Address Register—Low Byte	XX	R/W	163
EF	DMA0_SAR_H	DMA0 Source Address Register—High Byte	XX	R/W	163
F0	DMA0_SAR_U	DMA0 Source Address Upper Byte Register	XX	R/W	163
F1	DMA0_DAR_L	DMA0 Destination Address Register—Low Byte	XX	R/W	163
F2	DMA0_DAR_H	DMA0 Destination Address Register— High Byte	XX	R/W	163

Table 2. Register Map (Continued)

28

zilog[°]

Address (hex)	Mnemonic	Name	Reset (hex)	CPU Access	Page No.
F3	DMA0_DAR_U	DMA0 Destination Address Upper Byte Register	XX	R/W	163
F4	DMA0_BC_L	DMA0 Byte Count Register—Low Byte	XX	R/W	164
F5	DMA0_BC_H	DMA0 Byte Count Register—High Byte	XX	R/W	164
F6	DMA0_CTL	DMA0 Control Register	00	R/W	164
F7	DMA1_SAR_L	DMA1 Source Address Register—Low Byte	XX	R/W	163
F8	DMA1_SAR_H	DMA1 Source Address Register—High Byte	XX	R/W	163
F9	DMA1_SAR_U	DMA1 Source Address Upper Byte Register	XX	R/W	163
FA	DMA1_DAR_L	DMA1 Destination Address Register—Low Byte	XX	R/W	163
FB	DMA1_DAR_H	DMA1 Destination Address Register— High Byte	XX	R/W	163
FC	DMA1_DAR_U	DMA1 Destination Address Upper Byte Register	XX	R/W	163
FD	DMA1_BC_L	DMA1 Byte Count Register—Low Byte	XX	R/W	164
FE	DMA1_BC_H	DMA1 Byte Count Register—High Byte	XX	R/W	164
FF	DMA1_CTL	DMA1 Control Register	00	R/W	164

Table 2. Register Map (Continued)

Notes:

1. After an external pin reset, the WDT Control register resets to 00h. After a WDT time-out reset, the WDT Control register resets to 20h.

2. When the CPU reads this register, the pin value of the port is read.





eZ80[®] CPU Core

eZ80 CPU Core Overview

The eZ80 CPU is the first 8-bit microprocessor to support 16 MB linear addressing. Each software module or task under a real-time executive or operating system can operate in $Z80^{\ensuremath{\mathbb{R}}}$ compatible (64 KB) mode or full 24-bit (16 MB) address mode.

The eZ80 CPU instruction set is a superset of the instruction sets for the Z80 and Z180 CPUs. Z80 and Z180 programs can be executed on eZ80 CPU with little or no modification.

eZ80 CPU Core Features

The key features of eZ80 CPU Core are as follows:

- Upward code-compatible from Z80 and Z180 products
- 24-bit linear address space
- Single-cycle instruction fetch
- Pipelined fetch, decode, and execute
- Dual Stack Pointers for ADL (24-bit) and Z80 (16-bit) memory modes
- 24-bit CPU registers and ALU
- Zilog Debug Interface (ZDI) support
- Nonmaskable Interrupt (NMI) + support for 128 vectored interrupts

For more information on eZ80 CPU, its instruction set, and eZ80 programming, refer to $eZ80^{\text{\ensuremath{\mathbb{R}}}}$ CPU User Manual (UM0077).

zilog

Programmable Reload Timers

Programmable Reload Timers Overview

The eZ80190 device features six Programmable Reload Timers (PRT). Each PRT contains a 16-bit downcounter and a 16-bit reload register. In addition, each PRT features a 4-bit clock prescaler with four selectable taps for CLK \div 2, CLK \div 4, CLK \div 8 and CLK \div 16. Each timer can be individually enabled to operate in either SINGLE PASS or CONTINU-OUS mode. The timer can be programmed to start, stop, restart from the current value, or restart from the initial value, and generate interrupts for the CPU.

Each of the 6 PRTs available on the eZ80190 device can be controlled individually. They do not share the same counters, reload registers, control registers, or interrupt signals. Figure 3 displays a simplified block diagram of a programmable reload timer.



Figure 3. Programmable Reload Timer Block Diagram



Programmable Reload Timer Operation

Setting Timer Duration

There are three factors to consider when determining Programmable Reload Timer duration—clock frequency, clock divider ratio, and initial count value. Minimum duration of the timer is achieved by loading 0001h, because the timer times out on the next clock edge. Maximum duration is achieved by loading 0000h, because the timer rolls over to FFFFh on the next clock edge and then continues counting down to 0000h.

The time-out period of the PRT is returned by the following equation:

PRT Time-Out Period = Clock Divider Ratio x Reload Value
System Clock Frequency

SINGLE PASS Mode

In SINGLE PASS mode, when the end-of-count value, 0000h, is reached, counting halts, the timer is disabled, and the PRT_EN bit resets to 0. To restart the timer, the CPU must reenable the timer by setting the PRT_EN bit to 1. Figure 4 displays an example of a PRT operating in SINGLE PASS mode. Timer register information is listed in Table 3 on page 33.



Figure 4. PRT SINGLE PASS Mode Operation Example



Table 3. PRT Single-Pass Mode Operation Example

Parameter	Control Register(s)	Value
PRT Enabled	TMRx_CTL[0]	1
Reload and Restart Enabled	TMRx_CTL[1]	1
PRT Clock Divider = 2	TMRx_CTL[3:2]	00b
Single-Pass Mode	TMRx_CTL[4]	0
PRT Interrupt Enabled	TMRx_CTL[6]	0
PRT Reload Value	{TMRx_RR_H, TMRx_RR_L}	0004h

CONTINUOUS Mode

In CONTINUOUS mode, when the end-of-count value, 0000h, is reached, the timer automatically reloads the 16-bit start value from the Timer Reload registers, TMRx_RR_H and TMRx_RR_L. Downcounting continues on the next clock edge. In CONTINUOUS mode, the PRT continues to count until disabled. Figure 5 displays an example of a PRT operating in CONTINUOUS mode. Timer register information is listed in Table 4.



Figure 5. PRT Continuous Mode Operation Example

zilog

Parameter	Control Register(s)	Value
PRT Enabled	TMRx_CTL[0]	1
Reload and Restart Enabled	TMRx_CTL[1]	1
PRT Clock Divider = 2	TMRx_CTL[3:2]	00b
Continuous Mode	TMRx_CTL[4]	1
PRT Interrupt Enabled	TMRx_CTL[6]	0
PRT Reload Value	{TMRx_RR_H, TMRx_RR_L}	0004h

Table 4. PRT Continuous Mode Operation Example

Reading the Current Count Value

The eZ80^{$\ensuremath{\mathbb{R}}$} CPU is capable of reading the current count value while the timer is running. This READ event does not affect timer operation.

Timer Interrupts

The timer interrupt flag, PRT_IRQ, is set to 1 whenever the timer reaches its end-of-count value, 0000h, in SINGLE PASS mode, or when the timer reloads the start value in CON-TINUOUS mode. The timer interrupt flag is only set when the timer reaches 0000h (or reloads) from 0001h. The timer interrupt flag is not set to 1 when the timer is loaded with the value 0000h, which selects the maximum time-out period.

The CPU can be programmed to poll the PRT_IRQ bit for the time-out event. Alternatively, an interrupt service request signal can be sent to the CPU by setting the IRQ_EN bit to 1. Then, when the end-of-count value, 0000h, is reached and the PRT_IRQ bit is set to 1, an interrupt service request signal is passed to the CPU. The PRT_IRQ bit is cleared to 0 and the interrupt service request signal is inactivated whenever the CPU reads from the timer control register, TMRx_CTL.

The response of the CPU to this interrupt service request is a function of the CPU's interrupt enable flag, IEF1. For more information, refer to $eZ80^{\ensuremath{\mathbb{R}}}$ CPU User Manual (UM0077).

Programmable Reload Timer Registers

Each programmable reload timer is controlled using five 8-bit registers. These registers are the TIMER*x* Control register, TIMER*x* Reload Low Byte register, TIMER*x* Reload High Byte register, TIMER*x* Data Low Byte register, and TIMER*x* Data High Byte register. The variable *x* can be 0, 1, 2, 3, 4, or 5, representing each of the six available timers.

zilog |₃₅

The Timer Control register can be read or written to. The timer reload registers are Write Only and are located at the same I/O address as the timer data registers, which are Read Only.

Timer Control Registers

The Timer Control registers, listed in Table 5, are used to control operation of the timer, including enabling the timer, selecting the clock divider, enabling the interrupt, selecting between CONTINUOUS and SINGLE PASS modes, and enabling the automatic reload feature.

Table 5. Timer Control Register	(TMR0_CTL = 0080h, TMR1	_CTL = 0083h, TMR2_CTL =
0086h, TMR3_CTL = 0	089h, TMR4_CTL = 008Ch, T	MR5_CTL = 008Fh)

Bit		7	6	5	4	3	2	1	0				
Reset		0	0	0	0	0	0	0	0				
CPU Access	5	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W				
Note: R = Re	ad Only; R	/W = Read	I/Write.										
Bit Position	Value	Descri	ption										
7 0 The timer has not reached its end-of-count value. The presence of								e. This bi read.	t is				
	1	The tim 1, an in the TM	iterrupt s Rx_CTL	eached if signal is s register	ts end-of sent to tl is read.	-count v ne CPU.	alue. If I This bit	RQ_EN remains	is set to 1 until				
6	0	Timer i	nterrupt	requests	are disa	abled.							
IRQ_EN	1	Timer i	Timer interrupt requests are enabled.										
5	0	Reserv	ed										
4 PRT_MODE	0	The tim reset to reache	The timer operates in SINGLE PASS mode. PRT_EN (bit 0) is reset to 0, and counting stops when the end-of-count value is reached.										
	1	The tim value is reache	The timer operates in CONTINUOUS mode. The timer reload value is written to the counter when the end-of-count value is reached.										
[3:2]	00	Clock ÷	2 is the	timer in	put sour	ce.							
CLK_DIV	01	Clock ÷	- 4 is the	timer in	put sour	ce.							
	10	Clock ÷	- 8 is the	timer in	put sour	ce.							
	11	Clock ÷	- 16 is th	e timer i	nput sou	irce.							

zilog

1 RST_EN	0	The automatic reload and restart function is disabled.
	1	The automatic reload and restart function is enabled. When a 1 is written to RST_EN, the values in the reload registers are loaded into the downcounter and the timer restarts.
0	0	The programmable reload timer is disabled.
PRT_EN	1	The programmable reload timer is enabled.

Timer Data Low Byte Register

This Read Only register returns the Low byte of the current count value of the selected timer. The Timer Data Low Byte register, listed in Table 6, can be read while the timer is in operation. Reading the current count value does not affect timer operation. To read the 16-bit data of the current count value, {TMRx_DR_H[7:0], TMRx_DR_L[7:0]}, first read the Timer Data Low Byte register and then read the Timer Data High Byte register. The Timer Data High Byte register value is latched when a read of the Timer Data Low Byte register value is latched when a read of the Timer Data Low Byte register occurs.

Note: *The timer data registers and timer reload registers share the same 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		
Note: R = Read Or	Note: R = Read Only.									
Bit Position	Value	Descript	ion							
[7:0] TMR_DR_L	00h–FFh	These bit value, {The of the 16- bit (Isb) o								

Table 6. Timer Data Low Byte Register (TMR0_DR_L = 0081h, TMR0_DR_L = 0084h, TMR0_DR_L = 0087h, TMR0_DR_L = 008Ah, TMR0_DR_L = 008Dh)

Timer Data High Byte Register

This Read Only register returns the High byte of the current count value of the selected timer. The Timer Data High Byte register, listed in Table 7 on page 37, can be read while the timer is in operation. Reading the current count value does not affect timer operation. To read the 16-bit data of the current count value, {TMRx_DR_H[7:0], TMPx_DP_L[7:0]}, first need the Timer Data Low Pate register need the Timer

TMRx_DR_L[7:0]}, first read the Timer Data Low Byte register and then read the Timer

36

Zilog¹₃

Data High Byte register. The Timer Data High Byte register value is latched when a read of the Timer Data Low Byte register occurs.

Note: The timer data registers and timer reload registers share the same address space.

Table 7. Timer Data High Byte Registers	(TMR0_DR	_H = 0082h,	TMR1_	DR_	_H = 0085h,
TMR2_DR_H = 0088h, TMR3_	DR_H = 008B	h, TMR4_DI	R_H = 0	08E	h)

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
Note: R = Read	Only.							
Bit Position	Value	Descript	ion					
[7:0] TMR_DR_H	00h–FFh	These bit value, {TI most-sigr 0 is bit 8	s represe MRx_DR hificant bi	ent the H _H[7:0], t (msb) c -bit time	ligh byte TMRx_[of the 16 r data va	of the 2- DR_L[7:0 -bit time lue.	-byte tim)]}. Bit 7 i r data va	er data is bit 15 Ilue. Bit

Timer Reload Low Byte Registers

The Timer Reload Low Byte registers, listed in Table 8, stores the most significant byte (MSB) of the 2-byte timer reload value. In CONTINUOUS mode, the timer reload value is reloaded into the timer upon end-of-count. When the RST_EN bit (TMRx_CTL[1]) is set to 1 to enable the automatic reload and restart function, the timer reload value is written to the timer on the next rising edge of the clock.

Note: *The timer data registers and timer reload registers share the same address space.*

Table 8. Timer Reload Low Byte Registers (TMR0_RR_L = 0081h, TMR1_RR_L = 0084h, TMR2_RR_L = 0087h, TMR3_RR_L = 008Ah, TMR4_RR_L = 008Dh)

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



Bit Position	Value	Description
[7:0] TMR_RR_L	00h–FFh	These bits represent the Low byte of the 2-byte timer reload value, {TMRx_RR_H[7:0], TMRx_RR_L[7:0]}. Bit 7 is bit 7 of the 16-bit timer reload value. Bit 0 is bit 0 (lsb) of the 16-bit timer reload value.

Timer Reload High Byte Registers

The Timer Reload High Byte registers, listed in Table 9, stores the MSB of the 2-byte timer reload value. In CONTINUOUS mode, the timer reload value is reloaded into the timer upon reaching an end-of-count. When the RST_EN bit (TMRx_CTL[1]) is set to 1 to enable the automatic reload and restart function, the timer reload value is written to the timer on the next rising edge of the clock.

Note: *The timer data registers and timer reload registers share the same address space.*

Table 9. Timer Reload High Byte Registers (TMR0_RR_H = 0082h, TMR1_RR_H = 0085h, TMR2_RR_H = 0088h, TMR3_RR_H = 008Bh, TMR4_RR_H = 008Eh)

D:4		<u> </u>	-		2	2	4	•		
Bit	1	6	5	4	3	2	1	U		
Reset	0	0	0	0	0	0	0	0		
CPU Access	W	/ W	W	W	W	W	W	W		
Note: W = Write	Only.									
Bit										
Position	Value	Descripti	on							
[7:0] TMR_RR_H	00h–FFh	These bits reload val is bit 15 (I of the 16-	These bits represent the High byte of the 2-byte timer eload value, {TMRx_RR_H[7:0], TMRx_RR_L[7:0]}. Bit 7 s bit 15 (msb) of the 16-bit timer reload value. Bit 0 is bit 8 of the 16-bit timer reload value							

38





Watchdog Timer

WDT Overview

The key features of eZ80190 device includes:

- Four programmable time-out periods: 2^{18} , 2^{22} , 2^{25} , and 2^{27} clock cycles
- A WDT time-out RESET indicator flag
- A selectable time-out response: a time-out generates a RESET or a nonmaskable interrupt

Figure 6 displays the block diagram for the WDT.



Figure 6. WDT Block Diagram



WDT Operation

Enabling And Disabling The WDT

The WDT is disabled upon a system RESET. To enable the WDT, the application program must set the WDT_EN bit (bit 7) of the WDT_CTL register. When enabled, the WDT cannot be disabled without a system RESET.

Time-Out Period Selection

There are four choices of time-out periods for the WDT—2¹⁸, 2²², 2²⁵, and 2²⁷ system clock cycles. With a 50 MHz crystal oscillator, the available WDT time-out periods are approximately 5.24 ms, 83.9 ms, 671 ms, and 2.68 s. The WDT time-out period is defined by the WDT_PERIOD field of the WDT_CTL register (WDT_CTL[1:0]).

RESET Or NMI Generation

Upon a WDT time-out, the RST_FLAG bit in the WDT_CTL register is set to 1. In addition, the WDT can cause a system RESET or send a nonmaskable interrupt (NMI) signal to the CPU. The default operation is for the WDT to cause a system RESET. The reset pulse generated by a WDT time-out is 64 clock cycles wide. It asserts/deasserts on the rising edge of the clock. The RST_FLAG bit can be polled by the CPU to determine the source of the RESET event.

If the NMI_OUT bit in the WDT_CTL register is set to 1, then upon time-out, the WDT asserts an NMI for CPU processing. The RST_FLAG bit can be polled by the CPU to determine the source of the NMI event.

WDT Registers

WDT Control Register

The WDT Control register, listed in Table 10 on page 41, is an 8-bit Read/Write register used to enable the WDT, set the time-out period, indicate the source of the most recent RESET, and select the required operation upon WDT time-out.



Table 10. WDT (Control	Registe	er	(WDT_CTL = 93h)								
Bit		7	6	5	4	3	2	1	0			
Reset		0	0	0/1	0	0	0	0	0			
CPU Access		R/W	R/W	R	R	R	R	R/W	R/W			
Note: R = Read C	Only; R/W	′ = Read	/Write.									
Bit Position	Value	Desci	ription									
7	0	WDT	WDT is disabled.									
WDT_EN	1	WDT is enabled. When enabled, the WDT cannot be disabled without a full-chip reset.										
6	0	WDT	time-out	resets th	ne CPU.							
NMI_OUT	1	WDT time-out generates a nonmaskable interrupt (NMI) to the CPU.										
5	0	RESET caused by external full-chip reset or ZDI reset.										
RST_FLAG	1	RESE time-c poll th	RESET caused by WDT time-out. This flag is set by the WDT time-out, even if the NMI_OUT flag is set to 1. The CPU can poll this bit to determine the source of the RESET or NMI.									
[4:2]	000	Reser	ved.									
[1:0]	00	WDT	time-out	period is	3 134,21	7,728 (2	²⁷) cloc	k cycles.				
WDT_PERIOD	01	WDT	time-out	period is	33,554	,432 (2 ²	⁵) clock	cycles.				
	10	WDT	time-out	period is	s 4,194,3	304 (2 ²²)) clock d	cycles.				
	11	WDT	time-out	period is	3 262,14	4 (2 ¹⁸) c	lock cy	cles.				

WDT Reset Register

The WDT Reset register, listed in Table 11, is an 8-bit Write Only register. The WDT is reset when an A5h value followed by 5Ah is written to this register. Any amount of time can occur between the writing of the A5h value and the 5Ah value, so long as the WDT time-out does not occur prior to completion.

Table 11. WDT Rese	(WDT_RR = 94h)							
Bit	7	6	5	4	3	2	1	0
Reset	Х	Х	Х	Х	Х	Х	Х	Х
CPU Access	W	W	W	W	W	W	W	W
Note: X = Undefined;	W = Write Or	ıly.						



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[®]

General-Purpose Input/Output

GPIO Overview

The eZ80190 device features 32 General-Purpose Input/Output (GPIO) pins. The GPIO pins are assembled as four 8-bit ports—Port A, Port B, Port C, and Port D. All port signals can be configured for use as either inputs or outputs. In addition, all of the port pins can be used as vectored interrupt sources for the eZ80[®] CPU.

GPIO Operation

GPIO operation is the same for all four GPIO ports (Ports A, B, C, and D). Each port features eight GPIO port pins. The operating mode for each pin is controlled by four bits that are divided between four 8-bit registers. These GPIO mode control registers are:

- Port *x* Data Register (Px_DR)
- Port *x* Data Direction Register (Px DDR)
- Port *x* Alternate Register 1 (Px_ALT1)
- Port *x* Alternate Register 2 (Px_ALT2)

where, *x* can be *A*, *B*, *C*, or *D*, representing any of the four GPIO ports A, B, C, or D. The mode for each pin is controlled by setting each register bit pertinent to the pin to be configured. For example, the operating mode for Port B Pin 7 (PB7) is set by the values contained in PB_DR[7], PB_DDR[7], PB_ALT1[7], and PB_ALT2[7].

The combination of the GPIO control register bits allows individual configuration of each port pin for nine modes. In all modes, reading the Port x Data register returns the sampled state, or level, of the signal on the corresponding pin. Table 12 lists the function of each port signal based upon these four register bits. After a RESET event, all GPIO port pins are configured as standard digital inputs, with interrupts disabled.

GPIO Mode	Px_ALT2 Bits7:0	Px_ALT1 Bits7:0	Px_DDR Bits7:0	Px_DR Bits7:0	Port Mode	Output
1	0	0	0	0	Output	0
	0	0	0	1	Output	1
2	0	0	1	0	Input from pin	High impedance
	0	0	1	1	Input from pin	High impedance

Table 12. GPIO Mode Selection

zilog[®]

GPIO Mode	Px_ALT2 Bits7:0	Px_ALT1 Bits7:0	Px_DDR Bits7:0	Px_DR Bits7:0	Port Mode	Output		
3	0	1	0	0	Open-Drain output	0		
	0	1	0	1	Open-Drain I/O	High impedance		
4	0	1	1	0	Open source I/O	High impedance		
	0	1	1	1	Open source output	1		
5	1	0	0	0	Reserved	High impedance		
6	1	0	0	1	Interrupt-dual edge triggered	High impedance		
7	1	0	1	0	0 Port A or B—input from pin, high-impedance of Port C or D—alternate function controls port I			
	1	0	1	1	Port A or B—input from pin, high-i Port C or D—alternate function co	om pin, high-impedance output. ate function controls port I/O.		
8	1	1	0	0	Interrupt—active Low	High impedance		
	1	1	0	1	Interrupt—active High	High impedance		
9	1	1	1	0	Interrupt—falling edge triggered	High impedance		
	1	1	1	1	Interrupt—rising edge triggered	High impedance		

Table 12. GPIO Mode Selection (Continued)

GPIO Mode 1—The port pin is configured as a standard digital output pin. The value written to the Port x Data register (Px DR) is presented on the pin.

GPIO Mode 2—The port pin is configured as a standard digital input pin. The output is tristated (high impedance). The value stored in the Port *x* Data register produces no effect. As in all modes, a read from the Port *x* Data register returns the pin's value. GPIO Mode 2 is the default operating mode following a RESET.

GPIO Mode 3—The port pin is configured as open-drain I/O. The GPIO pins do not feature an internal pull-up to the supply voltage. To employ the GPIO pin in open-drain mode, an external pull-up resistor must connect the pin to the supply voltage. Writing a 0 to the Port x Data register outputs a Low at the pin. Writing a 1 to the Port x Data register results in high-impedance output.

GPIO Mode 4—The port pin is configured as open-source I/O. The GPIO pins do not feature an internal pull-down to the supply ground. To employ the GPIO pin in open-source mode, an external pull-down resistor must connect the pin to the supply ground. Writing a 1 to the Port *x* Data register outputs a High at the pin. Writing a 0 to the Port *x* Data register results in a high-impedance output.

GPIO Mode 5—Reserved. This pin produces high-impedance output.

zilog

GPIO Mode 6—The bit enables a dual-edge-triggered interrupt mode. Both a rising and a falling edge on the pin cause an interrupt request to be sent to the CPU. Writing a 1 to the Port *x* Data register bit position resets the corresponding interrupt request. Writing a 0 produces no effect. The programmer must set the Port *x* Data register before entering the dual-edge-triggered interrupt mode.

GPIO Mode 7—For Ports C and D, the port pin is configured to pass control over to the alternate functions assigned to the pin. For example, the alternate mode function for PC7 is $\overline{\text{RI1}}$. When GPIO Mode 7 is enabled, the pin output data and pin tristate control come from the alternate function's data output and tristate control, respectively. The value in the Port *x* Data register produces no effect on operation.

For Ports A and B, which do not feature alternate I/O functions, selecting GPIO Mode 7 results in a configuration of the pins for input from the pin and high-impedance output as in GPIO Mode 2.

GPIO Mode 8—The port pin is configured for level-sensitive interrupt modes. An interrupt request is generated when the level at the pin is the same as the level stored in the Port *x* Data register. The port pin value is sampled by the system clock. The input pin must be held at the selected interrupt level for a minimum of 2 clock periods to initiate an interrupt. The interrupt request remains active as long as this condition is maintained at the external source.

GPIO Mode 9—The port pin is configured for single-edge-triggered interrupt mode. The value in the Port x Data register determines if a positive or negative edge causes an interrupt request. A 0 in the Port x Data register bit sets the selected pin to generate an interrupt request for falling edges. A 1 in the Port x Data register bit sets the selected pin to generate an interrupt request for rising edges. The interrupt request remains active until a 1 is written to the Port x Data register bit's corresponding interrupt request. Writing a 0 produces no effect on operation. The programmer must set the Port x Data register before entering the single-edge-triggered interrupt mode.

A simplified block diagram of a GPIO port pin is displayed in Figure 7 on page 46.





Figure 7. GPIO Port Pin Block Diagram

GPIO Interrupts

Each port pin can be used as an interrupt source. Interrupts can be either level- or edge-triggered.

Level-Triggered Interrupts

When the port is configured for level-triggered interrupts, the corresponding port pin is tristated. An interrupt request is generated when the level at the pin is the same as the level stored in the Port x Data register. The port pin value is sampled by the system clock. The input pin must be held at the selected interrupt level for a minimum of 2 clock periods to initiate an interrupt. The interrupt request remains active as long as this condition is maintained at the external source.

For example, if PA3 is programmed for low-level interrupt and the pin is forced Low for 2 clock cycles, an interrupt request signal is generated from that port pin and sent to the CPU. The interrupt request signal remains active until the external device driving PA3

zilog ₄₇

forces the pin High. The CPU must be enabled to respond to interrupts for the interrupt request signal to be acted upon.

Edge-Triggered Interrupts

When the port is configured for edge-triggered interrupts, the corresponding port pin is tristated. If the pin receives the correct edge from an external device, the port pin generates an interrupt request signal to the CPU. Any time a port pin is configured for edge-triggered interrupt, writing a 1 to that pin's Port *x* Data register causes a reset of the edge-triggered interrupt. The programmer must set the bit in the Port *x* Data register to 1 before entering either single- or dual-edge-triggered interrupt mode for that port pin.

When configured for dual-edge-triggered interrupt mode (GPIO Mode 6), both a rising and a falling edge on the pin cause an interrupt request to be sent to the CPU.

When configured for single-edge-triggered interrupt mode (GPIO Mode 9), the value in the Port x Data register determines if a positive or negative edge causes an interrupt request. A 0 in the Port x Data register bit sets the selected pin to generate an interrupt request for falling edges. A 1 in the Port x Data register bit sets the selected pin to generate in interrupt request for rising edges.

GPIO Control Registers

The 16 GPIO Control Registers operate in groups of four with a set for each Port (A, B, C, and D). Each GPIO port features a Port Data register, Port Data Direction register, Port Alternate register 1, and Port Alternate register 2.

Port x Data Registers

When the port pins are configured for one of the output modes, the data written to the Port x Data registers, listed in Table 13 on page 48, are driven on the corresponding pins. In all modes, reading from the Port x Data registers always returns the current sampled value of the corresponding pins. When the port pins are configured as edge-triggered interrupt sources, writing a 1 to the corresponding bit in the Port x Data register clears the interrupt signal that is sent to the CPU. When the port pins are configured for edge-selectable interrupts or level-sensitive interrupts, the value written to the Port x Data register bit selects the interrupt edge or interrupt level. For more information see Table 12 on page 43.

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

Table 16. Port x Alternate Registers 2 (PA_ALT2 = 99h, PB_ALT2 = 9Dh, PC_ALT2 = A1h, PD_ALT2 = A5h)

		-	_					-
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.								

PS006614-1208

49



50

Chip Selects and Wait States

The eZ80190 device generates four Chip Selects for external devices. Each Chip Select may be programmed to access either memory space or I/O space. The Memory Chip Selects can be individually programmed on a 64 KB boundary. The I/O Chip Selects can each choose a 16-byte section of I/O space. In addition, each Chip Select may be programmed for up to 7 WAIT states.

Memory and I/O Chip Selects

Each of the four available Chip Selects can be enabled for either the memory address space or the I/O address space, but not both. To select the memory address space for a particular Chip Select, CS_IO (CSx_CTL[4]) must be reset to 0. To select the I/O address space for a particular Chip Select, CS_IO must be set to 1. After RESET, the default is for all Chip Selects to be configured for the memory address space. For either the memory address space or the I/O address space, the individual Chip Selects must be enabled by setting CS_EN (CSx_CTL[3]) to 1.

Memory Chip Select Operation

Each of the four Memory Chip Selects features three control registers. To enable a particular Memory Chip Select, the following conditions must be met:

- The Chip Select is enabled by setting CS EN to 1
- The Chip Select is configured for memory by clearing CS IO to 0
- The address is in the associated Chip Select range:

 $CSx_LBR[7:0] \le ADDR[23:16] \le CSx_UBR[7:0]$

- No higher priority (lower number) Chip Select meets the above three conditions
- No on-chip RAM is configured for the same address space, because on-chip RAM is prioritized higher than all Memory Chip Selects
- A memory access instruction must be executing

If all of the foregoing conditions are met to generate a Memory Chip Select, then the following actions occur:

- A Chip Select— $\overline{CS0}$, $\overline{CS1}$, $\overline{CS2}$, or $\overline{CS3}$ —is activated (driven Low)
- The MREQ signal is activated (driven Low)

Product Specification

eZ80190

- Depending upon the instruction, either \overline{RD} or \overline{WR} are activated (driven Low)

If the upper and lower bounds are set to the same value, such that $CSx_UBR = CSx_LBR$, then a particular Chip Select is valid for, at most, a single 64 KB page. Again, if a higher-priority Chip Select also encompasses the same set of addresses, then the lower-priority Chip Select is not generated.

Memory Chip Select Priority

A lower-numbered Chip Select is granted priority over a higher-numbered Chip Select. If the Chip Select 0 address space overlaps the Chip Select 1 address space, Chip Select 0 is active. If the address range programmed for any Chip Select signal overlaps with the address of internal RAM, then RAM is accorded higher priority. If the particular Chip Select(s) feature an address range that overlaps with the RAM address, then when RAM is selected, the Chip Select signal is not asserted.

Reset States

On reset, Chip Select 0 is active for all addresses, because its Lower Bound register resets to 0000h and its Upper Bound register resets to FFFFh. All of the other Chip Select Lower and Upper Bound registers reset to 0000h.

Memory Chip Select Example

The use of Memory Chip Selects is displayed in Figure 8 on page 52. The associated control register values listed in Table 17 on page 52. In this example, all 4 Chip Selects are enabled and configured for memory addresses. CS0, CS1, and CS2 are all configured such that their address spaces do not overlap. CS3 is allocated an address space that spans the entire 16 MB of available memory. Consequently, CS3 overlaps the address spaces for CS0, CS1, and CS2. However, because CS3 is the lowest-priority Chip Select, it only becomes active where it does not overlap either CS0, CS1, or CS2.

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

zilog

53	
00	

Chip Select	CS <i>x</i> _CTL[3] CS <i>x</i> _EN	CSx_CTL[4] CSx_IO	CSx_LBR	CSx_UBR	Description
CS2	1	0	A0h	CFh	CS2 is enabled as a Memory Chip Select. Valid addresses range from A00000h to CFFFFFh.
CS3	1	0	D0h	FFh	CS3 is enabled as a Memory Chip Select. Valid addresses range from D00000h to FFFFFh.

Table 17. Register Values for Memory Chip Select Example (Continued)

I/O Chip Select Operation

I/O Chip Selects can only be active when the CPU is performing I/O instructions. Because the I/O space is separate from the memory space in the eZ80190 device, there can never be a conflict between I/O and memory addresses.

The I/O Chip Select logic decodes 8 bits from the address bus, ADDR[11:4]. Because the upper byte of the address bus, ADDR[23:16], is ignored, the I/O devices can always be accessed from within any memory mode (ADL or Z80). The MBASE offset value used for setting the Z80 MEMORY mode page is also always ignored.

Four I/O Chip Selects are available with the eZ80190 device. To generate a particular I/O Chip Select, the following conditions must be met:

- The Chip Select is enabled by setting CS_EN to 1
- The Chip Select is configured for I/O by setting CS_IO to 1
- An I/O Chip Select address match occurs—ADDR[11:4] = CSx_LBR[7:0]
- No higher-priority (lower-number) Chip Select meets the above conditions
- The lower byte of the I/O address is not within the on-chip peripheral address range of 80h to FFh. On-chip peripheral registers assume priority for all addresses where 80h ≤ ADDR[7:0] ≤ FFh
- An I/O instruction must be executing

If all of the foregoing conditions are met to generate an I/O Chip Select, then the following actions occur:

- A Chip Select— $\overline{CS0}$, $\overline{CS1}$, $\overline{CS2}$, or $\overline{CS3}$ —is activated (driven Low)
- The IORQ signal is activated (driven Low)



• Depending upon the instruction, either \overline{RD} or \overline{WR} are activated (driven Low)

I/O Chip Select Precaution

For all I/O operations, the upper byte of the address bus, ADDR[23:16], is forced to 00h. The I/O Chip Selects do not compare the values stored in the CSx_LBR registers to what is generally considered to be the High byte of the I/O address, or ADDR[15:8]. Instead, the I/O Chip Selects compare the values stored in the CSx_LBR registers to a byte taken from the middle of the I/O address, or ADDR[11:4].

Wait States

For each of the four available Chip Selects, programmable WAIT states can be inserted to provide external devices with additional clock cycles to complete their read and write operations. The number of WAIT states for a particular Chip Select is controlled by the 3-bit field CSx_wait ($CSx_CTL[7:5]$). The Chip Selects can be independently programmed to provide 0 to 7 WAIT states. The WAIT states idle the CPU for the specified number of system clock cycles. An example of WAIT state operation is displayed in Figure 9 on page 55. In this example, a single WAIT state is added. It causes the instruction read operation to use an additional clock cycle. WAIT is an internal signal used by the eZ80190 device. See Figure 9 on page 55.









Chip Select Registers

Chip Select x Lower Bound Register

For Memory Chip Selects, the Chip Select x Lower Bound register, listed in Table 18 on page 56, defines the lower bound of the address range for which the corresponding Memory Chip Select, if enabled, can be active. For I/O Chip Selects, this register defines the address to which ADDR[11:4] is compared to generate an I/O Chip Select. All Chip Select lower bound registers reset to 00h.

zilog

56

Table 18. Chip Select x Lower Bound Register (CS0_LBR = A8h, CS1_LBR = ABh, CS2_LBR = AEh, CS3_LBR = B1h)

Bit		7	6	5	4	3	2	1	0
Reset		0	0	0	0	0	0	0	0
CPU Acces	S	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Note: R/W =	Read/Write	е.							
Bit Position	Value	Descript	ion						
[7:0] 00h– For Memory Chip Selects (CS_io = 0) CS_LBR FFh This bit specifies the lower bound of the Chip Select range. The upper byte of the address bus, ADDR[2: compared to the values contained in these registers determining if a Memory Chip Select signal should I						ect addre [23:16], i ers for d be gen	ess s erated		
		For I/O C This bit s compare determin	Chip Sel pecifies d to the ing if an	ects (C: the Chip values c I/O Chip	S_io = 1 Select contained Select) address d in thes signal st	value. A e registe nould be	ADDR[11 ers for generat	:4] is ed.

Chip Select x Upper Bound Register

For Memory Chip Selects, the Chip Select x Upper Bound register, listed in Table 19, defines the upper bound of the address range for which the corresponding Chip Select (if enabled) can be active. For I/O Chip Selects, this register produces no effect. The reset state for the Chip Select 0 Upper Bound register is FFh, while the reset state for the 3 other Chip Select upper bound registers is 00h.

Table 19. Chip Select x Upper Bound Register(CS0_UBR = A9h, CS1_UBR = ACh,CS2_UBR = AFh, CS3_UBR = B2h)

Bit	7	6	5	4	3	2	1	0
CS0 Reset	1	1	1	1	1	1	1	1
CS1 Reset	0	0	0	0	0	0	0	0
CS2 Reset	0	0	0	0	0	0	0	0
CS3 Reset	0	0	0	0	0	0	0	0
CPU Access	R/W							
Note: R/W = Read/Write.								

Zilog | 57

Bit Position	Value	Description
[7:0] CS_UBR	00h– FFh	This bit specifies the upper bound of the Chip Select address range. The upper byte of the address bus, ADDR[23:16], is compared to the values contained in these registers for determining if a Chip Select signal should be generated.

Chip Select x Control Register

The Chip Select *x* Control register, listed in Table 20, enables the Chip Selects, specifies the type of Chip Select, and sets the number of WAIT states. The reset state for the Chip Select 0 Control register is E8h, while the reset state for the 3 other Chip Select control registers is 00h.

Table 20. Chip Select x Control Register (CS0_CTL = AAh, CS1_CTL = ADh, CS2_CTL = B0h, CS3_CTL = B3h)

Bit	7	6	5	4	3	2	1	0
CS 0 Reset	1	1	1	0	1	0	0	0
CS 1 Reset	0	0	0	0	0	0	0	0
CS 2 Reset	0	0	0	0	0	0	0	0
CS 3 Reset	0	0	0	0	0	0	0	0
CPU Access	R/W							

Note: R/W = Read/Write.

D 14

Bit Position	Value	Description
[7:5]	000	0 WAIT states are inserted when this Chip Select is active.
CS_WAIT	001	1 WAIT state is inserted when this Chip Select is active.
	010	2 WAIT states are inserted when this Chip Select is active.
	011	3 WAIT states are inserted when this Chip Select is active.
	100	4 WAIT states are inserted when this Chip Select is active.
	101	5 WAIT states are inserted when this Chip Select is active.
	110	6 WAIT states are inserted when this Chip Select is active.
	111	7 WAIT states are inserted when this Chip Select is active.
4	0	An address match results in a Memory Chip Select.
CS_IO	1	An address match results in an I/O Chip Select.



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

Random Access Memory

The eZ80190 device features an 8 KBx8 single-port data Random Access Memory (RAM) for general-purpose use and a 1 KB x 8 dual-port static RAM for use with the Multiply-Accumulator unit. Both RAM spaces can be individually enabled or disabled, and can be relocated to the top of any 64KB page in memory. Data is passed to and from the two RAM spaces via the 8-bit data bus, DATA[7:0]. The dual-port MACC RAM can be used with the Multiply-Accumulator or as additional general-purpose RAM, if required. For details about using the MACC RAM with the Multiply-Accumulator, see MACC RAM on page 125.

The general-purpose data RAM occupies the memory addresses range {RAM_ADDR_U[7:0], E000h} to {RAM_ADDR_U[7:0], FFFFh}. The Multiply-Accumulator dual-port RAM occupies the address range {RAM_ADDR_U[7:0], DC00h} to {RAM_ADDR_U[7:0], DFFFh}. An example of the memory mapping for the two on-chip RAM spaces is displayed in Figure 10. In this example, the RAM Address Upper Byte register, RAM_ADDR_U, is set to 7Ah. Figure 10 is not drawn to scale, as RAM memories occupy only a very small fraction of the available 16 MB address space.



Figure 10. On-Chip RAM Memory Addressing Example

When enabled, on-chip RAM assumes priority over all Memory Chip Selects that may also be enabled in the same address space. If an address is generated in a range that is covered by both the RAM address space and a particular Memory Chip Select address space, the Memory Chip Select is not activated.





RAM Control Registers

RAM Control Register

The internal RAM spaces, data RAM and Multiply-Accumulator RAM, can be enabled by setting corresponding bits in this register.

Table 21. RAM C	Control R	Registe	r								
Bit		7	6	5	4	3	2	1	0		
Reset		0	0	0	0	0	0	0	0		
CPU Access		R/W	R/W	R	R	R	R	R	R		
Note: R/W = Read	d/Write.										
Bit Position	Value	Desc	ription								
7	0	On-chip general-purpose RAM is disabled.									
GPRAM_EN	1	On-c	On-chip general-purpose RAM is enabled.								
6	0	On-c	hip multi	ply-accı	umulator	RAM is	disable	d.			
MACRAM_EN	1	On-c	hip multi	ply-accı	umulator	RAM is	enabled	1.			
[5:0]	000000	Rese	erved								

RAM Address Upper Byte Register

The RAM_ADDR_U register defines the upper byte of the address for the two on-chip RAM blocks—general-purpose RAM and MACC RAM. If either or both of these on-chip RAM blocks are enabled, their addresses assume priority over any Chip Selects. The external Chip Select signals are not asserted if the corresponding RAM address is enabled.

Table 22. RAM Address	ble 22. RAM Address Upper Byte Register							(RAM_ADDR_U=B5h)					
Bit	7	6	5	4	3	2	1	0					
Reset	0	0	0	0	0	0	0	0					
CPU Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W					
Note: R/W = Read/Write.													

Zilog ₆₁

Bit Position	Value	Description
[7:0] RAM_ADDR_U	00h– FFh	These bits define the upper byte of the RAM address. When on-chip general-purpose RAM is enabled, the general-purpose RAM address space ranges from {RAM_ADDR_U, E000h} to {RAM_ADDR_U, FFFh}. When on-chip MACC RAM is enabled, the MACC RAM address space ranges from {RAM_ADDR_U, DC00h} to {RAM_ADDR_U, DFFFh}. On-chip RAM is prioritized higher than all other Memory Chip Selects. If the enabled RAM and Chip Select addresses overlap, the external Chip Select is not asserted.

zilog

62

Universal Zilog Interface

The eZ80190 device features two on-chip Universal Zilog Interface (UZI) devices. Each UZI contains three serial communication controller blocks: a Serial Peripheral Interface (SPI), a Universal Asynchronous Receiver/Transmitter (UART), and an Inter-Integrated Circuit serial bus (I²C). For each UZI device, any one of these three communication controllers can be enabled. The UZI devices are connected to GPIO pins on Port C (UZI 1) and Port D (UZI 0). Ports C and D must be configured for alternate-function I/O to allow the UZI devices to communicate via the Port C and D pins. Each UZI also contains control registers and a Baud Rate Generator (BRG) that generates a lower-frequency clock for the SPI and UART devices. The I²C block features its own clock generator. The entire UZI block, including the Baud Rate Generator, is inactive when none of the serial devices are selected. Figure 11 displays the UZI block diagram.



Figure 11. UZI Block Diagram

See Serial Peripheral Interface on page 84, Universal Asynchronous Receiver/Transmitter on page 67, and I2C Serial I/O Interface on page 92 chapters for detailed operating infor-


mation. A description of the UZI Baud Rate Generator and the UZI control registers appear in this chapter.

Baud Rate Generator

The Baud Rate Generator (BRG) is located within the UZI, but outside the three serial communication controllers. The Baud Rate Generator creates a lower frequency clock from the high-frequency system clock provided as an input to each UZI. Baud Rate Generator output is used as the clock source by the SPI and the UART. The I²C device generates its timing directly from the primary system clock.

Baud Rate Generator Functional Description

The Baud Rate Generator consists of a 16-bit downcounter, two registers, and associated decoding logic. The Baud Rate Generator's initial value is defined by the two BRG Divisor Latch registers, {BRGx_DLR_H, BRGx_DLR_L}. At the rising edge of each system clock, the BRG decrements until it reaches the value 0001h. On the next system clock rising edge, the BRG reloads the initial value from {BRGx_DLR_H, BRGx_DLR_L} and outputs a pulse to indicate the end-of-count. Calculate the BRG output frequency with the following equation:

BRGx Output Frequency = System Clock Frequency {BRGx_DLR_H, BRGx_DLR_L}

Upon RESET, the 16-bit BRG divisor value resets to 0002h. A minimum BRG divisor value of 0001h is also valid, and effectively bypasses the BRG. A software Write to either the Low- or High-byte registers for the BRG Divisor Latch causes both the Low and High bytes to load into the BRG counter, and causes the count to restart.

The divisor registers can only be accessed if bit 7 of the UART Line Control register $(UARTx_LCTL)$ is set to 1. After reset, this bit is reset to 0.

Recommended Usage of the Baud Rate Generator

The following is the normal sequence of operations that should occur after the eZ80190 device is powered on to configure the UZI Baud Rate Generator:

- Assert and deassert RESET
- Set UARTx_LCTL[7] to 1 to enable access of the BRG divisor registers
- Program the BRGx_DLR_L and BRGx_DLR_H registers



- zilog |₆
- Clear UARTx LCTL[7] to 0 to disable access of the BRG divisor registers.

UZI and BRG Control Registers

UZI Control Registers

The UZI Control registers select between the three available serial communication controllers: I²C, SPI and UART. Each of the two UZI devices on the eZ80190 device features its own UZI Control register.

Table 23. UZI (Table 23. UZI Control Registers				UZI0_CI	L=CFh	, UZI1_	CTL=DF	^r h)
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/W R/						R/W	
Note: R = Read Only; R/W = Read/Write.									
Bit Position	Value	Descr	iption						
[7:2]	000000	Reser	ved						
[1:0]	00	All UZ	I devices	s are dis	abled.				
UZI_MODE	01	UART is enabled.							
	10	SPI is enabled.							
	11	I ² C is	I ² C is enabled.						

BRG Divisor Latch Registers—Low Byte

This register holds the Low byte of the 16-bit divisor count loaded by the processor for baud rate generation. The 16-bit clock divisor value is returned by {BRGx_DLR_H, BRGx_DLR_L}, where x is either 0 or 1 to identify the two available UZI devices. Upon RESET, the 16-bit BRG divisor value resets to 0002h. The initial 16-bit divisor value must be between 0002h and FFFFh as the values 0000h and 0001h are invalid and proper operation is not guaranteed. Thus the minimum BRG clock divisor ratio is 2.

A write to either the Low or High byte registers for the BRG Divisor Latch causes both bytes to be loaded into the BRG counter and the count restarted.

Bit 7 of the associated UART Line Control register (UARTx_LCTL) must be set to 1 to access this register for each UZI device. For more information see UART Line Control Register on page 77 (UARTx_LCTL).

zilog[®]

The BRGx_DLR_L registers share the same address space with the UARTx_RBR and UARTx_THR registers. Bit 7 of the associated UART Line Control register (UARTx_LCTL) must be set to 1 to enable access for this register within each UZI device.

Table 24. BRG Divisor Latch Registers—Low Byte BRG0_DLR_L = C0h, BRG1_DLR_L = D0h

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

Bit Position	Value	Description
[7:0] BRG_DLR_L	00h– FFh	These bits represent the Low byte of the 16-bit Baud Rate Generator divider value. The complete BRG divisor value is returned by {BRG_DLR_H, BRG_DLR_L}.

BRG Divisor Latch Registers—High Byte

This register holds the High byte of the 16-bit divisor count loaded by the processor for baud rate generation. The 16-bit clock divisor value is returned by $\{BRGx_DLR_H, BRGx_DLR_L\}$ where *x* is either 0 or 1 to identify the two available UZI devices. Upon RESET, the 16-bit BRG divisor value resets to 0002h. The initial 16-bit divisor value must be between 0002h and FFFFh because the values 0000h and 0001h are invalid, and proper operation is not guaranteed. Therefore, the minimum BRG clock divisor ratio is 2.

A write to either the Low- or High-byte registers for the BRG Divisor Latch causes both bytes to load into the BRG counter, and causes the count to restart.

Bit 7 of the associated UART Line Control register (UART*x*_LCTL) must be set to 1 to access this register for each UZI device. For more information see UART Line Control Register on page 77 (UART*x*_LCTL).

The BRGx_DLR_H registers share the same address space with the UARTx_IER registers. Bit 7 of the associated UART Line Control register (UARTx_LCTL) must be set to 1 to enable access for this register within each UZI device.

Table 25. BRG Divisor Latch Registers—High Byte (BRG0_DLR_H = C1h, BRG1_DLR_H = D1h)

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



Bit		
Position	Value	Description
[7:0] BRG_DLR_H	00h– FFh	These bits represent the High byte of the 16-bit Baud Rate Generator divider value. The complete BRG divisor value is returned by {BRG_DLR_H, BRG_DLR_L}.

zilog

Universal Asynchronous Receiver/ Transmitter

The UART module implements all of the logic required to support asynchronous communications protocol. The module also implements two separate 16-byte FIFOs for both transmit and receive. A block diagram of the UART is displayed in Figure 12.



Figure 12. UART Block Diagram

The UART module provides the following asynchronous communications protocol related features/functions:

- 5, 6, 7 or 8-bit data transmission
- Even/odd or no parity bit generation and detection
- Start and stop bit generation and detection (supports up to two stop bits)
- Line break detection and generation
- Receiver overrun and framing error detection
- Logic and associated I/O to provide modem hand-shake capability



UART Functional Description

The core uses an externally-provided clock from the Baud Rate Generator for the serial transmit/receive function. The UART module supports all of the various options in the asynchronous transmission and reception protocol including:

- 5 to 8-bit transmit/receive
- Start bit generation and detection
- Parity generation and detection
- Stop bit generation and detection
- Break generation and detection

The UART contains 16-byte FIFOs in each direction. The FIFOs can be enabled or disabled by the application. The receive FIFO features trigger-level detection logic, which enables the processor to block transfer data bytes from the receive FIFO.

The UART data transfer rate is calculated in the following equation:

UART Data Transfer Rate (bits/s) =

System Clock Frequency

16 x Baud Rate Generator Divisor

UART Functions

The UART function implements:

- The transmitter and associated control logic
- The receiver and associated control logic
- The modem interface and associated logic

UART Transmitter

The transmitter block controls the data transmitted on the TXD output. It implements the FIFO, accessed through the UARTx_THR register, the transmit shift register, the parity generator, and control logic for the transmitter to control parameters for the asynchronous communications protocol.

The UARTx_THR is a Write Only register. The processor writes the data byte to be transmitted into this register. In FIFO mode, up to 16 data bytes can be written through the UARTx_THR register. The data byte from the FIFO is transferred to the transmit shift register and transmitted on the TXD pin. After SYNC_RESET, the UARTx_THR register is empty so the Transmit Holding Register Empty (THRE) bit (bit 5 of the UARTx_LSR reg-

zilog

ister) is set to 1 and an interrupt is sent to the processor (if interrupts are enabled). The processor can reset this interrupt by loading data into the UARTx_THR register, which clears the transmitter interrupt.

The transmit shift register places the byte to be transmitted on the TXD signal serially. The lsb of the byte to be transmitted is shifted out first and the msb is shifted out last. The control logic within the block adds the asynchronous communications protocol bits to the data byte being transmitted. The transmitter block obtains the parameters for the protocol from the bits programmed through the UARTx_LCTL register. The TXD output is set to 1 if the transmitter is idle (it does not contain any data to be transmitted).

The transmitter operates with the Baud Rate Generator (BRG) clock. The data bits are placed on the TXD output one time every 16 BRG clock cycles. The transmitter block also implements a parity generator and attaches the parity bit with the byte if programmed to do so.

UART Receiver

The receiver block controls data reception from the RXD signal. The receiver block implements a receiver shift register, receiver line error condition monitoring logic and receiver data ready logic. It also implements a parity checker.

The processor reads received data from UARTx_RBR, which is a Read Only register. The condition of the UARTx_RBR register is monitored by the DR bit (bit 0 of the UARTx_LSR register). The DR bit is set to 1 when a data byte is received and transferred to the UARTx_RBR register from the receiver shift register. The DR bit is reset only when the processor reads all received data bytes. If the number of bits received is less than eight, the unused msbs of the data byte read are reset to 0.

The receiver uses the clock from the BRG input of the UZI for receiving data. This clock must be 16 times the required baud rate. The receiver synchronizes the shift clock on the falling edge of the RXD input start bit. It then receives a complete byte according to the set parameters. The receiver also implements logic to detect framing errors, parity errors, overrun errors, and break signals.

UART Modem Control

The modem control logic provides two outputs and four inputs for handshaking with the modem. Any change in the modem status inputs, except \overline{RI} , is detected. An interrupt can then be generated. For \overline{RI} , an interrupt is generated only when the trailing edge of the \overline{RI} is detected. The module also provides a loop mode for self-diagnostic purposes.



70

UART Interrupts

There are five different sources of interrupts from the UART. These five sources of interrupts are:

- 1. Transmitter
- 2. Receiver (three different interrupts)
- 3. Modem status

UART Transmitter Interrupt

The transmitter interrupt is generated if there is no data available for transmission. This interrupt can be disabled using the individual interrupt enable bit, or cleared by writing data into the UARTx_THR register.

UART Receiver Interrupts

A receiver interrupt can be generated by three possible events. The first event, a receiver data ready interrupt event, indicates that one or more data bytes were received and are ready to be read. If the FIFO is enabled, and the trigger level is set, then this interrupt is generated if the number of bytes in the receive FIFO is greater than or equal to the trigger level. If the FIFO is not enabled, the interrupt is generated if the receive buffer contains a data byte. This interrupt is cleared by reading the UARTx_RBR.

The second interrupt source is the receiver time-out. A receiver time-out interrupt is generated when there are fewer data bytes in the receive FIFO than the trigger level. There are no READs and writes to or from the receive FIFO for four consecutive byte times. After the receiver time-out interrupt is generated, it is cleared only after it empties the entire receive FIFO.

The first two interrupt sources from the receiver (data ready and time-out) share an interrupt enable bit.

The third source of a receiver interrupt is a line status error indicating an error in byte reception. This error may result from:

- Incorrect received parity
- Incorrect framing (the stop bit is not detected by the receiver at the end of the byte)
- Receiver overrun condition
- A Break Indication being detected on the receive data input

An interrupt due to one of the above conditions is cleared when the UARTx_LSR register is read. In the case of FIFO mode, a line status interrupt is generated only after the received byte with an error reaches the top of the FIFO and is ready to be read.

zilog[°]

A line status interrupt is activated (provided this interrupt is enabled) as long as the read pointer of the receive FIFO points to the location of the FIFO that contains a byte with the error. The interrupt is immediately cleared when the UARTx_LSR register is read. The ERR bit of the UARTx_LSR register is active as long as an error byte is present in the receive FIFO.

UART Modem Status Interrupt

The modem status interrupt is generated if there is any change in state of the modem status inputs to the UART. This interrupt is cleared when the processor reads the UARTx_MSR register.

UART Recommended Usage

The following is the standard sequence of events that occurs in the eZ80190 device using the UART. A description of each follows.

- Module reset
- Control transfers to configure UART operation
- Data transfers

Module Reset

Upon reset, all internal registers return to their default values. All command status registers are programmed with their default values and the FIFOs are flushed.

Control Transfers

Based on the application requirement, the data transfer baud rate is determined and the BRG is configured to generate a 16X clock frequency, provided at the BRG signal input. Interrupts are disabled and communication control parameters are programmed in the UARTx_LCTL register. The FIFO configuration is determined and the receive trigger levels are set in the UARTx_FCTL register. The status registers, UARTx_LSR and UARTx_MSR, are read to ensure that no interrupt sources are active. Interrupts are enabled (except for the transmit interrupt) and the application is ready to use the module for transmission and reception.

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



with a value of 0. READ/WRITE attributes, reset conditions, and bit descriptions of all UART registers are provided in this section.

UART Transmit Holding Register

If less than eight bits are programmed for transmission, the lower bits of the byte written to the UART Transmit Holding Register, listed in Table 26, are selected for transmission. The transmit FIFO is mapped at this address. You can write up to 16 bytes for transmission at one time to this address if the FIFO is enabled by the application. If the FIFO is disabled, this buffer is only one byte deep. These registers share the same address space as the UARTx RBR and BRGx DLR L registers.

Table 26	6. UART	Transmit	Holding	Registers
----------	---------	----------	---------	-----------

(UART0 THR = C0h, UART1 THR = D0h)

Bit		7	6	5	4	3	2	1	0
Reset		Х	Х	Х	Х	Х	Х	Х	Х
CPU Acces	s	W	W	W	W	W	W	W	W
Note: W = W	rite Only.								
Bit Position	Value	Descr	iption						
[7:0] TXD	00h– FFh	Transmit data byte.							

UART Receive Buffer Register

R

R

CPU Access

Note: R = Read Only.

The bits in this register reflect the data received. If less than eight bits are programmed for the receive function, the lower bits of the byte reflect the bits received, whereas upper unused bits are 0. The receive FIFO is mapped at this address. If the FIFO is disabled, this buffer is only one byte deep.

The registers in the UART Receive Buffer, listed in Table 27, share the same address space as the UARTx THR and BRGx DLR L registers.

R

R

R

Table 27. UART Receive Buffer Registers				(UART	(UART0_RBR = C0h, UART1_RBR = D0h)				
Bit	7	6	5	4	3	2	1	0	-
Reset	0	0	0	0	0	0	0	0	-

R

R

R

 Bit
 Position
 Value
 Description

 [7:0]
 00h Receive data byte.

 RXD
 FFh

UART Interrupt Enable Register

The UARTx_IER register, listed in Table 28, is used to enable and disable UART interrupts. The UARTx_IER registers share the same I/O addresses as the BRGx_DLR_H registers.

 Table 28. UART Interrupt Enable Registers

(UART0_IER = C1h, UART1_IER = D1h)

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.

Bit Position	Value	Description				
[7:4]	0000	Reserved				
3 MIIE	0	The modem interrupt on edge detect of status inputs is disabled.				
	1	The modem interrupt on edge detect of status inputs is enabled.				
2	0	The line status interrupt is disabled.				
LSIE	1	The line status interrupt is enabled for receive data errors: incorrect parity bit received, framing error, overrun error, or break detection.				
1	0	The transmit interrupt is disabled.				
TIE	1	The transmit interrupt is enabled. An interrupt is generated when the transmit FIFO buffer is empty indicating no bytes are available for transmission.				
0	0	The receive interrupt is disabled.				
	1	The receive interrupt and receiver time-out interrupt are enabled. An interrupt is generated if the FIFO buffer contains data ready for a READ or if the receiver times out.				

74

zilog



UART Interrupt Identification Register

The Read Only UART Interrupt Identification register, listed in Table 29, allows you to check the status of interrupts and whether the FIFO is enabled. These registers share the same I/O addresses as the UARTx_FCTL registers. Status codes for the UARTx_IIR register are listed in Table 30.

Table 29. UART Interrupt Identification Registers (UART0_IIR = C2h, UART1_IIR = D2h)

Dit		-	~	-		•	•	4	•	
BIt		1	6	5	4	3	2	1	U	
Reset		0	0	0	0	0	0	0	1	
CPU Access	Access R R R R R R						R	R	R	
Note: R = Read	d Only.									
Bit										
Position	Value	Descr	Description							
[7:6]	00	FIFO i	FIFO is disabled.							
FSIS	11	FIFO i	FIFO is enabled.							
[5:4]	00	Reserv	/ed							
[3:1] INSTS	000– 110	Interru The co interru their re interru code is service	Interrupt Status Code The code indicated in these three bits is valid only if the interrupt bit is 1. If two internal interrupt sources are active and their respective enable bits are High, only the higher priority interrupt is seen by the application. The lower priority interrupt code is indicated only after the higher priority interrupt is serviced. Table 30 lists interrupt status codes.						ive and ority terrupt s	
0	0	There	is an ac	tive inte	rrupt sou	urce with	in the U	ART.		
INTBIT	1	There	is not ar	n active	interrupt	source	within th	e UART		

Table 30. UART Interrupt Status Codes

INSTS Value	Priority	Interrupt Type
011	Highest	Receiver Line Status
010	Second	Receive Data Ready or Trigger Level
110	Third	Character Time-out
001	Fourth	Transmit Buffer Empty
000	Lowest	Modem Status



UART FIFO Control Registers

These registers, listed in Table 31, are used to monitor trigger levels, clear FIFO pointers, and enable or disable the FIFO. The UARTx_FCTL registers share the same I/O addresses as the UARTx_IIR registers.

Table 31. UAR	s (l	(UART0_FCTL = C2h, UART1_FCTL = D								
Bit		7	6	5	4	3	2	1	0	
Reset		0	0	0	0	0	0	0	1	
CPU Access		W	W	W	W	W	W	W	W	
Note: W = Write	e Only.									
Bit Position	Value	Descr	iption							
[7:6] TRIG	00	The re interru is valio	eceive Fl ipt is ger d only if	IFO trigg nerated the FIFC	ger level when the) is enat	is set to ere is 1 b oled.	1. A rec oyte in th	eive data e FIFO.	a This bit	
	01	The re interru bit is v	eceive Fl pt is ger valid only	IFO trigg nerated y if the F	ger level when the IFO is e	set to 4. ere are 4 nabled.	The rec bytes ir	eive dat the FIF	a O. This	
	10	The receive FIFO trigger level set to 8. The receive data interrupt is generated when there are 8 bytes in the FIFO. This bit is valid only if the FIFO is enabled.								
	11	The re interru This b	eceive Fl ipt is gei it is valio	IFO trigg nerated d only if t	ger level when the the FIFC	set to 14 ere are 1) is enab	I. The re 4 bytes led.	ceive da in the FI	ita FO.	
[5:3]	000b	Reser	Reserved—must be 000b.							
1	0	This b	it produ	ces no e	ffect.					
CLRTXF	1	This b pointe	it clears r. This b	the tran it is valio	smit FIF d only if f	O and re the FIFC	esets the is enab	e transmi led.	t FIFO	
2	0	This b	it produ	ces no e	ffect.					
CLRRXF	1	This b and re FIFO i	it clears sets the	the rece receive ed.	eive FIF0 FIFO po	D, clears binter. Th	the recents bit is	eive erro valid on	r FIFO, ly if the	
0 FIFOEN	0	The tra receiv	ansmit a e buffers	ind recei s are onl	ve FIFO y one by	s are dis ⁄te deep	abled. T	he trans	mit and	
	1	The tr	ansmit a	and recei	ive FIFO	s are en	abled.			



UART Line Control Register

These registers, listed in Table 32, are used to control the communication control parameters. Table 33 on page 78 lists character length and stop bit parameters.

Bit		7	6	5	4	3	2	1	0	
Reset		0	0	0	0	0	0	0	0	
CPU Access	;	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Note: R/W = F	Read/Write.									
Bit Position	Value	Descr	iption							
7 DLAB	0	Acces and D	s to the 1h is en	UART re abled.	egisters	at I/O ad	dresses	C0h, C′	1h, D0h	
	1	Acces C0h, (Access to the Baud Rate Generator registers at I/O addresses C0h, C1h, D0h and D1h is enabled.							
6	0	Do no	t send a	break s	ignal.					
	1	Send UART next fo shift re output before time. / should registe recove	Send Break UART sends a continuous 0 on the transmit output from the next following bit boundary. The transmit data in the transmit shift register is ignored. After assigning this bit High, the TXD output is made 0 only after the bit boundary is reached. Just before assigning a 0 to TXD, it clears the transmit FIFO one time. Any new data written to the transmit FIFO during a break should be written only after the THRE bit of the UARTx_LSR register goes High. This new data is transmitted after the UART recovers from the break. After the break is removed, the UART recovers from break for the next BRG edge.							
5	0	Do not force a parity error.								
FPE	1	Force (PEN) data b	a parity are both yte.	error. W h 1, an ii	/hen this ncorrect	bit and parity bi	the party t is trans	y enable smitted w	bit /ith the	
4 EPS	0	Use o transn	dd parity nit data i	/ for tran plus pari	smissior ty bit is o	n. The to odd.	tal numt	per of 1 b	it in the	
	1	Use even parity for transmission. The total number of <i>1</i> bit in the transmit data plus parity bit is even.								

zilog ₇₈

Bit		
Position	Value	Description
3	0	Parity bit transmit and receive is disabled.
PEN	1	Parity bit transmit and receive is enabled. For transmit, a parity bit is generated and transmitted with every data character. For receive, the parity is checked for every incoming data character.
[2:0] CHAR	000– 111	UART Character Parameter Selection See Table 33 for a description of these values.

Table 33. UART Character Parameter Definition

CHAR[2:0]	Character Length (Tx/Rx Data Bits)	Stop Bits (Tx Stop Bits)
000	5	1
001	6	1
010	7	1
011	8	1
100	5	2
101	6	2
110	7	2
111	8	2

UART Modem Control Registers

These registers are used to control and check the modem status.

	Table 34. UART Modem C	ontrol Registers	(UARTO MO	CTL = C4h,	UART1 MC	CTL = D4h)
--	------------------------	------------------	-----------	------------	----------	------------

Dit	7	c	5	4	2	2	4	•
ЫІ	1	0	5	4	3	2	Ĩ	U
Reset	0	0	0	0	0	0	0	0
CPU Access	R/W							
Note: R/W = Read/Write.								

zilog

79

Bit Position	Value	Description
1031101		
[7:5]	0000	Reserved—must be 000b.
4	0	LOOP-BACK is not enabled.
LOOP	1	LOOP-BACK is enabled. The UART operates in internal LOOP-BACK mode. The transmit data output port is disconnected from the internal transmit data output and set to 1. The receive data input port is disconnected and internal receive data is connected to internal transmit data. The modem status input ports are disconnected and the four bits of the modem control register are connected as modem status inputs. The two modem control output ports (OUT1 &OUT2) are set to their inactive state.
3 OUT2	0–1	This bit does not function under normal operation. In LOOP- BACK mode, this bit is connected to the DCD bit in the UART Modem Status Register.
2 OUT1	0–1	This bit does not function under normal operation. In LOOP- BACK mode, this bit is connected to the RI bit in the UART Modem Status Register.
1 RTS	0–1	Request to Send Under normal operation, the RTS output port is the inverse of this bit. In LOOP-BACK mode, this bit is connected to the CTS bit in the UART Modem Status Register.
0 DTR	0–1	Data Terminal Ready Under normal operation, the DTR output port is the inverse of this bit. In LOOP-BACK mode, this bit is connected to the DSR bit in the UART Modem Status Register.

UART Line Status Registers

These registers are used to show the status of UART interrupts and registers.

Table 35. UART Line Status Registers				(UART0_LSR = C5h, UART1_LSR = D					
Bit	7	6	5	4	3	2	1	0	
Reset	0	1	1	0	0	0	0	0	
CPU Access	R	R	R	R	R	R	R	R	
Note: R = Read Only.									



Bit Position	Value	Description
7 ERR	0	This bit is always 0 when operating with the FIFO disabled. With the FIFO enabled, this bit is reset when the UARTx_LSR register is read and there are no more bytes with an error status in the FIFO.
	1	An error is detected in the FIFO. There is at least 1 parity, framing, or Break Indication (BI) error in the FIFO.
6 TEMT	0	The Transmit Holding Register FIFO is not empty, the Transmit Shift Register is not empty, or the transmitter is not idle.
	1	The Transmit Holding Register FIFO and the Transmit Shift Register are empty; the transmitter is idle. This bit cannot be set to 1 during the Break Indication (BI). This bit is set to 1 only after the BREAK command is removed.
5	0	The Transmit Holding Register FIFO is not empty.
THRE 1		The Transmit Holding Register FIFO bit cannot be set to 1 during the Break Indication (BI). This bit is set to 1 only after the BREAK command is removed.
4 Bl	0	The receiver does not detect a Break Indication. This bit is reset to 0 when the UARTx_LSR register is read.
	1	The receiver detects a Break Indication on the receive input line. This bit is set to 1 if the duration of the Break Indication on the receive data is longer than one character transmission time, the time depends on the programming of the UARTx_LSR register. In the case of a FIFO, only one null character is loaded into the receive FIFO with the framing error. The framing error is revealed to the eZ80 [®] whenever this particular string of data is read from the receive FIFO.
3 FE	0	No framing error is detected for the character at the top of the FIFO. This bit is reset to 0 when the UARTx_LSR register is read.
	1	A framing error is detected for the character at the top of the FIFO. This bit is set to 1 when the stop bit following the data/ parity bit is logic 0.
2 PE	0	The received character at the top of the FIFO does not produce a parity error. This bit is reset to 0 when the UARTx_LSR register is read.
	1	The received character at the top of the FIFO contains a parity error.

zilog[°]

Bit Position	Valuo	Description
1 OE	0	The received character at the top of the FIFO does not contain an overrun error. This bit is reset to 0 when the UARTx_LSR register is read.
	1	An overrun error is detected. If the FIFO is not enabled, this error indicates the data in the receive buffer register was not read before the next character was transferred into the receiver buffer register. If the FIFO is enabled, this error indicates the FIFO was already full when an additional character was received by the receiver shift register. The character in the receiver shift register is not placed into the receive FIFO.
0 DR	0	This bit is reset to 0 when the UARTx_RBR register is read or when all bytes are read from the receive FIFO.
	1	Data Ready If the FIFO is not enabled, this bit is set to 1 when a complete incoming character is transferred into the receiver buffer register from the receiver shift register. If the FIFO is enabled, this bit is set to 1 when a character is received and transferred to the receive FIFO.

UART Modem Status Registers

The UART Modem Status Registers, listed in Table 36, are used to show the status of the UART signals.

Table 36. UART Line Status Registers				(UART0_MSR = C6h, UART1_MSR = D6h					
Bit	7	6	5	4	3	2	1	0	
Reset	Х	Х	Х	Х	Х	Х	Х	Х	
CPU Access	R	R	R	R	R	R	R	R	
Note: R = Read Only.									

81



82

Bit Position	Value	Description
7 DCD	0–1	Data Carrier Detect In NORMAL mode, this bit reflects the inverted state of the DCDx input pin. In LOOP-BACK mode, this bit reflects the value of the UARTx_MCTL[3] = OUT2.
6 RI	0–1	Ring Indicator In NORMAL mode, this bit reflects the inverted state of the \overline{RIx} input pin. In LOOP-BACK mode, this bit reflects the value of the UARTx_MCTL[2] = OUT1.
5 DSR	0–1	Data Set Ready In NORMAL mode, this bit reflects the inverted state of the DSRx input pin. In LOOP-BACK mode, this bit reflects the value of the UARTx_MCTL[0] = DTR.
4 CTS	0–1	Clear to Send In NORMAL mode, this bit reflects the inverted state of the CTSx input pin. In LOOP-BACK mode, this bit reflects the value of the UARTx_MCTL[1] = RTS.
3 DDCD	0–1	Delta Status Change of $\overline{\text{DCD}}$ This bit is set to 1 whenever the $\overline{\text{DCDx}}$ pin changes state. This bit is reset to 0 when the UARTx_MSR register is read.
2 TERI	0–1	Trailing Edge Change on \overline{RI} This bit is set to 1 whenever a falling edge is detected on the RIx pin. This bit is reset to 0 when the UARTx_MSR register is read.
1 DDSR	0–1	Delta Status Change of $\overline{\text{DSR}}$ This bit is set to 1 whenever the $\overline{\text{DSRx}}$ pin changes state. This bit is reset to 0 when the UARTx_MSR register is read.
0 DCTS	0–1	Delta Status Change of $\overline{\text{CTS}}$ This bit is set to 1 whenever the $\overline{\text{CTSx}}$ pin changes state. This bit is reset to 0 when the UARTx_MSR register is read.

UART Scratch Pad Registers

The UARTx_SPR registers, listed in Table 37 on page 83, can be used by the system as general-purpose Read/Write registers.

zilog[°]

Table 37. UA	(UART0_SPR = C7h, UART1_SPR = D7								
Bit		7	6	5	4	3	2	1	0
Reset		0	0	0	0	0	0	0	0
CPU Acces	S	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Note: R/W =	Read/Write.								
Bit Position	Value	Descr	iption						
[7:0] SPR	00h– FFh	UART purpo:	scratch se Read	pad reg /Write r	gister is a egister.	available	for use	as a ger	eral-

83

zilog

Serial Peripheral Interface

The Serial Peripheral Interface (SPI) is a synchronous interface allowing several SPI-type devices to be interconnected. The SPI is a full-duplex, synchronous, character-oriented communication channel that employs a four-wire interface. The SPI block consists of a transmitter, receiver, clock generator, and control unit. During an SPI transfer, data is sent and received simultaneously by both the master and the slave SPI devices.

In a serial peripheral interface, separate signals are required for data and clock. The eZ80190 device contains two SPI devices—one within each Universal Zilog Interface (UZI) block. The SPI devices may be configured as either master or slave. The connection of two SPI devices (one master and one slave) and the direction of data transfer is displayed in Figure 13.



Figure 13. SPI Master—Slave Connection

SPI Signals

The four basic SPI signals (MISO, MOSI, \overline{SS} , and SCK) are discussed in the following paragraphs. Each signal is described in both MASTER and SLAVE modes.

Master In Slave Out

The Master In Slave Out (MISO) pin is configured as an input in a master device and as an output in a slave device. It is one of the two lines that transfer serial data, with the msb sent first. The MISO pin of a slave device is placed in a high-impedance state if the slave

zilog |

is not selected. When the SPI is not enabled by the UZI Control register, this signal operates in a high-impedance state.

Master Out Slave In

The Master Out Slave In (MOSI) pin is configured as an output in a master device and as an input in a slave device. It is one of the two lines that transfer serial data, with the msb sent first. When the SPI is not enabled by the UZI Control register, this signal operates in a high-impedance state.

Slave Select

The active Low Slave Select (\overline{SS}) input signal is used to select a slave SPI device. It must be operating in a Low state prior to all data communication and must stay Low for the duration of the data transfer.

The \overline{SS} input signal on the master must be in a High state. If the \overline{SS} signal goes Low, a Mode Fault error flag (MODF bit) is set in the SPIx_SR register. For more information see SPI Status Register (SPI0 SR = B7h, SPI1 SR = BBh) on page 90.

When the SPI Clock Phase (CPHA) bit = 0, the shift clock is the OR of \overline{SS} with SCK. In this clock phase mode, \overline{SS} must go High between successive characters in an SPI message. When CPHA = 1, \overline{SS} can remain Low for several SPI characters. In cases where there is only one SPI slave MCU, its \overline{SS} line could be tied Low as long as CPHA = 1 CLOCK mode is used. For more information on the CPHA bit see SPI Control Register (SPI0_CTL = B6h, SPI1_CTL = BAh) on page 89.

Serial Clock

The Serial Clock (SCK) is used to synchronize data movement both in and out of the device through its MOSI and MISO pins. The master and slave are each capable of exchanging a byte of data during a sequence of eight clock cycles. Because SCK is generated by the master, the SCK pin becomes an input on a slave device. The SPI contains an internal divide-by-two clock divider. The SPI serial clock is one-half the frequency of the clock signal created by the UZI Baud Rate Generator, as shown by the following equation:

SPI Data Transfer Rate (bits/s) = System Clock Frequency

2 x Baud Rate Generator Divisor

As displayed in Figure 14 on page 86 and Table 38 on page 86, four possible timing relations may be selected when using control bits CPOL and CPHA in the SPI Control register. See SPI Control Register (SPI0_CTL = B6h, SPI1_CTL = BAh) on page 89. Both the

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

zilog[®]|₈₇

responds by sending data to the master via the master's MISO signal. The result is a fullduplex transmission, with both data out and data in synchronized with the same clock signal. The byte transmitted is replaced by the byte received, eliminating the requirement for separate transmit-empty and receive-full status bits. A single status bit, SPIF, is used to signify that the I/O operation is completed, see SPI Status Register (SPI0_SR = B7h, SPI1_SR = BBh) on page 90.



Figure 15. SPI Block Diagram

The SPI is double-buffered on read, but not on write. If a write is performed during data transfer, the transfer occurs uninterrupted, and the write is unsuccessful. This condition causes the WRITE COLLISION (WCOL) status bit in the SPIx_SR register to be set. After a data byte is shifted, the SPI flag bit (SPIF) of the SPIx_SR register is set.

In SPI MASTER mode, the SCK pin is an output. It idles High or Low, depending on the CPOL bit in the SPIx_CTL register, until data is written to the shift register. When data is

zilog |

written to the shift register, eight clocks are generated to shift the eight bits of data in both directions. The SCK signal then enters the IDLE state.

In SPI SLAVE mode, the start logic receives a logic Low from the \overline{SS} pin and a clock input at the SCK pin, and the slave is synchronized to the master. Data from the master is received serially from the slave MOSI signal and loads the 8-bit shift register. After the 8bit shift register is loaded, its data is parallel transferred to the read buffer. During a write cycle data is written into the shift register, then the slave waits for the SPI master to initiate a data transfer, supply a clock signal, and shift the data out on the slave's MISO signal.

If the CPHA bit in the SPIx_CTL register is 0, a transfer begins when \overline{SS} pin signal goes Low and the transfer ends when \overline{SS} goes High after eight clock cycles on SCK. When the CPHA bit is set to 1, a transfer begins the first time SCK becomes active while \overline{SS} is Low and the transfer ends when the SPIF flag gets set.

SPI Flags

Mode Fault

The Mode Fault flag (SPIx_SR[4] = MODF) indicates that there may be a multimaster conflict for system control. The MODF bit is normally cleared to 0. It is only set to 1 when the master device's \overline{SS} pin is pulled Low. When a mode fault is detected, the following occurs:

- 1. The MODF flag (SPIx_SR[4]) is set to 1.
- 2. The SPI device is disabled by clearing the SPI_EN bit (SPIx_CTL[5]) to 0.
- 3. The MASTER_EN bit (SPIx_CTL[4]) is cleared to 0, forcing the device into SLAVE mode.
- 4. If enabled (IRQ_EN = SPIx_CTL[7] = 1), an SPI interrupt is generated.

Clearing the Mode Fault flag is performed by reading the SPI Status register. The other SPI control bits (SPI_EN and MASTER_EN) must be restored to their original states by user software after the Mode Fault flag is cleared.

Write Collision

The WRITE COLLISION flag (SPIx_SR[5] = WCOL) is set to 1 when an attempt is made to write to the SPI Transmit Shift register (SPIx_TSR) while data is being transferred. Clearing the WCOL bit is performed by reading SPIx_SR with the WCOL bit set.

zilog ₈₉

SPI Registers

There are four registers in the Serial Peripheral Interface which provide control, status, and data storage functions. These registers are called the SPI Control register (SPIx_CTL), SPI Status register (SPIx_SR), SPI Receive Buffer register (SPIx_RBR), and SPI Transmit Shift register (SPIx_TSR), where the x in each register name is either 0 or 1 depending on which UZI device the SPI is located within. The SPI registers are described in this section.

SPI Control Register

The SPI Control Register, listed in Table 39, is used to control and set up the serial peripheral interface.

Table 39	. SPI	Control	Register
----------	-------	---------	----------

(SPI0_CTL = B6h, SPI1_CTL = BAh)

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

Rit		
D 11	D	4
	-	IT

Position	Value	Description
7	0	The SPI system interrupt is disabled.
IRQ_EN	1	The SPI system interrupt is enabled.
6	0	Reserved—must be 0.
5	0	The SPI is disabled.
SPI_EN	1	The SPI is enabled.
4	0	When enabled, the SPI operates as a slave.
MASTER_EN	1	When enabled, the SPI operates as a master.
3	0	The master SCK pin idles in a Low (0) state.
CPOL	1	The master SCK pin idles in a High (1) state.
2	0	SS must go High after a transfer of every byte of data.
СРНА	1	SS can remain Low to transfer any number of data bytes.
[1:0]	00b	Reserved—must be 0.



SPI Status Register

The SPI Status Read Only register, listed in Table 40, returns the status of data transmitted using the serial peripheral interface. Reading the SPIx_SR register clears bits 7, 6, and 4 to a logical 0.

Table 40. SPI Status Register(SPI0_SR = B7h, SPI1_SR = BBh)

7	6	5	4	3	2	1	0
0	0	0	0	0	1	0	0
R	R	R	R	R	R	R	R
	7 0 R	7 6 0 0 R R	7 6 5 0 0 0 R R R	7 6 5 4 0 0 0 0 R R R R	7 6 5 4 3 0 0 0 0 0 R R R R R	7 6 5 4 3 2 0 0 0 0 1 R R R R R R	7 6 5 4 3 2 1 0 0 0 0 0 1 0 R R R R R R R

Note: R = Read Only.

Bit Position	Value	Description		
7	0	The SPI data transfer is not finished.		
SPIF	1	The SPI data transfer is finished. If enabled, an interrupt is generated. This bit flag is cleared to 0 by a read of the SPIx_SR register.		
6	0	An SPI write collision is not detected.		
WCOL	1	An SPI write collision is detected. This bit flag is cleared to 0 by a read of the SPIx_SR registers.		
5	0	Reserved—must be 0.		
4	0	A mode fault (multimaster conflict) is not detected.		
MODF	1	A mode fault (multimaster conflict) is detected. This bit flag is cleared to 0 by a read of the SPIx_SR register.		
[3:0]	0000b	Reserved—must be 0.		

SPI Transmit Shift Register

The SPI Transmit Shift register (SPIx_TSR) is used by the SPI master to transmit data onto the SPI serial bus to the slave device. A write to the SPIx_TSR register places data directly into the shift register for transmission. A write to this register within an SPI device configured as a master initiates transmission of a byte of the data loaded into the register. After completing this transmission, the SPIF status bit (SPIx_SR[7]) is set to 1 in both the master and slave devices.

The SPI Transmit Shift Write Only registers share the same address space as the SPI Receive Buffer Read Only registers.



Table 41. SPI T	ransmit S	hift Re	egister		(SPI0_TSR = B8h, SPI1_TSR = BCh)						
Bit		7	6	5	4	3	2	1	0		
Reset		Х	Х	Х	Х	Х	Х	Х	Х		
CPU Access		W	W	W	W	W	W	W	W		
Note: W = Write	Only.										
Bit Position	Value	Desc	ription								
7 TX_DATA	00h– FFh	SPI	data tran	ismissio	on.						

SPI Receive Buffer Register

The SPI Receive Buffer register (SPIx RBR) is used by the SPI slave to receive data from the serial bus. A write to the (SPIx TSR) register initiates reception of another byte, and only occurs in the master device. When you read the SPIx RBR register, a buffer is being read. The first SPIF bit must be cleared by the time a second transfer of data from the shift register is initiated or an OVERRUN condition exists. Should an overrun occur, the byte causing the overrun is lost.

The SPI Receive Buffer Read Only registers share the same address space as the SPI Transmit Shift Write Only registers.

Table 42. SPI Receive Buffer Register					(SPI0_	RBR = E	38h, SPI	1_RBR	= BCh)
Bit		7	6	5	4	3	2	1	0
Reset		Х	Х	Х	Х	Х	Х	Х	Х
CPU Access		R	R	R	R	R	R	R	R
Note: R = Read	Only.								
Bit	Valuo	Dosc	rintion						
FUSILIUII	value	Dest	Inpuon						
7 RX_DATA	00h– FFh	SPI	lata rece	eption.					



I²C Serial I/O Interface

I²C General Characteristics

The I²C serial I/O bus is a two-wire communication interface that can operate in four modes:

- 1. MASTER TRANSMIT
- 2. MASTER RECEIVE
- 3. SLAVE TRANSMIT
- 4. SLAVE RECEIVE

The I²C interface consists of the Serial Clock (SCL) and the Serial Data (SDA). Both SDA and SCL are bidirectional lines, connected to a positive supply voltage via an external pull-up resistor. When the bus is free, both lines are High. The output stages of devices connected to the bus must be configured as open-drain outputs. Data on the I²C bus can be transferred at a rate of up to 100 kbps in STANDARD mode, or up to 400 kbps in FAST mode. One clock pulse is generated for each data bit transferred.

Clocking Overview

If another device on the I^2C bus drives the clock line when the I^2C is operating in MAS-TER mode, the I^2C synchronizes its clock to the I^2C bus clock. The High period of the clock is determined by the device that generates the shortest High clock period. The Low period of the clock is determined by the device that generates the longest Low clock period.

A slave may stretch the Low period of the clock to slow down the bus master. The Low period can also be stretched for handshaking purposes. For both circumstances, this Low period can be stretched after each bit transfer or each byte transfer. The I²C stretches the clock after each byte transfer until the IFLG bit in the I2Cx_CTL register is cleared.

Bus Arbitration Overview

In MASTER mode, the I²C checks that each transmitted logic 1 appears on the I²C bus as a logic 1. If another device on the bus overrules and pulls the SDA signal Low, arbitration is lost. If arbitration is lost during the transmission of a data byte or a NACK bit, the I²C returns to the IDLE state. If arbitration is lost during the transmission of an address, the I²C switches to SLAVE mode so that it can recognize its own slave address or the general call address.



Data Validity

The data on the SDA line must be stable during the High period of the clock. The High or Low state of the data line can only change when the clock signal on the SCL line is Low, as displayed in Figure 16.



Figure 16. I²C Clock and Data Relationship

START and STOP Conditions

Within the I²C bus protocol, unique situations arise which are defined as START and STOP conditions, see Figure 17. A High-to-Low transition on the SDA line while SCL is High indicates a START condition. A Low-to-High transition on the SDA line while SCL is High defines a STOP condition.

START and STOP conditions are always generated by the master. The bus is considered to be busy after the START condition. The bus is considered to be free at a defined time after the STOP condition.



Figure 17. START and STOP Conditions In I²C Protocol

Transferring Data

Byte Format

Every character transferred on the SDA line must be a single 8-bit byte. The number of bytes that can be transmitted per transfer is unrestricted. Each byte must be followed by an

zilog |₉

 I^2C Acknowledge (ACK). Data is transferred with the msb first, see Figure 18. A receiver can hold the SCL line Low to force the transmitter into a WAIT state. Data transfer then continues when the receiver is ready for another byte of data and releases SCL.



Figure 18. I²C Frame Structure

Acknowledge

Data transfer with an ACK function is obligatory. The ACK-related clock pulse is generated by the master. The transmitter releases the SDA line (High) during the ACK clock pulse. The receiver must pull down the SDA line during the ACK clock pulse so that it remains Low during the High period of this clock pulse. See Figure 19 on page 95.

A receiver that is addressed is obliged to generate an ACK after each byte is received. When a slave receiver does not acknowledge the slave address (that is, the slave receiver is unable to receive because it is performing some real-time function), the data line must be left High by the slave. The master then generates a STOP condition to abort the transfer.

If a slave receiver acknowledges the slave address, but cannot receive any more data bytes, the master must abort the transfer. The abort is indicated by the slave generating the Not Acknowledge (NACK) on the first byte to follow. The slave leaves the data line High and the master generates the STOP condition.

If a master receiver is involved in a transfer, it must signal the end of the data transfer to the slave transmitter by not generating an ACK on the final byte clocked out of the slave. The slave transmitter must release the data line to allow the master to generate a STOP or a repeated START condition.



Figure 19. I²C Acknowledge

Clock Synchronization

All masters generate their own clocks on the SCL line to transfer messages on the I^2C bus. Data is only valid during the High period of each clock.

Clock synchronization is performed using the wired AND connection of the I^2C interfaces to the SCL line, meaning that a High-to-Low transition on the SCL line causes the relevant devices to start counting from their Low period. When a device clock goes Low, it holds the SCL line in that state until the clock High state is reached. See Figure 20 on page 96. The Low-to-High transition of this clock, however, may not change the state of the SCL line if another clock still exists within its Low period. The SCL line is held Low by the device with the longest Low period. Devices with shorter Low periods enter a High WAIT state during this time.

When all devices complete counting off their Low periods, the clock line goes High. There is no difference between the device clocks and the state of the SCL line; therefore, all of the devices begin counting their High periods. The first device to complete its High period again pulls the SCL line Low. In this way, a synchronized SCL clock is generated with its Low period determined by the device with the longest clock Low period, and its High period determined by the device with the shortest clock High period.

95

eZ80190

Product Specification

zilog



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



In other words, arbitration is not allowed between:

- A repeated START condition and a data bit
- A STOP condition and a data bit
- A repeated START condition and a STOP condition

Clock Synchronization for Handshake

The Clock synchronizing mechanism can function as a handshake, enabling receivers to cope with fast data transfers, on either a byte or bit level. The byte level allows a device to receive a byte of data at a fast rate, but allows the device more time to store the received byte or to prepare another byte for transmission. Slaves hold the SCL line Low after reception and acknowledge the byte, forcing the master into a WAIT state until the slave is ready for the next byte transfer in a handshake procedure.

Operating Modes

Master Transmit

In MASTER TRANSMIT mode, the I²C transmits a number of bytes to a slave receiver.

The device enters MASTER TRANSMIT mode by setting the Master Mode Start bit (STA) bit in the I2Cx_CTL register to 1. The I²C then tests the I²C bus and transmits a START condition when the bus is free. When a START condition is transmitted, the IFLG bit is set to 1 and the status code in the I2Cx_SR register is 08h. Before this interrupt is serviced, the I2Cx_DR register must be loaded with either a 7-bit slave address or the first part of a 10-bit slave address, with the lsb cleared to 0 to specify TRANSMIT mode. The IFLG bit should now be cleared to 0 to prompt the transfer to continue.

After the 7-bit slave address (or the first part of a 10-bit address) plus the write bit are transmitted, the IFLG is set again. A number of status codes are possible in the I2Cx_SR register.

zilog[°]

98

Code	I ² C State	Microprocessor Response	Next I ² C Action
18h	Addr+W transmitted, ACK received	For a 7-bit address: Write a byte to DATA, clear IFLG	Transmit data byte, receive ACK
		Or set STA, clear IFLG	Transmit repeated START
		Or set STP, clear IFLG	Transmit STOP
		Or set STA & STP, clear IFLG	Transmit STOP then START
		For a 10-bit address: Write an extended address byte to DATA, clear IFLG	Transmit extended address byte
20h	Addr+W transmitted, ACK not received	Same as code 18h	Same as code 18h
38h	Arbitration lost	Clear IFLG	Return to the IDLE state
		Or set STA, clear IFLG	Transmit START when bus is free
68h	Arbitration lost, SLA+W received, ACK	Clear IFLG, ACK = 0	Receive data byte, transmit NACK
	transmitted	Or clear IFLG, ACK = 1	Receive data byte, transmit ACK
78h	Arbitration lost, General call addr received, ACK transmitted	Same as code 68h	Same as code 68h
B0h	Arbitration lost, SLA+R received, ACK	Write byte to DATA, clear IFLG, clear ACK = 0	Transmit last byte, receive ACK
	transmitted	Or write byte to DATA, clear IFLG, set ACK = 1	Transmit data byte, receive ACK

Table 43. I²C Master Transmit Status Codes

If 10-bit addressing is being used, then the status code is 18h or 20h after the first part of a 10-bit address, plus the write bit, are successfully transmitted.

After this interrupt is serviced and the second part of the 10-bit address is transmitted, the I2Cx_SR register contains one of the codes in Table 44 on page 99.
zilog[°]

Table 44. I²C 10-Bit Master Transmit Status Codes

Code	I ² C State	Microprocessor Response	Next I ² C Action
38h	Arbitration lost	Clear IFLG	Return to the IDLE state
		Or set STA, clear IFLG	Transmit START when the bus is free
68h	Arbitration lost, SLA+W received, ACK	Clear IFLG, clear ACK = 0	Receive data byte, transmit NACK
transmitted		Or clear IFLG, set ACK=1	Receive data byte, transmit ACK
B0h	Arbitration lost, SLA+R received, ACK	Write byte to DATA, clear IFLG, clear ACK = 0	Transmit last byte, receive ACK
transmitted	transmitted	Or write byte to DATA, clear IFLG, set ACK = 1	Transmit data byte, receive ACK
D0h	Second Address byte + W transmitted, ACK	Write byte to DATA, clear IFLG	Transmit data byte, receive ACK
	received	Or set STA, clear IFLG	Transmit repeated START
		Or set STP, clear IFLG	Transmit STOP
		Or set STA & STP, clear IFLG	Transmit STOP then START
D8h	Second Address byte + W transmitted, ACK not received	Same as code D0h	Same as code D0h

If a repeated START condition is transmitted, the status code is 10h instead of 08h. After each data byte is transmitted, the IFLG is set to 1 and one of the status codes listed in Table 45 on page 100 is contained in the I2Cx_SR register.

99

zilog

Code	I ² C State	Microprocessor Response	Next I ² C Action
28h	Data byte transmitted, ACK received	Write byte to DATA, clear IFLG	Transmit data byte, receive ACK
		Or set STA, clear IFLG	Transmit repeated START
		Or set STP, clear IFLG	Transmit STOP
		Or set STA & STP, clear IFLG	Transmit START then STOP
30h	Data byte transmitted, ACK not received	Same as code 28h	Same as code 28h
38h	Arbitration lost	Clear IFLG	Return to the IDLE state
		Or set STA, clear IFLG	Transmit START when bus free

Table 45. I²C Master Transmit Status Codes For Data Bytes

When all bytes are transmitted, the microprocessor should write a 1 to the Master Mode Stop bit (STP) bit in the I2Cx_CTL register. The I²C then transmits a STOP condition, clears the STP bit, and returns to the IDLE state.

Master Receive

In MASTER RECEIVE mode, the I²C receives a number of bytes from a slave transmitter.

After the START condition is transmitted, the IFLG bit is set to 1 and the status code 08h is loaded in the I2Cx_SR register. The I2Cx_DR register should be loaded with the slave address (or the first part of a 10-bit slave address), with the lsb set to 1 to signify a READ. The IFLG bit should be cleared to 0 as a prompt for the transfer to continue.

When the 7-bit slave address (or the first part of a 10-bit address) and the read bit are transmitted, the IFLG bit is set and one of the status codes listed in Table 46 on page 101 is contained in the I2Cx_SR register.

zilog[°]

Code	I ² C State	Microprocessor Response	Next I ² C Action
40h	Addr + R transmitted, ACK received	For a 7-bit address, clear IFLG, ACK = 0	Receive data byte, transmit NACK
		Or clear IFLG, ACK = 1	Receive data byte, transmit ACK
		For a 10-bit address, write extended address byte to DATA, clear IFLG	Transmit extended address byte
48h	Addr + R transmitted, ACK not received	For a 7-bit address: set STA, clear IFLG	Transmit repeated START
		Or set STP, clear IFLG	Transmit STOP
		Or set STA & STP, clear IFLG	Transmit STOP then START
		For a 10-bit address: Write extended address byte to DATA, clear IFLG	Transmit extended address byte
38h	Arbitration lost	Clear IFLG	Return to the IDLE state
		Or set STA, clear IFLG	Transmit START when bus is free
68h	Arbitration lost, SLA+W received, ACK	Clear IFLG, clear ACK = 0	Receive data byte, transmit NACK
	transmitted	Or clear IFLG, set ACK = 1	Receive data byte, transmit ACK
78h	Arbitration lost, General call addr received, ACK transmitted	Same as code 68h	Same as code 68h
B0h	Arbitration lost, SLA+R received, ACK	Write byte to DATA, clear IFLG, clear ACK = 0	Transmit last byte, receive ACK
	transmitted	Or write byte to DATA, clear IFLG, set ACK = 1	Transmit data byte, receive ACK
Note: F	R = Read bit. The lsb is set t	o 1.	

Table 46. I²C Master Receive Status Codes

If 10-bit addressing is being used, the slave is first addressed using the full 10-bit address plus the Write bit. The master then issues a restart followed by the first part of the 10-bit



address again, but with the READ bit. The status code is then 40h or 48h. The slave remains selected prior to the restart.

If a repeated START condition is received, the status code is 10h instead of 08h.

After each data byte is received, the IFLG is set and one of the status codes listed in Table 47 is contained in the I2Cx_SR register.

Table 47. I²C Master Receive Status Codes For Data Bytes

Code	I ² C State	Microprocessor Response	Next I ² C Action
50h	Data byte received, ACK transmitted	Read DATA, clear IFLG, clear ACK = 0	Receive data byte, transmit NACK
		Or read DATA, clear IFLG, set ACK = 1	Receive data byte, transmit ACK
58h	Data byte received, NACK transmitted	Read DATA, set STA, clear IFLG	Transmit repeated START
		Or read DATA, set STP, clear IFLG	Transmit STOP
		Or read DATA, set STA & STP, clear IFLG	Transmit STOP then START
38h	Arbitration lost in NACK bit	Same as master transmit	Same as master transmit

When all bytes are received, a NACK is sent. Next, the microprocessor writes a 1 to the STP bit in the I2Cx_CTL register. The I²C then transmits a STOP condition, clears the STP bit, and returns to the IDLE state.

Slave Transmit

In SLAVE TRANSMIT mode, a number of bytes are transmitted to a master receiver. The I²C enters SLAVE TRANSMIT mode when it receives its own slave address and a read bit after a START condition. The I²C then transmits an I²C Acknowledge bit (ACK) if it is set to 1, and sets the IFLG bit in the I2Cx_CTL register. The I2Cx_SR register contains the status code A8h.

Note:

When the I²C contains a 10-bit slave address (signified by F0h–F7h in the I2Cx_SAR register), it transmits an ACK after the first address byte is received after a restart. An interrupt is generated, IFLG is set; however, the status does not change. No second address byte is sent by the master. The slave remains selected prior to the restart.

zilog

I²C goes from MASTER mode to SLAVE TRANSMIT mode when arbitration is lost dur-

ing the transmission of an address, and the slave address and read bit are received. This action is confirmed by the status code B0h in the I2Cx_SR register.

The data byte to be transmitted is loaded into the I2Cx_DR register and the IFLG bit is cleared. After the I²C transmits the byte and receives an ACK, the IFLG bit is set and the I2Cx_SR register contains B8h. After the last byte to be transmitted is loaded into the I2Cx_DR register, the ACK bit is cleared when the IFLG is cleared. After the last byte is transmitted, the IFLG is set and the I2Cx_SR register contains C8h. The I²C returns to the IDLE state. The ACK bit must be set to 1 before SLAVE mode can be reentered.

If no ACK is received after transmitting a byte, the IFLG is set and the I2Cx_SR register contains C0h. The I²C then returns to the IDLE state.

If a STOP condition is detected after an ACK bit, the I²C returns to the IDLE state.

Slave Receive

In SLAVE RECEIVE mode, a number of data bytes are received from a master transmitter.

The I²C enters SLAVE RECEIVE mode when it receives its own slave address and a write bit (lsb = 0) after a START condition. The I²C transmits an ACK bit and sets the IFLG bit in the I2Cx_CTL register. The I2Cx_SR register then contains the status code 60h. The I²C also enters SLAVE RECEIVE mode when it receives the general call address 00h (if the GCE bit in the I2Cx_SAR register is set). The status code is then 70h.

Note:

When the l^2C contains a 10-bit slave address (signified by F0h–F7h in the $l2Cx_SAR$ register), it transmits an ACK after the first address byte is received; however, no interrupt is generated. IFLG is not required to be set and the status does not change. The l^2C generates an interrupt only after the second address byte is received. The l^2C then sets the IFLG bit and loads the status code, as described above.

I²C goes from MASTER mode to SLAVE RECEIVE mode when arbitration is lost during the transmission of an address, and the slave address and write bit (or the general call address if the CGE bit in the I2Cx_SAR register is set to 1) are received. The status code in the I2Cx_SR register is 68h if the slave address is received or 78h if the general call address is received. The IFLG bit must be cleared to 0 to allow data transfer to continue.

If the ACK bit in the I2Cx_CTL register is set to 1, then an ACK bit (Low level on SDA) is transmitted and the IFLG bit is set after each byte is received. The I2Cx_SR register contains the status code 80h or 90h if SLAVE RECEIVE mode is entered with the general call address. The received data byte can be read from the I2Cx_DR register and the IFLG bit must be cleared to allow the transfer to continue. If a STOP condition or a repeated START condition is detected after the ACK bit, the IFLG bit is set and the I2Cx_SR register contains status code A0h.

zilog

If the ACK bit is cleared to 0 during a transfer, the I^2C transmits a NACK bit (High level on SDA) after the next byte is received, and sets the IFLG bit. The I2Cx_SR register contains the status code 88h or 98h if SLAVE RECEIVE mode is entered with the general call address. The I^2C returns to the IDLE state when the IFLG bit is cleared to 0.

I²C Registers

Addressing

The processor interface provides access to six 8-bit registers: four Read/Write registers, one Read Only register, and two Write Only registers. See Table 48.

Register	Description
I2Cx_SAR	Slave address register
I2Cx_xSAR	Extended slave address register
I2Cx_DR	Data byte register
I2Cx_CTL	Control register
I2Cx_SR	Status register (Read Only)
I2Cx_CCR	Clock Control register (Write Only)
I2Cx_SRR	Software reset register (Write Only)
Note: The lower	case x in the register name can be either 0 or 1 depending upon which of the two s are referenced within the eZ80190 device.

 Table 48. I²C Register Descriptions

Resetting the I²C Registers

Hardware Reset—When the I^2C is reset by a hardware reset of the eZ80190 device, the I2Cx_SAR, I2Cx_xSAR, I2Cx_DR and I2Cx_CTL registers are cleared to 00h. The I2Cx_SR register is set to F8h.

Software Reset—Perform a software reset by writing any value to the I²C Software Reset register (I2Cx_SRR). A software reset sets the I²C back to the IDLE state and the STP, STA, and IFLG bits of the I2Cx_CTL register to 0.

zilog

105

I²C Slave Address Register

The I2Cx_SAR register, indicated in Table 49, lists the 7-bit address of the I²C when in SLAVE mode and allows 10-bit addressing in conjunction with the I2Cx_xSAR register. I2Cx_SAR[7:1] = SLA[6:0] is the 7-bit address of the I²C when in 7-bit SLAVE mode. When the I²C receives this address after a START condition, it enters SLAVE mode. I2Cx_SAR[7] corresponds to the first bit received from the I²C bus.

When the register receives an address starting with F7h to F0h (I2Cx_SAR[7:3] = 11110b), the I²C recognizes that a 10-bit slave addressing mode is being selected. The I²C sends an ACK after receiving the I2Cx_SAR byte (the device does not generate an interrupt at this point). After the next byte of the address (I2Cx_xSAR) is received, the I²C generates an interrupt and enters SLAVE mode. Then I2Cx_SAR[2:1] is used as the upper 2 bits of the 10-bit extended address. The full 10-bit address is returned by $\{I2Cx_SAR[2:1], I2Cx_xSAR[7:0]\}$.

(12C0 SAR = C8h 12C1 SAR = D8h)

								Boni,	
	7	6	5	4	3	2	1	0	
	0	0	0	0	0	0	0	0	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Read/Write.									
Value	Descri	otion							
00h– 7Fh	The 7-b when o	it slave a perating	address, in 10-bi	, or the lo t mode.	ower 7 b	its of the	e slave a	ddress,	
0	The I ² C	is not e	nabled t	o recogr	nize the	General	Call Add	dress.	
1	The I ² C is enabled to recognize the General Call Address.								
	Value 00h- 7Fh 0 1	Value Descrip 00h- The 7-b 7 0 R/W Descrip 00h- The 7-b 7Fh when o 0 The 1 ² C 1 The 1 ² C	7 6 0 0 R/W R/W Value Description 00h- The 7-bit slave 7Fh when operating 0 The I ² C is not e 1 The I ² C is enable	7 6 5 0 0 0 R/W R/W R/W Value Description 00h- The 7-bit slave address, when operating in 10-bit 0 The I ² C is not enabled to re 1 The I ² C is enabled to re	7 6 5 4 0 0 0 0 R/W R/W R/W R/W Value Description 00h- The 7-bit slave address, or the low when operating in 10-bit mode. 0 The I ² C is not enabled to recognize 1 The I ² C is enabled to recognize	7 6 5 4 3 0 0 0 0 0 R/W R/W R/W R/W R/W Itead/Write. Itead/Write. Itead/Write. Itead/Write. Value Description Itead/Write. Itead/Write. 0 0h- The 7-bit slave address, or the lower 7 b Itead/Write. 0 The 1 ² C is not enabled to recognize the Generational to recognice the Generational to recognize to recognize	7 6 5 4 3 2 0 0 0 0 0 0 0 R/W R/W R/W R/W R/W R/W Value Description 00h- The 7-bit slave address, or the lower 7 bits of the when operating in 10-bit mode. 0 The I ² C is not enabled to recognize the General Cal 0 The I ² C is enabled to recognize the General Cal 0	76543210000000R/WR/WR/WR/WR/WR/WR/WRead/Write.Value Description00h-The 7-bit slave address, or the lower 7 bits of the slave a when operating in 10-bit mode.0The I^2C is not enabled to recognize the General Call Address1The I^2C is enabled to recognize the General Call Address	

Table 49. I²C Slave Address Registers

I²C Extended Slave Address Register

The I2Cx_xSAR register, listed in Table 50, is used in conjunction with the I2Cx_SAR register to provide 10-bit addressing for the I²C when in SLAVE mode. The I2Cx_SAR value forms the lower 8 bits of the 10-bit slave address. The full 10-bit address is returned by $\{I2Cx_SAR[2:1], I2Cx_xSAR[7:0]\}$.

When the register receives an address starting with F7h to F0h (I2Cx_SAR[7:3] = 11110b), the I²C recognizes that a 10-bit slave addressing mode is being selected. The I²C sends an ACK after receiving the I2Cx_SAR byte (the device does not generate an interrupt at this point). After the next byte of the address (I2Cx_xSAR) is received, the I²C generates an interrupt and enters SLAVE mode. Then I2Cx_SAR[2:1] is used as the upper 2 bits of the 10-bit extended address. The full 10-bit address is returned by $\{I2Cx_SAR[2:1], I2Cx_xSAR[7:0]\}$.

zilog'.

Bit		7	6	5	4	3	2	1	0
Reset		0	0	0	0	0	0	0	0
CPU Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Note: R/W = Re	ead/Write.								
Bit Position	Value	Desc	ription						
[7:0] SLAX	00h– FFh	Leas	t signific	ant 8 bit	s of the	10-bit ex	ktended	slave ac	dress.

Table 50. I²C Extended Slave Address Registers (I2C0_xSAR=C9h, I2C1_xSAR=D9h)

I²C Data Register

The I²C Data Register, listed in Table 51, contains the data byte/slave address to be transmitted or the data byte just received. In MASTER TRANSMIT or SLAVE TRANSMIT modes, the msb of the byte is transmitted first. In MASTER RECEIVE or SLAVE RECEIVE modes, the first bit received is placed in the msb of the register. After each byte is transmitted, the I2Cx_DR register contains the byte that is present in the event of lost arbitration.

Table 51. I ² C Data Registers	(I2C0_DR = CAh, I2C1_DR = DAh)
---	--------------------------------

Bit		7	6	5	4	3	2	1	0
Reset		0	0	0	0	0	0	0	0
CPU Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Note: R/W = Read/	Write.								
Bit Position	Value	Desci	ription						
[7:0] DATA	00h– FFh	l ² C da	ata byte.						

I²C Control Register

The I2Cx_CTL register, listed in Table 52 is a control register that is used to control the interrupts and the master slave relationships on the I^2C bus.

When the Interrupt Enable bit (IEN) is set to 1, the interrupt line goes High when the IFLG is set to 1. When IEN is cleared to 0, the interrupt line always remains Low.

zilog

When the Bus Enable bit (ENAB) is set to 0, the I^2C bus inputs SCLx. SDAx is ignored and the I^2C module does not respond to any address on the bus. When ENAB is set to 1, the I^2C responds to calls to its slave address and to the general call address if the GCE bit (I2Cx SAR[0]) is set to 1.

When the MASTER Mode Start bit (STA) is set to 1, the I²C enters MASTER mode and sends a START condition on the bus when the bus is free. If the STA bit is set to 1 when the I²C module is already in MASTER mode and one or more bytes are transmitted, then a repeated START condition is sent. If the STA bit is set to 1 when the I²C block is being accessed in SLAVE mode, the I²C completes the data transfer in SLAVE mode and then enters MASTER mode when the bus is released. The STA bit is automatically cleared after a START condition is set. Writing a 0 to this bit produces no effect.

If the MASTER Mode Stop bit (STP) is set to 1 in MASTER mode, a STOP condition is transmitted on the I^2C bus. If the STP bit is set to 1 in SLAVE mode, the I^2C module behaves as if a STOP condition is received, but no STOP condition is transmitted. If both STA and STP bits are set, the I^2C block first transmits the STOP condition (if in MASTER mode) and then transmits the START condition. The STP bit is cleared automatically. Writing a 0 to this bit produces no effect.

The I²C Interrupt Flag (IFLG) is set to 1 automatically when the device enters any of 30 of the possible 31 I²C states. The only state that does not set the IFLG bit is state F8h. If IFLG is set to 1 and the IEN bit is also set to 1, an interrupt is generated. When IFLG is set by the I²C, the Low period of the I²C bus clock line is stretched and the data transfer is suspended. When a 0 is written to IFLG, the interrupt is cleared and the I²C clock line is released.

When the I²C Acknowledge bit (ACK) is set to 1, an acknowledgement is sent during the Acknowledge clock pulse on the I²C bus if:

- Either the whole of a 7-bit slave address or the first or second byte of a 10-bit slave address is received
- The general call address is received and the General Call Enable bit in I2Cx_SAR is set to 1
- A data byte is received in MASTER or SLAVE mode

When ACK is cleared to 0, a NACK is sent when a data byte is received in MASTER or SLAVE mode. If ACK is cleared to 0 in SLAVE TRANSMIT mode, the byte in the I2Cx_DR register is presumed to be the last byte. After this byte is transmitted, the I²C block enters state C8h, then returns to the IDLE state. The I²C module does not respond to its slave address unless ACK is set.

zilog

108

Table 52. I ² C Co	ontrol Re	gister	S	(I2C0_CTL = CBh, I2C1_CTL = DBh)							
Bit		7	6	5	4	3	2	1	0		
Reset		0	0	0	0	0	0	0	0		
CPU Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W		
Note: R/W = Rea	d/Write.										
Bit Position	Value	Des	cription								
7	0	I ² C i	nterrupt	is disabl	ed.						
IEN	1	I ² C interrupt is enabled.									
6 ENAB	0	The I ² C bus (SCLx/SDAx) is disabled and all inputs are ignored.									
	1	The I ² C bus (SCLx/SDAx) is enabled.									
5	0	A MASTER Mode START condition is sent.									
STA	1	A MA bus.	ASTER I	Mode ST	ART TR	ANSMIT	conditio	on occurs	s on the		
4	0	AMA	ASTER I	Mode ST	OP con	dition is	sent.				
STP	1	A M/ bus.	ASTERI	Mode ST	OP TRA	NSMIT	conditio	n occurs	on the		
3	0	The	I ² C inter	rupt flag	is not se	et.					
IFLG	1	The I ² C interrupt flag is set.									
2	0	Not /	Acknowl	edge.							
ACK	1	Ackr	owledge	e.							
[1:0]	00	Rese	erved.								

I²C Status Register

The I2Cx_SR register, listed in Table 53, is a Read Only register that contains a 5-bit status code in the five msbs. The three lsbs are always 0. The Read Only I2Cx_SR registers share the same I/O addresses as the Write Only I2Cx_CCR registers.

Table 53. I ² C Status RegistersBit76			(12C0_SF	R = CCh	I2C1_S	R = DC	h)
Bit	7	6	5	4	3	2	1	0
Nata: D - Dead Only								

Note: R = Read Only.



zilog

109

				1	1	1	1	1	1
Reset		1	1	1	1	1	0	0	0
CPU Access R R R R R						R	R	R	R
Note: R = Read Only.							·		
Bit									
Position	Value	Desc	ription						
[7:3]	00000-	5-bit	5-bit l ² C status code.						
STAT	11111								
[2:0]	000	Rese	rved.						

There are 29 possible status codes, listed in Table 54. When the I2Cx_SR register contains the status code F8h, no relevant status information is available, no interrupt is generated and the IFLG bit in the I2Cx_CTL register is not set. All other status codes correspond to a defined state of the I^2C .

(I2C0_SR = CCh, I2C1_SR = DCh) (Continued)

When the device enters each of these states, the corresponding status code appears in this register and the IFLG bit in the I2Cx_CTL register is set. When the IFLG bit is cleared, the status code returns to F8h.

Code	Status
00h	Bus error
08h	START condition transmitted
10h	Repeated START condition transmitted
18h	Address + write bit transmitted, ACK received
20h	Address + write bit transmitted, ACK not received
28h	Data byte transmitted in MASTER mode, ACK received
30h	Data byte transmitted in MASTER mode, ACK not received
38h	Arbitration lost in address or data byte
40h	Address + read bit transmitted, ACK received
48h	Address + read bit transmitted, ACK not received
50h	Data byte received in MASTER mode, ACK transmitted
58h	Data byte received in MASTER mode, NACK transmitted
60h	Slave address + write bit received, ACK transmitted

Table 54. I²C Status Codes

Table 53. I²C Status Registers



Table 54. I²C Status Codes (Continued)

Code	Status
68h	Arbitration lost in address as master, slave address + write bit received, ACK transmitted
70h	General Call address received, ACK transmitted
78h	Arbitration lost in address as master, General Call address received, ACK transmitted
80h	Data byte received after slave address received, ACK transmitted
88h	Data byte received after slave address received, NACK transmitted
90h	Data byte received after General Call received, ACK transmitted
98h	Data byte received after General Call received, NACK transmitted
A0h	STOP or repeated START condition received in SLAVE mode
A8h	Slave address + read bit received, ACK transmitted
B0h	Arbitration lost in address as master, slave address + read bit received, ACK transmitted
B8h	Data byte transmitted in SLAVE mode, ACK received
C0h	Data byte transmitted in SLAVE mode, ACK not received
C8h	Last byte transmitted in SLAVE mode, ACK received
D0h	Second Address byte + write bit transmitted, ACK received
D8h	Second Address byte + write bit transmitted, ACK not received
F8h	No relevant status information, IFLG = 0

If an illegal condition occurs on the I^2C bus, the bus error state is entered (status code 00h). To recover from this state, the STP bit in the I2Cx_CTL register must be set and the IFLG bit cleared. The I^2C then returns to the IDLE state. No STOP condition is transmitted on the I^2C bus.



Note: The STP and STA bits may be simultaneously set to 1 to recover from the bus error. The I^2C then sends a START.

I²C Clock Control Register

The I2Cx_CCR register, listed in Table 55 on page 111, is a Write Only register. The seven least significant byte (LSB)s control the frequency at which the I²C bus is sampled and the frequency of the I²C clock line (SCL) when the I²C is operating in MASTER mode. The Write Only I2Cx_CCR registers share the same I/O addresses as the Read Only I2Cx_SR registers.

zilog ₁₁.

Table 55. I ² C Clock Control Registers						(I2C0_CCR = CCh, I2C1_CCR = DCh				
Bit		7	6	5	4	3	2	1	0	
Reset		0	0	0	0	0	0	0	0	
CPU Access		W	W	W	W	W	W	W	W	
Note: W = Re	ead Only.									
Bit Position	Value	Desc	cription							
7	0	Rese	erved.							
[6:3] M	0000– 1111	l ² C c	I ² C clock divider scalar value.							
[2:0] N	000–111	l ² C c	² C clock divider exponent.							

The I²C clocks are derived by the eZ80190 device's system clock, which provides a frequency of f_{sclk} . The I²C bus is sampled by the I²C block at the frequency f_{samp} in the following equation.

$$f_{SAMP} = \frac{f_{SCLK}}{2^N}$$

In MASTER mode, the I²C clock output frequency on SCLx (f_{scl}) is provided by:

$$f_{SCL} = \frac{f_{SCLK}}{10 \text{ x (M+1) x } 2^{N}}$$

The use of two separately-programmable dividers allows the MASTER mode output frequency to be set independently of the frequency at which the I²C bus is sampled. These dividers are particularly useful in multimaster systems because the I²C bus sampling frequency must be at least 10 times the frequency of the fastest master on the bus to ensure that START and STOP conditions are always detected. By using two programmable clock divider stages, a high sampling frequency can be ensured, while allowing the MASTER mode output to be set to a lower frequency.

Bus Clock Speed

The I²C bus is defined for bus clock speeds up to 100 kbps (400 kbps in FAST mode).

zilog ₁₁

To ensure correct detection of START and STOP conditions on the bus, the I^2C must sample the I^2C bus at least ten times faster than the bus clock speed of the fastest master on the bus. The sampling frequency should therefore be at least 1 MHz (4 MHz in FAST mode) to guarantee correct operation with other bus masters.

The I²C sampling frequency is determined by the frequency of the eZ80190 device system clock and the value in the I2Cx_CCR bits 2 to 0. The bus clock speed generated by the I²C in MASTER mode is determined by the frequency of the input clock and the values in I2Cx_CCR[2:0] and I2Cx_CCR[6:3].

I²C Software Reset Register

The I2Cx_SRR register, listed in Table 56 on page 112, is a Write Only register. Writing any value to this register will perform a software reset of the I²C module.

Table 56. I ² 0	(I2C0_SRR = CDh, I2C1_SRR = DDh)								
Bit	7	6	5	4	3	2	1	0	
Reset	Х	Х	Х	Х	Х	Х	Х	Х	
CPU Acces	s W	W	W	W	W	W	W	W	
Note: W = W	rite Only.								
Bit Position	Value	Descripti	on						
[7:0] SRR	00h–FFh	Writing ar I ² C modu	Writing any value to this register performs a software reset of the I ² C module.						



Multiply-Accumulator

MACC Overview

The most significant process in digital signal processing is the Multiply-Accumulate (MACC) function, which forms a sum of products, as the following equation shows.

$$n \\ \sum_{i} x_{i} \times y_{i}$$

where x and y are vectors (tables of values, one-dimensional arrays) in memory.

The MACC block on the eZ80190 device performs DSP functions without incurring the control overhead costs associated with a separate DSP.

The key features of MACC block include:

- Two 40-bit accumulators
- A 16-bit x 16-bit multiplier with a 32-bit product
 - The 32-bit output is added to the value stored in 1 of the 2 available 40-bit accumulators
 - The 40-bit sum is written back to the selected 40-bit accumulator
- Each multiply-accumulate operation completes in a single clock cycle
- Two 256 x 16 dual-port RAM spaces labeled x and y
 - One port of each RAM is 16-bit Read Only and feeds the multiplier
 - The second port is 8-bit Read/Write and is connected to the CPU data bus, allowing the dual RAM to be part of the CPU memory space
- A set of control registers in the CPU's I/O space are used to set up the next multiplyaccumulate operation, initiate the operation, determine when the Multiply-Accumulator completes the current calculation, and retrieve the result

Figure 21 on page 114 displays a simplified block diagram of the Multiply-Accumulator.







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.





Figure 22. Simplified MACC Status Progression

Software Control of the MACC

The Multiply-Accumulator is designed so that CPU software can set up a calculation by writing to the MACC registers using a single OTI2R block output instruction. For details refer to $eZ80^{\ensuremath{\mathbb{R}}}$ CPU User Manual (UM0077). Depending upon the calculation required, this calculation may require writing to all of the MACC control registers, or just a partial subset.

Similarly, the MACC is designed so that eZ80[®] CPU software can read the results of a calculation from the MACC Accumulator registers using a single INI2R block input

zilog[®] | 11

instruction. Depending upon the number of bytes of result required, the INI2R instruction can read all 5 of the MACC_ACx registers or as few as 1.

The Multiply-Accumulator decodes its I/O addresses from ADDR[7:0]. In addition, it monitors ADDR[15:8] to detect the final transfer of a block using the OTI2R or INI2R instructions. These instructions drive the value in the CPU's B register onto ADDR[15:8] and the value in the CPU's C register onto ADDR[7:0]. The B register decrements after completion of each transfer in the OTI2R or INI2R block transfer. The C register increments after completion of each transfer. The final transfer occurs when B contains the value of 01h. After this final transfer, B decrements to 0 and the block instruction terminates. For more information on these CPU instructions and CPU registers refer to $eZ80^{\mathbb{R}}$ *CPU User Manual (UM0077)*.

Defining a New Calculation as READY

When writing a new calculation to the MACC control registers, any of the following actions change the state of the DATA bank from EMPTY to READY:

- A write to MACC_AC4, the MSB of the MACC Accumulator.
- A write to MACC_CTL with ADDR[15:8] = 01h (an OTI2R instruction satisfies this write requirement during its final transfer)
 - This write also clears the MACC Accumulator
- A write to MACC_AC0, MACC_AC1, MACC_AC2, or MACC_AC3 with ADDR[15:8] = 01h (an OTI2R instruction satisfies this write requirement during its final transfer)

If the MACC is prepared to begin a new calculation (CALC bank status is EMPTY or DONE), the banks are immediately swapped as soon as the new calculation defined in the DATA bank is READY. When this swap occurs, the CALC bank status becomes IN PROGRESS.

Defining the DATA Bank as EMPTY

Defining the DATA Bank as EMPTY indicates completion of a result read operation. When reading a result from the MACC Accumulator registers, any of the following actions change the state of the DATA bank from DONE to EMPTY:

- A read from MACC_AC4, the MSB of the MACC Accumulator
- A read from MACC_AC0, MACC_AC1, MACC_AC2, or MACC_AC3 with ADDR[15:8] = 01h, as in the final transfer, using an INI2R instruction

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.



Table 57. MACC DATA Bank Status Codes (Continued)

DATA Bank Status MACC_STAT[1:0]	Description
01b	The DATA bank is READY. Calculation is prepared for execution as soon as the MACC is ready to begin a new calculation.
10b	Invalid.
11b	The DATA bank is DONE. The DATA bank accumulator registers contain the result from a recently completed calculation. This result is not yet read by the CPU.

Table 58. MACC CALC Bank Status Codes

CALC Bank Status MACC_STAT[3:2]	Description
00b	The CALC bank is EMPTY. No calculation is set up for execution.
01b	Invalid.
10b	The CALC bank is IN PROGRESS. MACC is currently executing the operation defined by the CALC bank control registers.
11b	The CALC bank is DONE. The MACC has completed execution of the operations defined by the CALC bank control registers. The result is stored in the CALC bank accumulator registers. The CALC bank must be swapped with the DATA bank to allow the CPU to access the result. Depending upon the full MACC status, this swap may occur automatically.

The combination of possible status values for the DATA and CALC banks define operating states for the MACC. The MACC progresses between states as new calculations are defined, running calculations are completed, results are read, etc. All possible MACC states, the next states possible, and the operation that causes such a state transition are listed in Table 59 on page 119.

zilog ₁₁₉

Cur	rent State		Ne	xt State
DATA Bank	CALC Bank	Operation	DATA Bank	CALC Bank
EMPTY	EMPTY	 Define a new calculation by loading the MACC control registers using the OTI2R instruction. When the OTI2R instruction completes and the final write is to either MACC_CTL or any byte of MACC_ACx, the banks swap and a new calculation begins. The CALC bank status changes from EMPTY to IN PROGRESS as it begins the new calculation. Any write to MACC_AC4 produces the same effect. 	EMPTY	IN PROGRESS
EMPTY	IN PROGRESS	 Define a new calculation by loading the MACC control registers using the OTI2R instruction. When the OTI2R instruction completes and the last write is to either MACC_CTL or any byte of MACC_ACx, the DATA bank state changes from EMPTY to READY. This change in status indicates the DATA bank contains a new calculation that is ready to execute as soon as the MACC completes its current calculation. Any write to MACC_AC4 produces the same effect. 	READY	IN PROGRESS
EMPTY	IN PROGRESS	If the MACC completes execution of the current calculation, the CALC bank status changes from IN PROGRESS to DONE.	EMPTY	DONE
EMPTY	DONE	Write a value of 80h to the MACC_STAT register to force a swap of the CALC and DATA banks. The CALC bank status is now EMPTY. The DATA bank status changes to DONE indicating that it now holds the result from the most recent MACC calculation.	DONE	EMPTY

Table 59. State Progression of the MACC During Operation

zilog[°]

120

Curi	rent State		Next State		
DATA Bank	CALC Bank	Operation	DATA Bank	CALC Bank	
EMPTY	DONE	 Define a new calculation by loading the MACC control registers using the OTI2R instruction. When the OTI2R instruction completes and the last write is to either MACC_CTL or any byte of MACC_ACx, the banks swap and a new calculation begins. The CALC bank status changes from DONE to IN PROGRESS as it begins the new calculation. The DATA bank status changes from EMPTY to DONE as it now contains the result of the previous calculation. Any write to MACC_AC4 produces the same effect. 	DONE	IN PROGRESS	
READY	IN PROGRESS	When the MACC completes execution of the current calculation, the banks swap. The DATA bank status changes to DONE to indicate the availability of the just completed calculation. The MACC begins the new calculation so the CALC bank status remains IN PROGRESS.	DONE	IN PROGRESS	
DONE	EMPTY	 Read the result from the MACC Accumulator registers using the INI2R instruction. When the INI2R instruction completes and the last read is from any byte of MACC_ACx, the DATA bank status changes from DONE to EMPTY. Any read from MACC_AC4 produces the same effect. Any write to any MACC register except for MAC_STAT produces the same effect. 	EMPTY	EMPTY	

Table 59. State Progression of the MACC During Operation (Continued)

zilog

Cur	rent State		Ne	ext State
DATA Bank	CALC Bank	Operation	DATA Bank	CALC Bank
DONE	IN PROGRESS	If the MACC completes execution of the current calculation, the CALC bank status changes from IN PROGRESS to DONE.	DONE	DONE
DONE	IN PROGRESS	 Read the result from the MACC Accumulator registers using the INI2R instruction. When the INI2R instruction completes and the last read is from any byte of MACC_ACx, the DATA bank status changes from DONE to EMPTY. Any read from MACC_AC4 produces the same effect. Any write to any MACC register except for MAC_STAT produces the same effect. 	EMPTY	IN PROGRESS
DONE	DONE	 Read the result from the MACC Accumulator registers using the INI2R instruction. When the INI2R instruction completes and the last read is from any byte of MACC_ACx, the DATA bank status changes from DONE to EMPTY. Any read from MACC_AC4 produces the same effect. Any write to any MACC register except for MAC_STAT produces the same effect. 	EMPTY	DONE

Table 59. State Progression of the MACC During Operation (Continued)

IN_SHIFT and OUT_SHIFT

The Multiply-Accumulator on the eZ80190 device features two additional functions, IN_SHIFT and OUT_SHIFT, that can be useful in many DSP operations. Both of these optional functions are controlled by the MACC Control register, MACC_CTL.

IN_SHIFT Function

The IN_SHIFT field, bits 2:0 of the MACC_CTL register, defines the magnitude of the left-shift that is performed when the CPU writes a starting value to the MACC Accumulator registers MACC_AC0, MACC_AC1, MACC_AC2, MACC_AC3, and MACC_AC4. The MACC automatically handles the shift of the 40-bit value as it is written as a succession of 8-bit values. The writes can be left-shifted 0 to 7 bits depending upon the value of IN_SHIFT. The NOISE field, bit 6 of the MACC_CTL register, sets the value used to fill the lsbs vacated during the left-shift operation.



Example 1—When IN_SHIFT = 000b, writes to the MACC Accumulator registers are not shifted. If the MACC Accumulator is loaded with a 40-bit value using a succession of 8-bit writes, the procedure appears as follows:

- Write the LSB to the MACC Accumulator MACC Accumulator [7:0] = MACC_AC0[7:0] = DATA_IN[7:0]
- Write the second byte to the MACC Accumulator MACC Accumulator [15:8] = MACC AC1[7:0] = DATA IN[7:0]
- Write the third byte to the MACC Accumulator MACC Accumulator [23:16] = MACC_AC2[7:0] = DATA_IN[7:0]
- Write the fourth byte to the MACC Accumulator
 MACC Accumulator [31:24] = MACC_AC3[7:0] = DATA_IN[7:0]
- 5. Write the MSB to the MACC Accumulator

MACC Accumulator [39:32] = MACC_AC4[7:0] = DATA_IN[7:0]

Example 2—When IN_SHIFT = 011b and NOISE = 1, writes to the MACC Accumulator registers are left-shifted by 3 bits. The 3 lsbs are filled with a NOISE value of 1. If the MACC Accumulator is loaded with a 40-bit value using a succession of 8-bit writes, the procedure appears as follows:

- Write the LSB to the MACC Accumulator MACC Accumulator [10:0] = {MACC_AC0[7:0], 111b} = {DATA_IN[7:0], 111b}
- Write the second byte to the MACC Accumulator MACC Accumulator [18:11] = MACC AC1[7:0] = DATA IN[7:0]
- Write the third byte to the MACC Accumulator
 MACC Accumulator [26:19] = MACC_AC2[7:0] = DATA_IN[7:0]
- Write the fourth byte to the MACC Accumulator MACC Accumulator [34:27] = MACC_AC3[7:0] = DATA_IN[7:0]
- Write the MSB to the MACC Accumulator MACC Accumulator [39:35] = MACC_AC4[4:0] = DATA_IN[4:0]

In Example 2, notice that the upper 3 bits of the final write are ignored.

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]



- Read the fourth byte from the MACC Accumulator
 DATA OUT[7:0] = MACC AC3[7:0] = MACC Accumulator [34:27]
- 5. Read the MSB from the MACC Accumulator

DATA_OUT[7:0] = MACC_AC4[7:0] = {000b, MACC Accumulator [39:35]}

In Example 2, notice that the upper 3 bits of the final read contain copies of the sign bit (in this example, the sign bit is 0, which represents a positive number).

Recommended Operation

Setting Up A New Calculation

The following procedure sets up a new calculation.

- 1. Load the data into the MACC's x and y RAM spaces.
- 2. Read the status register, MACC_STAT. If the DATA bank status is EMPTY or DONE, a new calculation can be written to the DATA bank registers. If the DATA bank status is DONE, the result currently available in the MACC Accumulator registers are lost if not read prior to a write.
- 3. Use the OTI2R instruction to load the new calculation. Registers to be written may include nearly any combination of MACC_xSTART, MACC_xEND, MACC_xRELOAD, MACC_vSTART, MACC_yEND, MACC_yRELOAD, MACC_LENGTH, MACC_CTL, and MACC_ACx. If the OTI2R instruction is set up to end with either MACC_CTL or any of the MACC_ACx registers, the DATA bank status changes to READY.
- 4. If the MACC is ready to begin a new calculation (CALC bank is EMPTY or DONE), the banks are automatically switched to begin execution. The equation that is set up in the DATA bank is transferred to the CALC bank. The CALC bank status changes to IN PROGRESS.

Retrieve A Calculation

The following procedure retrieves the results of a calculation.

1. Read the status register. If the Multiply-Accumulator has not completed the previous calculation provided, the application must wait until the Multiply-Accumulator completes the calculation, at which time the CALC bank status changes to DONE.

zilog |

125

- 2. If the DATA bank status is EMPTY and the CALC bank status is DONE, write 80h to the status register. As a result, the register banks are swapped so that the DATA status becomes DONE.
- 3. If both status fields indicate EMPTY, there is no result to retrieve.
- 4. If the DATA bank status is DONE, the application reads as many of the MAC_AC0-3 registers as desired. Because the Multiply-Accumulator decodes the A15:8 lines to determine when a transfer is complete, this register READ can be initiated with an INI2R instruction. Reading the final byte of the result changes the DATA bank status to EMPTY unless there is another result to retrieve. If such is the case, the CALC bank status changes to EMPTY and the DATA bank status changes to DONE.

MACC RAM

The eZ80190 device features 1KB of dual-port RAM available for use with the Multiply-Accumulator, as displayed in Figure 23 on page 126. From the CPU, MACC RAM appears as a 1KB block of 8-bit RAM. To the Multiply-Accumulator, MACC RAM appears as two blocks of 256x 16-bit RAM. The CPU provides Read/Write access to one port of the MACC RAM. The Multiply-Accumulator provides Read Only access to the second port of the MACC RAM.

As described in Random Access Memory on page 59, MACC RAM is accessed by the CPU in the memory address space from $\{RAM_ADDR_U[7:0], DC00h\}$ to $\{RAM_ADDR_U[7:0], DFFFh\}$. The upper byte of the MACC RAM address is received from the RAM Address Upper Byte register, RAM_ADDR_U. The MACC X data is stored in the lower 512 bytes of the MACC RAM memory address space from DC00h to DDFFh. The MACC y data is stored in the upper 512 bytes of the MACC RAM memory address space from DE00h to DFFFh. The LSB, bits [7:0] of the 16-bit x and y data, is stored in the even memory addresses. The MSB, bits [15:8], are stored in the odd memory addresses. The data in MACC RAM must be stored in two's-complement form.

MACC RAM Address Indexing

For each calculation that the MACC is to perform, the software must arrange the two vectors/arrays to be multiplied and accumulated. One vector must be written to x RAM while the other vector must be written to y RAM. The software then writes values to the MACC control registers to indicate where the x and y data is to be stored for the current calculation. For both x and y data, there are 3 values defining the data location:

1. MACC_xSTART and MACC_ySTART define the address of the first *x* and *y* values to be multiplied together.

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





Figure 24. MACC RAM Address Indexing

Multiply-Accumulator Control And Data Registers

The MACC is divided into two separate operating banks. The CPU can only access the current DATA bank via the control and data registers (described in this section). To access the registers associated with the current CALC bank, the two banks must be swapped.

MACC x DATA Starting Address Register

The MACC_xSTART register, listed in Table 60, defines the starting address for the MACC to read 16-bit values from the *x* DATA for performing its calculations.

Table 60. MACC x DATA	Starting Address Register				(MACC_ <i>x</i> START = E0h)			
Bit	7	6	5	4	3	2	1	0
Note: R/W = Read/Write.								

zilog

128

Table 60. MACC x DATA Starting Address Register (MACC_xSTART = E0h) (Continued) Reset 0 0 0 0 0 0 0 0 R/W **CPU Access** R/W R/W R/W R/W R/W R/W R/W Note: R/W = Read/Write. Bit Position Value Description [7:0] 00h-The starting address for MACC RAM x DATA. MACC_xSTART FFh

MACC x DATA Ending Address Register

The MACC_xEND register, listed in Table 61, defines the ending address for the MACC to read 16-bit values from the *x* DATA for performing its calculations.

Table 61. MACC	x DATA	Endin	g Addre	ess Regi	ster		(MACC_	xEND =	E1h)
Bit		7	6	5	4	3	2	1	0
Reset		0	0	0	0	0	0	0	0
CPU Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Note: R/W = Read	d/Write.								
Bit Position	Value	Desc	cription						
[7:0] MACC_xEND	00h– FFh	The	The ending address for MACC RAM <i>x</i> DATA.						

MACC x DATA Reload Address Register

The MACC_xRELOAD register, listed in Table 62, defines the reload address within the *x* data of MACC RAM. When the *x* data address increments to the value in the MACC_xEND register, the next *x* data address is taken from this MACC_xRELOAD register.

Table 62. MACC X DATA Reload Address Register (MACC_XRELOAD = E2
--

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.								

<mark>z</mark>ilog[°]

129

Bit Position	Value	Description
[7:0] MACC_ <i>x</i> RELOAD	00h– FFh	The reload address for MACC RAM <i>x</i> DATA.

MACC Length Register

The MACC_LENGTH register, listed in Table 63, defines the total number of x and y data pairs that are multiplied and accumulated for the MACC operation.

Table 63. MACC Le		(MACC_LENGTH = E3h)							
Bit	7		6	5	4	3	2	1	0
Reset	0		0	0	0	0	0	0	0
CPU Access	R/\	Ν	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Note: R/W = Read/W	/rite.								
Bit Position	Value	De	escriptio	on					
[7:0] MACC_LENGTH	00h– FFh	Total number of address pairs to be multiplied and accumulated for the current MACC operation.							

MACC y DATA Starting Address Register

The MACC_*y*START register, listed in Table 64, defines the starting address for the MACC to read 16-bit values from the *y* DATA for performing its calculations.

Table 64. MACC y	(MACC_ <i>y</i> START = E4h)								
Bit	7		6	5	4	3	2	1	0
Reset	0		0	0	0	0	0	0	0
CPU Access	R/\	N	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Note: R/W = Read/W	Vrite.								
Bit Position	Value	D	escriptio	on					
[7:0] MACC_ <i>y</i> START	00h– FFh	Starting address for the <i>y</i> DATA of MACC RAM.							

zilog

130

MACC y DATA Ending Address Register

The MACC_yEND register, listed in Table 65 on page 130, defines the ending address for the MACC to read 16-bit values from the *y* DATA for performing its calculations.

Table 65. MACC y DATA Ending Address Register (MACC_yEND = E5h)								E5h)	
Bit	7		6	5	4	3	2	1	0
Reset	0		0	0	0	0	0	0	0
CPU Access	R/\	R/W		R/W	R/W	R/W	R/W	R/W	R/W
Note: R/W = Read/	Write.								
Bit Position	Value	De	escriptio	on					
[7:0] MACC_ <i>y</i> END	00h– FFh	Ending address for the <i>y</i> DATA of MACC RAM.							

MACC y DATA Reload Address Register

The MACC_yRELOAD register, listed in Table 66, defines the reload address within the y DATA of MACC RAM. When the y DATA address increments to the value in MACC_yEND, the next y DATA address is taken from this MACC_yRELOAD register.

Table 66. MACC y [(MACC_yRELOAD = E6h)								
Bit	7		6	5	4	3	2	1	0
Reset	0		0	0	0	0	0	0	0
CPU Access	R/	R/W		R/W	R/W	R/W	R/W	R/W	R/W
Note: R/W = Read/W	rite.								
Bit Position	Value	De	escriptio	on					
[7:0] MACC_ <i>y</i> RELOAD	00h– FFh	Re	eload ad	dress fo	r the y D	ATA of	MACC F	RAM.	

MACC Control Register

The MACC Control register, listed in Table 67, provides added MACC features including interrupt enable on completion of calculation. All writes to this register clear the 40-bit accumulator to 0 (MACC_ACx = 00h}.

zilog

131

Table 67. MA	CC Con	trol Regis	ter		(MACC_CTL = E7h)						
Bit		7	6	5	4	3	2	1	0		
Reset		0	0	0	0	0	0	0	0		
CPU Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W		
Note: R/W = R	Read/Write	Э.									
Bit Position	Value	Descript	ion								
7	0	MACC in	terrupt i	s disable	ed.						
MACC_IE	1	The MAC defined. when it c	The MACC interrupt is enabled for the calculation currently being defined. The MACC generates an interrupt request to the CPU when it completes this calculation (DONE).								
6	0	All NOIS	E bits ad	dded to t	he accu	mulator	using IN	_SHIFT	are 0.		
NOISE	1	All NOIS	E bits ad	dded to t	he accu	mulator	using IN	_SHIFT	are 1.		
[5:3] OUT_SHIFT	000	No right- Accumul DATA_O	shift is p ator regi UT[40:0	erforme isters by] = MAC	d during the CPL C_ACx[READs J. 39:0].	from the	MACC			
	001	Reads fro right-shif DATA_O	om the N ted by 1 UT[40:0	MACC A bit with)] = {2{M	ccumula a fill by t ACC_A0	tor regis he sign Cx[39]}, I	ters by t bit (msb MACC_/	he CPU = bit 39) ACx[38:1	are).]}.		
	010	Reads fro right-shif DATA_O	om the N ted by 2 UT[40:0	MACC A bits with)] = {3{M	ccumula n a fill by ACC_A0	tor regis the sign Cx[39]}, I	ters by t bit (ms MACC_/	he CPU b = bit 39 ACx[38:2	are 9). 2]}.		
	011	Reads fro right-shif DATA_O	om the N ted by 3 UT[40:0	MACC A bits with)] = {4{M	ccumula n a fill by ACC_A0	tor regis the sign Cx[39]}, I	ters by t bit (ms MACC_/	he CPU b = bit 39 ACx[38:3	are 9). 3]}.		
	100	Reads fro right-shif DATA_O	om the N ted by 4 UT[40:0	MACC A bits with)] = {5{M	ccumula n a fill by ACC_AC	tor regis the sign Cx[39]}, I	ters by t bit (ms MACC_/	he CPU b = bit 39 ACx[38:4	are 9). !]}.		
	101	Reads fro right-shif DATA_O	om the N ted by 5 UT[40:0	MACC A bits with)] = {6{M	ccumula n a fill by ACC_AC	tor regis the sign Cx[39]}, I	ters by t bit (ms MACC_/	he CPU b = bit 39 ACx[38:5	are 9). 5]}.		
	110	Reads fro right-shif DATA_O	om the N ted by 6 UT[40:0	MACC A bits with)] = {7{M	ccumula n a fill by ACC_A0	tor regis the sign Cx[39]}, I	ters by t bit (ms MACC_/	he CPU b = bit 39 ACx[38:6	are 9). 6]}.		
	111	Reads fro right-shif DATA_O	om the N ted by 7 UT[40:0	MACC A bits with] = {8{M	ccumula n a fill by ACC_A0	tor regis the sign Cx[39]}, I	ters by t bit (ms MACC_/	he CPU b = bit 39 ACx[38:7	are 9). 7]}.		



1	32
	~

Bit	Value	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 Ad		(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

133

Bit Position	Value	Description
[7:0] MACC_AC0	00h– FFh	MACC Accumulator bits 7:0.

MACC Accumulator Byte 1 Register

The MACC_AC1 register, listed in Table 69, contains bits 15:8 of the 40-bit MACC Accumulator.

Table 69. MACC		(MACC_AC1 = E9h)						(MACC_AC1 = E9h)				
Bit	7		6	5	4	3	2	1	0			
Reset	Х	<u> </u>	Х	Х	Х	Х	Х	Х	Х			
CPU Access	R/	N	R/W	R/W	R/W	R/W	R/W	R/W	R/W			
Note: X = Undefine	ed; R/W = Re	ead/	Write.									
Bit Position	Value	D	escriptio	าท								
[7·0]	00h-	M	MACC Accumulator bits 15:8									
MACC AC1	FFh											

MACC Accumulator Byte 2 Register

The MACC_AC2 register, listed in Table 70, contains bits 23:16 of the 40-bit MACC Accumulator.

Table 70. MACC	(MACC_AC2 = EAh)									
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 = Undefine	ed; R/W = Re	ead/	Write.							
Bit Position	Value	De	escriptio	on						
[7:0] MACC_AC2	00h– FFh	M	MACC Accumulator bits 23:16.							



MACC Accumulator Byte 3 Register

The MACC_AC3 register, listed in Table 71, contains bits 31:24 of the 40-bit MACC Accumulator.

Table 71. MACC Accumulator Byte 3 Register						(MACC_AC3 = EBh)					
Bit	7		6	5	4	3	2	1	0		
Reset	Х	<u> </u>	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 = Undefine	ed; R/W = Re	ead/	Write.								
Bit											
Position	Value	De	Description								
[7:0] MACC_AC3	00h– FFh	M	MACC Accumulator bits 31:24.								

MACC Accumulator Byte 4 Register

The MACC_AC4 register contains the MSB (bits 39:32) of the 40-bit MACC Accumulator. Reading this register changes the status of the DATA bank to EMPTY. Also, if the CALC bank status is DONE, reading this register swaps the banks. In this case, the ending status of the DATA bank is DONE while the CALC bank is EMPTY.

Writing to the MACC_AC4 register, listed in Table 72, changes the status of the DATA bank from EMPTY to READY. If the MACC is ready to begin a new calculation, the banks are swapped and the new calculation begins (CALC bank status becomes IN PROGRESS).

Table 72. MACC	Accumulat	or l	Byte 4 F	Register		(MACC_AC4 = ECh)					
Bit	7		6	5	4	3	2	1	0		
Reset	Х		Х	Х	Х	Х	Х	Х	Х		
CPU Access	R/\	Ν	R/W	R/W	R/W	R/W	R/W	R/W	R/W		
Note: X = Undefine	ed; R/W = Re	ead/	Write.								
Bit Position	Value	De	escriptio	on							
[7:0] MACC_AC4	00h– FFh	M	MACC Accumulator bits 39:32.								
zilog

135

MACC Status Register

The MACC STAT register, listed in Table 73, reflects the current status of the Multiply-Accumulator. Writing a value of 80h to the MACC STAT register when the CALC bank has completed its calculation (DONE) and the DATA register is not loaded with a new calculation (EMPTY) swaps the banks to allow the pending result to be retrieved.

The eZ80190 device uses two distinct numbered banks, banks 0 and 1. The value in bit 4 of the MACC STAT register indicates which of these two banks is currently accessible as the DATA bank. In general, there is no requirement for software to monitor which numbered bank is currently the DATA bank and which is the CALC bank.

Table 73. MACC		(MACC_STAT = EDh)								
Bit	7	6	5	4	3	2	1	0		
Reset	0	0	0	0	0	0	0	0		
CPU Access	W	/ R	R	R	R	R	R	R		
Note: R = Read; V	v = Write.									
Bit Position	Value	Descripti	on							
[7:5]	000	Reserved	Reserved.							
4 BANK	0 The current DATA bank is Bank 0. The current DAT is always reset to Bank 0 when both banks are EM							A bank PTY.		
	1	The current CALC bank is Bank 1.								
[3:2] CALC_STAT	00	The CALC execution	C bank is	EMPTY	′. No cal	culation	is set up	for		
	01	Invalid.								
	10	The CALC bank is IN PROGRESS. The MACC is currer executing on a data set.						urrently		
	11	The CALC bank is DONE. The MACC has completed execution of the operations defined by the CALC bank control registers. The result is stored in the CALC bank accumulator registers. The CALC bank must be swapp with the DATA bank to allow the CPLL to access the res						ed ank ank apped e result.		



136

Bit		
Position	Value	Description
[1:0] DATA_STAT	00	The DATA bank is EMPTY. No calculation is set up for execution.
	01	The DATA bank is READY. The calculation is prepared for execution.
	10	Invalid.
	11	The DATA bank is DONE. The DATA bank accumulator registers contain the result from a recently completed calculation. This result is not yet read by the CPU.

zilog[®]

Interrupt Controller

The interrupt controller on the eZ80190 device routes the interrupt request signals from the internal peripherals and external devices (via the internal port I/O) to the eZ80[®] CPU. On the eZ80190 device, all interrupts use the CPU's vectored interrupt function. Table 74 lists the vector for each of the interrupt sources. The interrupt sources are listed in order of their priority, with vector 00h being the highest-priority interrupt.

Vector	Source	Vector	Source	Vector	Source	Vector	Source
00h	MACC	18h	Port A 1	30h	Port B 5	48h	Port D 1
02h	DMA 0	1Ah	Port A 2	32h	Port B 6	4Ah	Port D 2
04h	DMA 1	1Ch	Port A 3	34h	Port B 7	4Ch	Port D 3
06h	PRT 0	1Eh	Port A 4	36h	Port C 0	4Eh	Port D 4
08h	PRT 1	20h	Port A 5	38h	Port C 1	50h	Port D 5
0Ah	PRT 2	22h	Port A 6	3Ah	Port C 2	52h	Port D 6
0Ch	PRT 3	24h	Port A 7	3Ch	Port C 3	54h	Port D 7
0Eh	PRT 4	26h	Port B 0	3Eh	Port C 4	56h	Reserved
10h	PRT 5	28h	Port B 1	40h	Port C 5	58h	Reserved
12h	UZI 0	2Ah	Port B 2	42h	Port C 6	5Ah	Reserved
14h	UZI 1	2Ch	Port B 3	44h	Port C 7	5Ch	Reserved
16h	Port A 0	2Eh	Port B 4	46h	Port D 0	5Eh	Reserved

Table 74. Interrupt Vector Sources by Priority

When any one or more of the interrupt requests (IRQs) become active, an interrupt request is generated by the interrupt controller and sent to the CPU. The corresponding 8-bit interrupt vector for the highest priority interrupt is placed on the 8-bit interrupt vector bus, IVECT[7:0]. The interrupt vector bus is internal to the eZ80190 device and is therefore not visible externally. The response time of the CPU to an interrupt request is a function of the current instruction being executed as well as the number of WAIT states being inserted. The interrupt vector, {I[7:0], IVECT[7:0]}, is visible on the address bus, ADDR[16:0], when the Interrupt Service Routine (ISR) begins. The response of the CPU to a vectored interrupt on the eZ80190 device is listed in Table 75. The eZ80190 device does not support eZ80 Mode 0, Mode 1, or Mode 2 interrupts. Interrupt sources are required to be active until the ISR starts.



138

Table 75. Vectored Interrupt Operation

Memory Mode	ADL Bit	MADL Bit	Operation
Z80 [®] Mode	0	0	 Read the LSB of the interrupt vector placed on the internal vectored interrupt bus, IVECT [7:0], by the interrupting peripheral. IEF1 ← 0 IEF2 ← 0 The Starting Program Counter is effectively {MBASE, PC[15:0]}. Push the 2-byte return address PC[15:0] onto the ({MBASE,SPS}) stack. The ADL mode bit remains cleared to 0. The interrupt vector address is {MBASE, I[7:0], IVECT[7:0]}. PC[15:0] ← ({MBASE, I[7:0], IVECT[7:0]}). The Ending Program Counter is effectively {MBASE, PC[15:0]} = ({MBASE, I[7:0], IVECT[7:0]}) The interrupt service routine must end with RETI.
ADL Mode	1	0	 Read the LSB of the interrupt vector placed on the internal vectored interrupt bus, IVECT [7:0], by the interrupting peripheral. IEF1 ← 0 IEF2 ← 0 The Starting Program Counter is PC[23:0]. Push the 3-byte return address, PC[23:0], onto the SPL stack. The ADL mode bit remains set to 1. The interrupt vector address is {00h, I[7:0], IVECT[7:0]}. PC[23:0] ← ({00h, I[7:0], IVECT[7:0]}). The Ending Program Counter is PC[23:0] = ({00h, I[7:0], IVECT[7:0]}). The interrupt service routine must end with RETI.



139

Table 75. Vectored Interrupt Operation (Continued)

Z80 [®] Mode	0	1	 Read the LSB of the interrupt vector placed on the internal vectored interrupt bus, IVECT[7:0], bus by the interrupting peripheral. IEF1 ← 0 IEF2 ← 0 The Starting Program Counter is effectively {MBASE, PC[15:0]}. Push the 2-byte return address, PC[15:0], onto the SPL stack. Push a 02h byte onto the SPL stack to indicate an interrupt from Z80 mode (because ADL = 0). Set the ADL mode bit to 1. The interrupt vector address is {00h, I[7:0], IVECT[7:0]}. PC[23:0] ← ({00h, I[7:0], IVECT[7:0]}). The Ending Program Counter is PC[23:0] = ({00h, I[7:0], IVECT[7:0]}). The interrupt service routine must end with RETI.L
ADL Mode	1	1	 Read the LSB of the interrupt vector placed on the internal vectored interrupt bus, IVECT [7:0], by the interrupting peripheral. IEF1 ← 0 IEF2 ← 0 The Starting Program Counter is PC[23:0]. Push the 3-byte return address, PC[23:0], onto the SPL stack. Push a 03h byte onto the SPL stack to indicate an interrupt from ADL mode (because ADL = 1). The ADL mode bit remains set to 1. The interrupt vector address is {00h, I[7:0], IVECT[7:0]}. PC[23:0] ← ({00h, I[7:0], IVECT[7:0]}). The Ending Program Counter is PC[23:0] = ({00h, I[7:0], IVECT[7:0]}). The interrupt service routine must end with RETI.L

zilog

140

Direct Memory Access Controller

The eZ80190 device features two Direct Memory Access (DMA) channels. The DMA controller can be used for direct memory to memory data transfers without CPU intervention. There are two DMA channels, Channel 0 and Channel 1, each featuring independent control registers. Transfers can be either in BURST mode or CYCLE-STEAL mode.

In BURST mode, after the DMA controller gains access to the bus, it maintains control of the bus until the block data transfer is complete for that channel. In CYCLE-STEAL mode, after the DMA gains access to the bus, it transfers only one byte and then returns control of the bus to the CPU for eight clock cycles. The DMA then again requests the bus and gains access to transfer the next byte. This process continues until the programmed number of bytes are transferred.



The DMA channel cannot be used to transfer data to or from internal I/O registers. However, it can be used with external memory-mapped I/O devices.

DMA Programming

There are 18 registers that control DMA operation—nine control registers for DMA channel 0 operation and nine control registers for DMA channel 1 operation. In each channel, there are three registers for the 24-bit data transfer source address, three registers for the 24-bit data transfer destination address, two registers for the 16-bit byte count, and one register for DMA channel control.

If the DMA channel is enabled, it can take control of the system buses—ADDR[23:0], DATA[7:0], $\overline{\text{RD}}$, and $\overline{\text{WR}}$ —and direct the transfer of data between memory locations. If the DMA channel is disabled, the DMA cannot initiate bus requests nor transfer data. The DMA is always disabled after RESET. External DMA master devices can force the eZ80190 device to release the bus for their use by driving the $\overline{\text{BUSREQ}}$ pin Low. To the eZ80190 CPU, this bus request signal operates the same as if it had originated from the internal DMA controllers. If both of these signals should occur simultaneously, the internal DMA bus request will hold a higher priority than a request from an external bus master device.

To configure the DMA registers for memory transfer, the Source and Destination address registers must be programmed. The byte count registers must be programmed with the number of bytes to be transferred. The DMA Control register must be programmed to select whether the Source and Destination address registers are incremented, decremented, or remain fixed during a transfer, whether the DMA outputs an interrupt when finished, and what data transfer mode the DMA employs. Finally, the DMA channel must be enabled to allow transfers to begin.



DMA Transfer Modes

There are two modes of operation for the DMA channels. The DMA can transfer data in BURST mode or CYCLE-STEAL mode. The data transfer mode is controlled by the BURST bit in the DMA Control registers (DMAx_CTL[4]).

In BURST mode, the DMA controller takes control of the bus within the eZ80190 device for the entire time period required to complete the data transfer. The CPU is idled while the DMA controller completes its BURST mode data transfer.

The default operation for the DMA controller is CYCLE-STEAL mode in which the DMA controller requests and then gains access to the bus for the transfer of only one byte at a time. After the transfer of each byte, the DMA returns control of the bus back to the CPU. The DMA then waits for the CPU to complete 8 clock cycles before again requesting control of the bus. As a result, other activities can proceed while the DMA is transferring data in the background. CYCLE-STEAL mode slows down the processing of the main program task of the CPU. See Figure 25.



Figure 25. DMA CYCLE-STEAL Timing

zilog ₁₄₂

DMA Channel Priorities

In all operating mode combinations, DMA Channel 0 is prioritized higher than DMA Channel 1. If Channel 0 is configured for BURST mode operation, Channel 0 completes its entire block transfer before Channel 1 begins its transfer.

When both channels are configured for CYCLE-STEAL mode, the 2 DMA channels alternate stealing execution cycles from the CPU. First, DMA Channel 0 performs a cycle-steal single-byte transfer then releases the bus to the CPU for the next 8 clock cycles. Then, DMA channel 1 requests the bus and gains access to pass one of its bytes. After DMA channel 1 completes the transfer of its byte, control is returned to the CPU for another 8 clock cycles. This process repeats until one or both of the DMA channels complete the transfer of all required bytes.

If DMA channel 0 is programmed in CYCLE-STEAL mode and DMA channel 1 is programmed in BURST mode, DMA channel 1 is not allowed to transfer its data until DMA channel 0 completes its entire transfer.

DMA Interrupts

Each DMA controller can generate an interrupt request to the CPU when its memory transfer is complete. The DMA interrupts are enabled by setting bit 6 in the DMA Control register (either DMA0_CTL or DMA1_CTL) to 1. The default operation is for the DMA interrupts to be disabled. Each DMA channel is capable of generating an interrupt when its 16-bit data byte transfer counter register reaches its terminal count of 0000h. The interrupts are cleared by resetting the DMA_EN bit field in the DMA Control registers to disable the DMA channel that is generating the input. Clearing the interrupt enable bit (DMAx_CTL[6] = IRQ_DMA) does not clear the interrupt to the CPU after it is set.

DMA Control Registers

Table 76 lists the control registers used by the DMA controller. These registers are accessed by the CPU using I/O instructions.

Name	Description	CPU Access	Reset Value	Register Address
DMA0_SAR_L	DMA0 Source Address Low Byte register	R/W	XX	EEh
DMA0_SAR_H	DMA0 Source Address High Byte register	R/W	XX	EFh
DMA0_SAR_U	DMA0 Source Address Upper Byte register	R/W	XX	F0h

Table 76. DMA Registers

zilog

Name	Description	CPU Access	Reset Value	Register Address
DMA0_DAR_L	DMA0 Destination Address Low Byte register	R/W	XX	F1h
DMA0_DAR_H	DMA0 Destination Address High Byte register	R/W	XX	F2h
DMA0_DAR_U	DMA0 Destination Address Upper Byte register	R/W	XX	F3h
DMA0_BC_L	DMA0 Byte Count Low Byte register	R/W	00h	F4h
DMA0_BC_H	DMA0 Byte Count High Byte register	R/W	00h	F5h
DMA0_CTL	DMA0 Control register	R/W	00h	F6h
DMA1_SAR_L	DMA1 Source Address Low Byte register	R/W	XX	F7h
DMA1_SAR_H	DMA1 Source Address High Byte register	R/W	XX	F8h
DMA1_SAR_U	DMA1 Source Address Upper Byte register	R/W	XX	F9h
DMA1_DAR_L	DMA1 Destination Address Low Byte register	R/W	XX	FAh
DMA1_DAR_H	DMA1 Destination Address High Byte register	R/W	XX	FBh
DMA1_DAR_U	DMA1 Destination Address Upper Byte register	R/W	XX	FCh
DMA1_BC_L	DMA1 Byte Count Low Byte register	R/W	00h	FDh
DMA1_BC_H	DMA1 Byte Count High Byte register	R/W	00h	FEh
DMA1_CTL	DMA1 Control register	R/W	00h	FFh

Table 76. DMA Registers (Continued)

DMA Source Address Registers

These two groups of registers hold the 24-bit addresses of the source memory location for DMA Channel 0 and Channel 1. Depending upon settings within the DMA Control registers' SARx_CTL fields, the 24-bit address values can automatically be incremented, decremented, or unchanged following the transfer of each byte of data. See Table 77.

Table 77. DMA Source Address RegistersDMA0_SAR_L = EEh, DMA0_SAR_H = EFh,DMA0_SAR_U = F0h, DMA1_SAR_L = F7h, DMA1_SAR_H = F8h, DMA1_SAR_U = F9h

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.									

zilog .

Bit Position	Value	Description
[7:0] DMAx_SAR_L, DMAx_SAR_H, or DMAx_SAR_U	00h– FFh	The 2 sets of DMA Source address registers contain the memory location addresses for the source of the data transfer. The 24-bit addresses are returned by {DMAx_SAR_U, DMAx_SAR_H, DMAx_SAR_L}, where x is either 0 or 1.

DMA Destination Address Registers

This group of registers holds the 24-bit address of the current destination memory location. Depending upon settings within the DMA Control registers' DMA_CTL fields, the 24-bit address values can automatically be incremented, decremented, or unchanged following transfer of each byte of data. See Table 78.

Table 78. DMA Destination Address Registers DMA0_DAR_L = F1h, DMA0_DAR_H = F2h, DMA0_DAR_U = F3h, DMA1_DAR_L = FAh, DMA1_DAR_H = FBh, DMA1_DAR_U = FCh

Bit	7	6	5	4	3	2	1	0
Reset	Х	Х	Х	Х	Х	Х	Х	Х
CPU Access	R/	N R/W	R/W	R/W	R/W	R/W	R/W	R/W
Note: X = Undefined	l; R/W = Re	ead/Write.						
Bit Position	Value	Descripti	on					
[7:0] DMAX_DAR_L, DMAX_DAR_H, or DMAX_DAR_U	00h– FFh	The 2 sets the memo data trans {DMAx_D is either 0	s of DMA ory location fer. The AR_U, D or 1.	A Destina on addre 24-bit ao MAx_D	ation add esses for ddresses AR_H, D	tress reg the des are ret MAx_D	gisters co tination urned by AR_L} w	ontain of the / /here x

DMA Byte Count Registers

The two pairs of DMA Byte Count registers, listed in Table 79 on page 145, contain the number of bytes to be transferred by the DMA channels. The 16-bit value, {DMAx_BC_H, DMAx_BC_L}, is decremented after each transfer. The DMA transfer is complete when the value decrements to 0000h. One to 65535 bytes can be transferred.

zilog ₁₄₅

Table 79. DMA Byte Count Registers DMA0_BC_L = F4h, DMA0_BC_H = F5h, DMA1_BC_L = FDh, DMA1_BC_H = FEh

Bit	7	6	5	4	3	2	1	0
Reset	0	0	0	0	0	0	0	0
CPU Access	R/\	W R/W	R/W	R/W	R/W	R/W	R/W	R/W
Note: R/W = Read	/Write.							
Bit Position	Value	Descript	ion					
[7:0] DMAX_BC_L or DMAX_BC_H	00h– FFh	The 2 pai number o operation {DMAx_E	rs of DM of bytes to . The 16- BC_H, DN	A Byte C b be tran bit byte //Ax_BC	Count reg sferred (count va _L}, whe	gisters co during th alues are ere x is e	ontain th le currer e returne ither 0 o	e it d by r 1.

DMA Control Registers

Table 80 lists the control registers used by the DMA controller. These registers are accessed by the CPU using I/O instructions.

Table 80. DMA Control	(DMA0_CTL = F6h, DMA1_CTL = FFh)							
Bit	7	6	5	4	3	2	1	0
Reset	0	0	0	0	0	0	0	0
CPU Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Note: R/W = Read/Write.								

Bit Position	Value	Description
7 DMA_EN	0	The DMA channel is disabled. This bit must be reset to 0 by the software to remove DMA interrupt service requests.
	1	The DMA channel is enabled. This bit is not reset to 0 following completion of a DMA transfer.
6	0	The interrupt is disabled for this DMA channel.
IRQ_DMA	1	The interrupt is enabled for this DMA channel.
5	0	Reserved—must be 0.
4	0	The DMA is configured for CYCLE-STEAL mode.
BURST	1	The DMA is configured for BURST mode.



Bit		
Position	Value	Description
[3:2] DAR_CTL	00	The destination address is unchanged following the transfer of each byte.
	01	The destination address increments following the transfer of each byte.
	10	The destination address decrements following the transfer of each byte.
	11	Reserved.
[1:0] SAR_CTL	00	The source address is unchanged following the transfer of each byte.
	01	The source address increments following the transfer of each byte.
	10	The source address decrements following the transfer of each byte.

11

Reserved.

146



Zilog Debug Interface

ZDI Overview

The Zilog Debug Interface (ZDI) provides a built-in debugging interface to the $eZ80^{\ensuremath{\mathbb{R}}}$ CPU. ZDI provides basic in-circuit emulation features such as:

- Examining and modifying internal registers
- Examining and modifying memory
- Starting and stopping the user program
- Setting program and data break points
- Single-stepping the user program
- Executing user-supplied instructions
- Debugging the final product with the inclusion of one small connector
- Downloading code into SRAM
- C source-level debugging using Zilog Developer Studio II (ZDS II)

The above features are built into the silicon. Control is provided via a two-wire interface that is connected to the ZPAK II emulator. Figure 26 displays a typical setup using a a target board, ZPAK II, and the host PC running Zilog Developer Studio. For more information on ZPAK II and ZDS II, refer to <u>www.zilog.com</u>.



Figure 26. Typical ZDI Debug Setup

The ZDI block for the eZ80190 device provides increased functionality from previous versions. ZDI allows reading and writing of most of the internal registers without disturbing the state of the machine. New features allow READs and writes to memory to occur as fast

zilog[®] | 14

as the ZDI can download and upload data, with a maximum frequency of one-half the CPU clock frequency.

ZDI Interface

ZDI supports a bidirectional serial protocol. The protocol defines any device that sends data as the *transmitter* and any receiving device as the *receiver*. The device controlling the transfer is the *master* and the device being controlled is the *slave*. The master always initiates the data transfers and provides the clock for both receive and transmit operations. The ZDI block on the eZ80190 device is considered a slave in all data transfers.

Figure 27 displays the schematic for building a connector on a target board. This connector allows you to connect directly to the ZPAK II emulator using a six-pin header.



Figure 27. Schematic For Building a Target Board ZPAK II Connector

ZDI Clock and Data Conventions

The two pins used for communication with the ZDI block are the ZDI Clock pin (ZCL) and the ZDI Data pin (ZDA). For general data communication, the data value on the ZDA pin can change only when ZCL is Low (0). The only exception is the ZDI START bit, which is indicated by a High-to-Low transition (falling edge) on the ZDA pin while ZCL is High.

Data is shifted into and out of ZDI, with the msb (bit 7) of each byte being transferred first, and the lsb (bit 0) transferred last. All information is passed between the master and the slave in 8-bit (single-byte) units. Each byte is transferred with nine clock cycles: eight to shift the data, and the ninth for internal operations.

zilog ₁₄₉

ZDI START Condition

All ZDI commands are preceded by the ZDI START signal, which is a High-to-Low transition of ZDA when ZCL is High. The ZDI slave on the eZ80190 device continually monitors the ZDA and ZCL lines for the START signal and does not respond to any command until this condition is met. The master pulls ZDA Low, with ZCL High, to indicate the beginning of a data transfer with the ZDI block. Figure 28 and Figure 29 display a valid ZDI START signal prior to writing and reading data, respectively. A Low-to-High transition of ZDA while the ZCL is High produces no effect.

Data is shifted in during a write to the ZDI block on the rising edge of ZCL, as displayed in Figure 28. Data is shifted out during a read from the ZDI block on the falling edge of ZCL, as displayed in Figure 29. When an operation is completed, the master stops during the ninth cycle and holds the ZCL signal High.



Figure 28. ZDI Write Timing



Figure 29. ZDI Read Timing

zilog

150

ZDI Single-Bit Byte Separator

Following each 8-bit ZDI data transfer, a single-bit byte separator is used. The ZDA pin should be forced High (1) prior to the ZCL rising edge for this 9th bit. For most ZDI operations, the ZDI register address automatically increments during this single-bit byte separator period. The same read or write operation as just completed can then be immediately performed on the next ZDI register. If a different operation or register address is required, a ZDI START signal during the byte separator bit can be used to terminate the previous read or write operation and signify initiation of a new ZDI register operation.

ZDI Register Addressing

Following a START signal, the ZDI master must output the ZDI register address. All data transfers with the ZDI block use special ZDI registers. The ZDI control registers that reside in the ZDI register address space should not be confused with the eZ80190 device peripheral registers that reside in the I/O addressing space of the eZ80190 device.

Many locations in the ZDI control register address space are shared by two registers, one for Read Only access and one for Write Only access. As an example, a read from ZDI register address 00h returns the Product ID Low Byte while a write to this same location, 00h, stores the Low byte of one of the address match values used for generating break points.

The format for a ZDI address is seven bits of address, followed by one bit for read or write control, and completed by a single-bit byte separator in which ZDA must be 1. The data separator time period is used to allow the ZDI master to send a new ZDI START signal, if necessary. The ZDI executes a read or write operation depending on the state of the R/W bit (0 = write, 1 = read). Figure 30 displays the timing for address writes to ZDI registers.



Figure 30. ZDI Address Write Timing

zilog ₁₅₁

ZDI Write Operations

ZDI Single-Byte Write

For single-byte write operations, the address and write-control bit are first written to the ZDI block. Following the 1-bit byte separator, the data is shifted into the ZDI block on the next 8 rising edges of ZCL. The master terminates activity after 8 clock cycles. Figure 31 displays the timing for ZDI single-byte WRITE operations.



Figure 31. ZDI Single-Byte Data Write Timing

ZDI Block Write

The block WRITE operation is initiated in the same manner as the single-byte write operation. After the first data byte is transferred, the ZDI master continues to transmit additional bytes of data to the ZDI slave on the eZ80190 device. After the receipt of each byte of data the ZDI register address increments by one. If the ZDI register address reaches the end of the Write Only ZDI register address space (30h), the address stops incrementing. Figure 32 on page 152 displays the timing for ZDI block WRITE operations.

zilog ₁₅₂



Figure 32. ZDI Block Data Write Timing

ZDI Read Operations

ZDI Single-Byte READ

Single-byte read operations are initiated in the same manner as single-byte write operations, with the exception that the R/W bit of the ZDI register address is set to 1. Upon receipt of a slave address with the R/W bit set to 1, the eZ80190 device's ZDI block loads the selected data into the shifter at the beginning of the first cycle following the 1-bit data separator. The msb is shifted out first. Figure 33 displays the timing for ZDI single-byte WRITE operations.



Figure 33. ZDI Single-Byte Data Read Timing

zilog

ZDI Block READ

A block READ operation is initiated the same as a single-byte read; however, the ZDI master continues to clock in the next byte from the ZDI slave as the ZDI slave continues to output data. The ZDI register address counter increments with each read. If the ZDI register address reaches the end of the Read Only ZDI register address space (20h), the address stops incrementing. Figure 34 displays ZDI block READ timing.



Figure 34. ZDI Block Data Read Timing

Operation Of The eZ80190 Device During ZDI Breakpoints

If the ZDI forces the CPU to break, only the CPU suspends operation. The system clock continues to operate and drive other peripherals. Those peripherals that can operate autonomously from the CPU may continue to operate, if so enabled. For example, the WDT and Programmable Reload Timers continue to count during a ZDI breakpoint.

When using the ZDI interface, any write or read operations of peripheral registers in the I/O address space produces the same effect as read or write operations using the CPU. Because many register Read/Write operations exhibit secondary effects, such as clearing flags or causing operations to commence, the effects of the Read/Write operations during a ZDI break must be taken into consideration. As an example, reading or writing the MACC Accumulator Byte 4 register can cause a bank switch to occur within the MACC.

ZDI Write Only Registers

 Table 81 on page 154 lists the ZDI Write Only registers. Many of the ZDI Write Only addresses are shared with ZDI Read Only registers.

zilog[°]

154

ZDI Address	ZDI Register Name	ZDI Register Function	Reset Value	Page #
00h	ZDI_ADDR0_L	Address Match 0 Low Byte	XXh	156
01h	ZDI_ADDR0_H	Address Match 0 High Byte	XXh	
02h	ZDI_ADDR0_U	Address Match 0 Upper Byte	XXh	
04h	ZDI_ADDR1_L	Address Match 1 Low Byte	XXh	
05h	ZDI_ADDR1_H	Address Match 1 High Byte	XXh	
06h	ZDI_ADDR1_U	Address Match 1 Upper Byte	XXh	
08h	ZDI_ADDR2_L	Address Match 2 Low Byte	XXh	
09h	ZDI_ADDR2_H	Address Match 2 High Byte	XXh	
0Ah	ZDI_ADDR2_U	Address Match 2 Upper Byte	XXh	
0Ch	ZDI_ADDR3_L	Address Match 3 Low Byte	XXh	
0Dh	ZDI_ADDR3_H	Address Match 3 High Byte	XXh	
0Eh	ZDI_ADDR3_U	Address Match 4 Upper Byte	XXh	
10h	ZDI_BRK_CTL	Break Control register	00h	156
13h	ZDI_WR_DATA_L	Write Data Low Byte	XXh	159
14h	ZDI_WR_DATA_H	Write Data High Byte	XXh	159
15h	ZDI_WR_DATA_U	Write Data Upper Byte	XXh	159
16h	ZDI_RW_CTL	Read/Write Control register	00h	160
21h	ZDI_IS4	Instruction Store 4	XXh	161
22h	ZDI_IS3	Instruction Store 3	XXh	
23h	ZDI_IS2	Instruction Store 2	XXh	
24h	ZDI_IS1	Instruction Store 1	XXh	_
25h	ZDI_IS0	Instruction Store 0	XXh	_
30h	ZDI_WR_MEM	Write Memory register	XXh	162

Table 81. ZDI Write Only Registers

ZDI Read Only Registers

Table 82 lists the ZDI Read Only registers. Many of the ZDI Read Only addresses are shared with ZDI Write Only registers.

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

156

Table 83. ZDI Address Match Registers(ZDI_ADDR0_L = 00h, ZDI_ADDR0_H = 01h,
ZDI_ADDR0_U = 02h, ZDI_ADDR1_L = 04h, ZDI_ADDR1_H = 05h, ZDI_ADDR1_U = 06h,
ZDI_ADDR2_L = 08h, ZDI_ADDR2_H = 09h, ZDI_ADDR2_U = 0Ah, ZDI_ADDR3_L = 0Ch,
ZDI_ADDR3_H = 0Dh, ZDI_ADDR3_U = 0Eh)

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

Bit Position	Value	Description
[7:0] ZDI_ADDRX_L, ZDI_ADDRX_H, or ZDI_ADDRX_U	00h– FFh	The four sets of ZDI address match registers are used for setting the addresses for generating break points. The 24-bit addresses are returned by {ZDI_ADDRx_U, ZDI_ADDRx_H, ZDI_ADDRx_L, where x is 0, 1, 2, or 3.

ZDI Break Control Register

The ZDI Break Control register, Table 84, is used to enable break points.

Table 84. ZDI Break Control Register	(ZDI_BRK_CTL = 10h in the ZDI Write Only Register
	Address Space)

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

D	14	

Position	Value	Description
7	0	The ZDI break on the next CPU instruction is disabled.
BRK_NEXT	1	The ZDI break on the next CPU instruction is enabled. The CPU is instructed to use multiple Op Codes and multiple byte operands. This function only breaks on the first Op Code in a multiple Op Code instruction. If both the ZCL and ZDA pins are forced Low (0) during a RESET, this bit is set to 1 and a break occurs on the first instruction following the RESET.

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.
	0	The ZDI break, upon matching break address 1, is disabled.
	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[°]

Bit Position	Value	Description
4	0	The ZDI break, upon matching break address 0, is disabled.
BRK_ADDR0	1	The ZDI break, upon matching break address 0, is enabled. ZDI asserts a break when the CPU address, ADDR[23:0], matches the value in the ZDI Address Match 1 registers, {ZDI_ADDR0_U, ZDI_ADDR0_H, ZDI_ADDR0_L}. If the IGN_LOW_0 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 0 High and Low Byte registers, {ZDI_ADDR0_U, ZDI_ADDR0_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.
2 IGN_LOW_1	0	The <i>Ignore the Low byte</i> function of the ZDI Address Match 1 registers is disabled. If BRK_ADDR1 is set to 1, ZDI initiates a break when the entire 24-bit address, ADDR[23:0], matches the 3-byte value {ZDI_ADDR1_U, ZDI_ADDR1_H, ZDI_ADDR1_L}.
	1	The <i>Ignore the Low byte</i> function of the ZDI Address Match 1 registers is enabled. If BRK_ADDR1 is set to 1, ZDI initiates a break when only the upper 2 bytes of the 24-bit address, ADDR[23:8], match the 2-byte value {ZDI_ADDR1_U, ZDI_ADDR1_H}. As a result, a break can occur anywhere within a 256-byte page.
1 IGN_LOW_0	0	The <i>Ignore the Low byte</i> function of the ZDI Address Match 1 registers is disabled. If BRK_ADDR0 is set to 1, ZDI initiates a break when the entire 24-bit address, ADDR[23:0], matches the 3-byte value {ZDI_ADDR0_U, ZDI_ADDR0_H, ZDI_ADDR0_L}.
	1	The <i>Ignore the Low byte</i> function of the ZDI Address Match 1 registers is enabled. If the BRK_ADDR1 is set to 0, ZDI initiates a break when only the upper 2 bytes of the 24-bit address, ADDR[23:8], match the 2 bytes value {ZDI_ADDR0_U, ZDI_ADDR0_H}. As a result, a break can occur anywhere within a 256-byte page.
	0	ZDI SINGLE STEP mode is disabled.
SINGLE_STEP	1	ZDI SINGLE STEP mode is enabled. ZDI asserts a break following execution of each instruction.

158

zilog

159

ZDI Write Data Registers

Three registers are used in the ZDI Write Only register address space to store the data that is written when a write instruction is sent to the ZDI Read/Write Control register (ZDI_RW_CTL). The ZDI Read/Write Control register, listed in Table 86 on page 160, is located at ZDI address 16h immediately following the ZDI Write Data registers, in Table 85. As a result, the ZDI master is allowed to write the data to {ZDI_WR_U, ZDI WR H, ZDI WR L} and the write command in one data transfer operation.

Table 85. ZDI Write Data Registers (ZDI_WR_U = 13h, ZDI_WR_H = 14h, ZDI_WR_L = 15h)

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

Position	Value	Description
[7:0] ZDI_WR_L, ZDI_WR_H, or ZDI_WR_L	00h– FFh	These registers contain the data that is written during execution of a write operation defined by the ZDI_RW_CTL register. The 24-bit data value is stored as {ZDI_WR_U, ZDI_WR_H, ZDI_WR_L}. If less than 24 bits of data are required to complete the required operation, the data is taken from the LSB(s).

ZDI Read/Write Control Register

The ZDI Read/Write Control register is used in the ZDI Write Only Register address to read data from, write data to, and manipulate the CPU's registers or memory locations. When this register is written, the eZ80190 device immediately performs the operation corresponding to the data value written as listed in Table 86 on page 160. When a read operation is executed via this register, the requested data values are placed in the ZDI Read Data registers {ZDI_RD_U, ZDI_RD_H, ZDI_RD_L}. When a write operation is executed via this register, the write data is taken from the ZDI Write Data registers {ZDI_WR_U, ZDI_WR_L}. For information regarding the CPU registers refer to $eZ80^{\ensuremath{\mathbb{R}}}$ CPU User Manual (UM0077).

(ZDI_RW_CTL = 16h)



160

Hex Value	Command	Hex Value	Command
00	Read {MBASE, A, F} ZDI_RD_U \leftarrow MBASE ZDI_RD_H \leftarrow F ZDI_RD_L \leftarrow A	80	Write AF MBASE \leftarrow ZDI_WR_U F \leftarrow ZDI_WR_H A \leftarrow ZDI_WR_L
01	Read BC ZDI_RD_U \leftarrow BCU ZDI_RD_H \leftarrow B ZDI_RD_L \leftarrow C	81	Write BC BCU \leftarrow ZDI_WR_U B \leftarrow ZDI_WR_H C \leftarrow ZDI_WR_L
02	Read DE ZDI_RD_U \leftarrow DEU ZDI_RD_H \leftarrow D ZDI_RD_L \leftarrow E	82	Write DE DEU \leftarrow ZDI_WR_U D \leftarrow ZDI_WR_H E \leftarrow ZDI_WR_L
03	$\begin{array}{l} \text{Read HL} \\ \text{ZDI_RD_U} \leftarrow \text{HLU} \\ \text{ZDI_RD_H} \leftarrow \text{H} \\ \text{ZDI_RD_L} \leftarrow \text{L} \end{array}$	83	Write HL HLU \leftarrow ZDI_WR_U H \leftarrow ZDI_WR_H L \leftarrow ZDI_WR_L
04	$\begin{array}{l} \text{Read IX} \\ \text{ZDI_RD_U} \leftarrow \text{IXU} \\ \text{ZDI_RD_H} \leftarrow \text{IXH} \\ \text{ZDI_RD_L} \leftarrow \text{IXL} \end{array}$	84	Write IX IXU \leftarrow ZDI_WR_U IXH \leftarrow ZDI_WR_H IXL \leftarrow ZDI_WR_L
05	Read IY ZDI_RD_U \leftarrow IYU ZDI_RD_H \leftarrow IYH ZDI_RD_L \leftarrow IYL	85	Write IY IYU ← ZDI_WR_U IYH ← ZDI_WR_H IYL ← ZDI_WR_L
06	Read SP In ADL mode, SP = SPL. In Z80 [®] mode, SP = SPS.	86	Write SP In ADL mode, SP = SPL. In Z80 mode, SP = SPS.
07	Read PC ZDI_RD_U \leftarrow PC[23:16] ZDI_RD_H \leftarrow PC[15:8] ZDI_RD_L \leftarrow PC[7:0]	87	Write PC PC[23:16] \leftarrow ZDI_WR_U PC[15:8] \leftarrow ZDI_WR_H PC[7:0] \leftarrow ZDI_WR_L
08	Set ADL ADL ← 1	88	Reserved
09	Reset ADL ADL ← 0	89	Reserved

Table 86. ZDI Read/Write Control Register Functions

PS006614-1208

zilog .

Hex Value	e Command	Hex Value	Command
0A	Exchange CPU register sets $AF \leftarrow AF'$ $BC \leftarrow BC'$ $DE \leftarrow DE'$ $HL \leftarrow HL'$	8A	Reserved
0B	Read memory from current PC value, increment PC	8B	Write memory from current PC value, increment PC
Note:	The CPU's alternate register set (A', F', B' programmer must execute the exchange i	, C', D', E nstruction	', HL') cannot be read directly. The ZDI (EXX) to gain access to the alternate

Table 86. ZDI Read/Write Control Register Functions

(ZDI_RW_CTL = 16h) (Continued)

 $(ZDI_IS4 = 21h, ZDI_IS3 = 22h, ZDI_IS2 = 23h,$

Instruction Store 4:0 Registers

The ZDI Instruction Store registers, listed in Table 87, are located in the ZDI Register Write Only address space. They can be written with instruction data for direct execution by the CPU. When the ZDI_IS0 register is written, the eZ80190 device exits the ZDI BREAK mode and executes a single instruction. The Op Codes and operands for the instruction are received from these Instruction Store registers. Instruction Store Register 0 is the first byte fetched, followed by Instruction Store registers 1, 2, 3 and 4, as necessary. Only the bytes the processor requires to execute the instruction must be stored in these registers. Some eZ80[®] CPU instructions, when combined with the MEMORY mode suffixes (.SIS, .SIL, .LIS, or .LIL), require 6 bytes to operate. These 6-byte instructions cannot be executed directly using the ZDI Instruction Store registers.

Note: The Instruction Store 0 register resides at a higher ZDI address than the other Instruction Store registers. This feature allows the use of the ZDI auto-address increment function to load up and execute an instruction with a single data stream from the ZDI master.

ZDI_IS1 = 24h, ZDI_IS0 = 25h)								
Bit	7	6	5	4	3	2	1	0
Reset	Х	Х	Х	Х	Х	Х	Х	Х
CPU Access	W	W	W	W	W	W	W	W

Note: X = Undefined; W = Write.

 Table 87. Instruction Store 4:0 Registers

zilog[®]

Bit Position	Value	Description
[7:0] ZDI_IS4, ZDI_IS3, ZDI_IS2, ZDI_IS1, or ZDI_IS0	00h– FFh	These registers contain the Op Codes and operands for immediate execution by the CPU following a write to ZDI_IS0. The ZDI_IS0 register contains the first Op Code of the instruction. The remaining ZDI_ISx registers contain any additional Op Codes or operand dates required for execution of the required instruction.

ZDI Write Memory Register

A write to the ZDI Write Memory register, listed in Table 88, causes the eZ80190 device to write the 8-bit data to the memory location specified by the current address in the program counter. In Z80[®] MEMORY mode, this address is {MBASE, PC[15:0]}. In ADL MEMORY mode, this address is PC[23:0]. The program counter, PC, increments after each data write. However, the ZDI register address does not increment automatically when this register is accessed. As a result, the ZDI master is allowed to write any number of data bytes by writing to this address one time, and then writing any number of data bytes.

Table 88. ZDI Write Memory Register (ZDI_WR_MEM = 30h in the ZDI Register Write Only Address Space)

Bit		7	6	5	4	3	2	1	0
Reset		Х	Х	Х	Х	Х	Х	Х	Х
CPU Access	JAccess W W W W W W							W	
Note: X = Undefi	ned; W =	Write.							
Bit Position	Value	Descri	ntion						
[7:0] ZDI_WR_MEM	00h– FFh	The 8- write to current followin	Description The 8-bit data that is transferred to the ZDI slave following a write to this address is written to the address indicated by the current program counter. The program counter is incremented following each 8-bits of data. In Z80 MEMORY mode, ({MBASE RO[15:0]]) (8 bits of transferred data. In ADI, MEMORY						

eZ80[®] Product ID Low Byte Register

The Product ID Low and High Byte registers, listed in Table 89 on page 163, combine to provide a means for an external device to determine the particular Product product being addressed. For the eZ80190 device, these two bytes {ZDI_ID_H, ZDI_ID_L}, return the value {00h, 05h}.

mode, $(PC[23:0]) \leftarrow 8$ -bits of transferred data.



Table 89. eZ80[®] Product ID Low Byte Register (ZDI_ID_L = 00h in ZDI Register Read Only Address Space)

Bit	7		6	5	4	3	2	1	0
Reset	0		0	0	0	0	1	0	1
CPU Access	R	2	R	R	R	R	R	R	R
Note: R = Read Only.									
Bit Position	Value	Desc	ription						
[7:0] ZDI_ID_L	05h	{ZDI_ produ	ID_H, Z ict.	ZDI_IC	D_L} = {0	0h, 05h	} indicate	es the e	Z80190

eZ80 Product ID High Byte Register

The Product ID Low and High Byte registers, listed in Table 90, combine to provide a means for an external device to determine the particular Product product being addressed. For the eZ80190 device, these two bytes {ZDI_ID_H, ZDI_ID_L}, return the value {00h, 05h}.

Table 90. eZ80[®] Product ID High Byte Register (ZDI_ID_H = 01h in the ZDI Register Read Only Address Space)

Bit	7	6	5	4	3	2	1	0
Reset	0	0	0	0	0	1	0	0
CPU Access	R	R	R	R	R	R	R	R
Note: R = Read Only.								
Bit Position	Value	Descri	ption					
[7:0] ZDI_ID_H	00h	{ZDI_I[produc	D_H, ZDI <u></u> t.	_ID_L} =	{00h, 05	5h} indica	tes the e	Z80190

eZ80[®] Product ID Revision Register

The Product ID Revision register, listed in Table 91, identifies the current revision of the eZ80190 product. This number is changed for each revision.

<mark>z</mark>ilog[°]

164

Table 91. eZ80[®] Product ID Revision Register (ZDI_ID_REV = 02h in the ZDI Register Read Only Address Space)

Bit	7	6	5	4	3	2	1	0
Reset	Х	X	Х	Х	Х	Х	Х	Х
CPU Access	R	R	R	R	R	R	R	R
Note: X = Undeter	mined; R = R	lead Only	•					
Bit Position	Value	Descri	otion					
[7:0] ZDI_ID_REV	00h– FFh	Identifie	es the curi	rent revis	ion of the	e eZ8019	90 produ	ict.

ZDI Status Register

The ZDI Status register, listed in Table 92, provides current information on the eZ80190 device and the CPU.

Table 92. ZDI Status Register(ZDI_STAT = 03h in the 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
Note: R = Read Only.								

Bit Position	Value	Description
7	0	The CPU is not functioning in ZDI mode.
ZDI_ACTIVE	1	The CPU is currently functioning in ZDI mode.
6	0	No RESET event is currently in progress.
RESEI_PEND	1	A RESET event is in progress.
5	0	eZ80190 is not currently in HALT mode.
HALI	1	eZ80190 is currently in HALT mode.
4	0	The CPU is operating in Z80 MEMORY mode (ADL bit flag = 0).
ADL	1	The CPU is operating in ADL MEMORY mode (ADL bit flag = 1).
3	0	The CPU's Mixed-Memory mode (MADL) bit is reset to 0.
MADL	1	The CPU's Mixed-Memory mode (MADL) bit is set to 1.

Bit		
Position	Value	Description
2 IEF1	0	The CPU's Interrupt Enable Flag 1 is reset to 0. Maskable interrupts are disabled.
	1	The CPU's Interrupt Enable Flag 1 is set to 1. Maskable interrupts are enabled.
[1:0]	00b	Reserved—must be 0.

ZDI Read Register Low, High, and Upper

The ZDI register Read Only address space offers Low, High, and Upper functions, which contain the value read by a read operation from the ZDI Read/Write Control register (ZDI_RW_CTL). This data is valid only while in ZDI BREAK mode and only if the instruction is read by a request from the ZDI Read/Write Control register. See Table 93.

Table 93. ZDI Read Registers—Low, High and Upper (ZDI_RD_L = 10h, ZDI_RD_H = 11h, ZDI_RD_U = 12h)

				-				
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
Note: R = Read Only.								
Bit Position	Value	Descrip	tion					
[7:0] ZDI_RD_L, ZDI_RD_H, or ZDI_RD_U	00h– FFh	Values r ZDI Rea 24-bit va ZDI_RD	read from t Id Control alue is stor _L}.	the mem register o red in {ZI	ory locat during a DI_RD_I	tion as re ZDI read J, ZDI_F	equesteo d operati RD_H,	J by the on. The

ZDI Read Memory Data Value Register

When a read is executed from the ZDI Read Memory Data Value register, listed in Table 94, the eZ80190 device fetches the data from the memory address currently pointed to by the program counter, PC, and the program counter is incremented. In Z80[®] mode, the memory address is {MBASE, PC[15:0]}. In ADL mode, the memory address is PC[23:0]. For more information regarding Z80 and ADL MEMORY modes, refer to $eZ80^{®}$ CPU User Manual (UM0077). The program counter, PC, increments after each data read. However, the ZDI register address does not increment automatically when this

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 da CPU's transfe data tra	ata read from program contred out \leftarrow ansferred out	om the me ounter. Ir \cdot ({MBAS out \leftarrow (SF	emory a i Z80 mo E, SPS} PL).	ddress ir ode, 8-bi). In ADL	ndicated t data _ mode,	by the 8-bit



167

eZ80[®] CPU Instruction Set

Table 95 through Table 104 list the eZ80 CPU instructions available for use with the eZ80190 device. The instructions are grouped by class. More detailed information is available in $eZ80^{\text{®}}$ CPU User Manual (UM0077).



The Sleep (SLP) instruction is not supported on the eZ80190 device. Executing a SLP instruction causes the eZ80190 device to behave as if it has received a two-cycle NOP instruction.

Mnemonic	Instruction
ADC	Add with Carry
ADD	Add without Carry
СР	Compare with Accumulator
DAA	Decimal Adjust Accumulator
DEC	Decrement
INC	Increment
MLT	Multiply
NEG	Negate Accumulator
SBC	Subtract with Carry
SUB	Subtract without Carry

Table 95. Arithmetic Instructions

Table 96. Bit Manipulation Instructions

Mnemonic	Instruction
BIT	Bit Test
RES	Reset Bit
SET	Set Bit



168

Table 97. Block Transfer and Compare Instructions

Mnemonic	Instruction
CPD (CPDR)	Compare and Decrement (with Repeat)
CPI (CPIR)	Compare and Increment (with Repeat)
LDD (LDDR)	Load and Decrement (with Repeat)
LDI (LDIR)	Load and Increment (with Repeat)

Table 98. Exchange Instructions

Mnemonic	Instruction
EX	Exchange registers
EXX	Exchange CPU Multibyte register banks

Table 99. Input/Output Instructions

Mnemonic	Instruction
IN	Input from I/O
IN0	Input from I/O on Page 0
IND (INDR)	Input from I/O and Decrement (with Repeat)
IND2 (IND2R)	Input from I/O and Decrement (with Repeat)
INDM (INDMR)	Input from I/O and Decrement (with Repeat)
INI (INIR)	Input from I/O and Increment (with Repeat)
INI2 (INI2R)	Input from I/O and Increment (with Repeat)
INIM (INIMR)	Input from I/O and Increment (with Repeat)
OTDM (OTDMR)	Output to I/O and Decrement (with Repeat)
OTIM (OTIMR)	Output to I/O and Increment (with Repeat)
OUT	Output to I/O
OUT0	Output to I/0 on Page 0
OUTD (OTDR)	Output to I/O and Decrement (with Repeat)
OUTD2 (OTD2R)	Output to I/O and Decrement (with Repeat)

zilog[°]

169

Table 99. Input/Output Instructions (Continued)

Mnemonic	Instruction
OUTI (OTIR)	Output to I/O and Increment (with Repeat)
OUTI2 (OTI2R)	Output to I/O and Increment (with Repeat)
TSTIO	Test I/O

Table 100. Load Instructions

Mnemonic	Instruction
LD	Load
LEA	Load Effective Address
PEA	Push Effective Address
POP	Рор
PUSH	Push

Table 101. Logical Instructions

Mnemonic	Instruction
AND	Logical AND
CPL	Complement Accumulator
OR	Logical OR
TST	Test Accumulator
XOR	Logical Exclusive OR

Table 102. Processor Control Instructions

Mnemonic	Instruction
CCF	Complement Carry Flag
DI	Disable Interrupts
El	Enable Interrupts
HALT	Halt



Table 102. Processor Control Instructions (Continued)

Mnemonic	Instruction
IM	Interrupt Mode
NOP	No Operation
RSMIX	Reset Mixed-Memory Mode Flag
SCF	Set Carry Flag
SLP	Sleep (not supported on the eZ80190 device)
STMIX	Set Mixed-Memory Mode Flag

Table 103. Program Control Instructions

Mnemonic	Instruction
CALL	Call Subroutine
CALL cc	Conditional Call Subroutine
DJNZ	Decrement and Jump if Nonzero
JP	Jump
JP cc	Conditional Jump
JR	Jump Relative
JR cc	Conditional Jump Relative
RET	Return
RET cc	Conditional Return
RETI	Return from Interrupt
RETN	Return from Nonmaskable interrupt
RST	Restart

Table 104. Rotate and Shift Instructions

Mnemonic	Instruction
RL	Rotate Left
RLA	Rotate Left–Accumulator
RLC	Rotate Left Circular
zilog ₁₇₁

Table 104. Rotate and Shift Instructions (Continued)

Mnemonic	Instruction
RLCA	Rotate Left Circular–Accumulator
RLD	Rotate Left Decimal
RR	Rotate Right
RRA	Rotate Right–Accumulator
RRC	Rotate Right Circular
RRCA	Rotate Right Circular–Accumulator
RRD	Rotate Right Decimal
SLA	Shift Left
SRA	Shift Right Arithmetic
SRL	Shift Right Logical

zilog ₁₇₂

Op-Code Map

Table 105 through Table 111 on page 178 list the hex values for each of the $eZ80^{\ensuremath{\mathbb{R}}}$ instructions.

Table 105. Op Code Map—First Op Code

I	Leg	end	Lower C	Dp Code N	libble												
	_	Upper	ŧ														
	C	Dp Code Nibble 🛰	4	<u>.</u>													
		110010		J ⊲ — Mr	nemonic												
	Fi	irst Opera	nd 🖈	Seco	nd Operar	nd											
								Lower	Nibble	(Hex)							
		0	1	2	3	4	5	6	7	8	9	А	В	С	D	Е	F
	0	NOP	LD	LD	INC	INC	DEC	LD	RLCA	EX	ADD	LD	DEC	INC	DEC	LD	RRCA
			BC,	(BC),A	BC	В	В	B,n		AF,AF'	HL,BC	A,(BC)	BC	С	С	C,n	
			Mmn													. –	
	1	DJNZ	LD		INC	INC	DEC	LD	RLA	JR	ADD	LD	DEC	INC	DEC		RRA
		d	DE,	(DE),A	DE	D	D	D,n		d	HL,DE	A,(DE)	DE	E	E	E,n	
	2		Mmn														
	2			LD (Masa)			DEC		DAA	JR			DEC		DEC		CPL
		NZ,U	ΠL, Mmn	(ivimn),	ΠL	п	п	п,n		Z,ŭ	nl,nl	HL,	ΠL	L	L	L,N	
	3	ID					DEC		SCE	ID			DEC		DEC		CCE
	5	NC d	SP	(Mmn)	SP	(HL)		(HL) n	301				SP			Δn	001
(Xi		NO,U	Mmn	Δ	0	(112)	(11)	(112),11		Or,u	112,01	(Mmn)	0	~	~	7,11	
	4	SIS	ID		ID	ID	ID	ID	ID	ID	JIS		ID	ID	ID	ID	1 D
		suffix	B.C	B.D	B.F	B.H	BI	B.(HI.)	B.A	C.B	suffix	C.D	C.F	C.H	CI	C.(HI.)	C.A
	5	LD	LD	.SIL	LD	LD	LD	LD	LD	LD	LD	LD	LIL	LD	LD	LD	LD
		D,B	D,C	suffix	D,E	D,H	D,L	D,(HL)	D,A	E,B	E,C	E,D	suffix	E,H	E,L	E,(HL)	E,A
	6	LD	LD	LD	LD	LD	LD	ĹD	LD	LD	LD	LD	LD	LD	LD	ĹD	LD
Ĭ.		H,B	H,C	H,D	H,E	H,H	H,L	H,(HL)	H,A	L,B	L,C	L,D	L,E	L,H	L,L	L,(HL)	L,A
e	7	LD	LD	LD	LD	LD	LD	HALT	LD	LD	LD	LD	LD	LD	LD	LD	LD
ld	-	(HL),B	(HL),C	(HL),D	(HL),E	(HL),H	(HL),L		(HL),A	A,B	A,C	A,D	A,E	A,H	A,L	A,(HL)	A,A
ž	8	ADD	ADD	ADD	ADD	ADD	ADD	ADD	ADD	ADC	ADC	ADC	ADC	ADC	ADC	ADC	ADC
٦.	~	A,B	A,C	A,D	A,E	A,H	A,L	A,(HL)	A,A	A,B	A,C	A,D	A,E	A,H	A,L	A,(HL)	A,A
ð	9	SUB	SUB	SOR	SUB	SUB	SUB	SUB	SUB	SBC	SBC	SBC	SBC	SBC	SBC	SBC	SBC
Ĵ	Δ									A,B XOR			A,E XOR		A,L XOR	A,(HL)	A,A XOR
	в	OR	OR	OR	OR	OR	OR	OR	OR	CP	CP	CP	CP	CP	CP	CP	CP
	_	A.B	A.C	A.D	A.E	A.H	A.L	A.(HL)	A.A	A.B	A.C	A.D	A.E	A.H	A.L	A.(HL)	A.A
	С	RET	POP	JP	JP	CALL	PUSH	ADD	RST	RET	RET	JP	Table	CALL	CALL	ADC	RST
		NZ	BC	NZ,	Mmn	NZ,	BC	A,n	00h	Z		Ζ.	106	Ζ,	Mmn	A,n	08h
				Mmn		Mmn		,				Mmn		Mmn			
	D	RET	POP	JP	OUT	CALL	PUSH	SUB	RST	RET	EXX	JP	IN	CALL	Table	SBC	RST
		NC	DE	NC,	(n),A	NC,	DE	A,n	10h	CF		CF,	A,(n)	CF,	107	A,n	18h
				Mmn		Mmn						Mmn		Mmn			
	Е	RET	POP	JP	EX	CALL	PUSH	AND	RST	RET	JP	JP	EX	CALL	Table	XOR	RST
		PO	HL	PO,	(SP),H	PO,	HL	A,n	20h	PE	(HL)	PE,	DE,HL	PE,	108	A,n	28h
				Mmn	L	Mmn						Mmn		Mmn	- 11		
	F	RET	POP	JP	DI	CALL	PUSH	OR	RST	RET	LD	JP	EI	CALL	Table	CP	RST
		Р	AF	P,		P,	AF	A,n	30h	M	SP,HL	M,		M,	109	A,n	38h
		Nataci	-0 -	Mmn	1	Mmn	hit or la			h:+ +'		Mmn	diam la	Mmn			
		inoles:	11=0-D	ii uala: I	virnn = 1	0- 01 Z4	1006 JIU-1	a or dai	.a. u=ŏ-	บแเพด	s-comp	ement (uispiace	ment.			



Table 106. Op Code Map—Second Op Code after 0CBh

Lower Nibble of 2nd Op Code

Legend

ا h of Se Op	Upper Nibble 4 of Second A RES Mnemonic															
Firs	t Opera	nd $\frac{4}{4}$	Se	econd O	perand											
							L٥١	ver Nit	oble (H	ex)						
	0	1	2	3	4	5	6	7	8	9	А	В	С	D	Е	F
0	RLC	RLC	RLC	RLC	RLC	RLC	RLC	RLC	RRC	RRC	RRC	RRC	RRC	RRC	RRC	RRC
	В	С	D	E	Н	L	(HL)	Α	В	С	D	E	Н	L	(HL)	A
1	RL	RL	RL	RL	RL	RL	RL	RL	RR	RR	RR	RR	RR	RR	RR	RR
-	В	С	D	E	H	L	(HL)	A	B	C	D	E	H	L	(HL)	A
2	SLA	SLA	SLA	SLA	SLA	SLA	SLA	SLA	SRA	SRA	SRA	SRA	SRA	SRA	SRA	SRA
~	В	C	D	E	н	L	(HL)	A	B			E	H		(HL)	A
3									SRL	SRL	SRL	SRL		SRL		SRL
1	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT		RIT
-	0 B			0 F	0 H	01		0 4	1 B	1 C	1 D	1 F	1 H	11	1 (HI.)	1 A
5	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT
-	2.B	2.C	2.D	2.E	2.H	2.L	2.(HL)	2.A	3.B	3.C	3.D	3.E	3.H	3.L	3.(HL)	3.A
$\widehat{\mathbf{x}}$ 6	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT
Ч.	4,B	4,C	4,D	4,E	4,H	4,L	4,(HL)	4,A	5,B	5,C	5,D	5,E	5,H	5,L	5,(HL)	5,A
_ ص 7	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT	BIT
lq	6,B	6,C	6,D	6,E	6,H	6,L	6,(HL)	6,A	7,B	7,C	7,D	7,E	7,H	7,L	7,(HL)	7,A
^ĭ Ž 8	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES
ъ.	0,B	0,C	0,D	0,E	0,H	0,L	0,(HL)	0,A	1,B	1,C	1,D	1,E	1,H	1,L	1,(HL)	1,A
đ 9	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES
⊃ .	2,B	2,C	2,D	2,E	2,H	2,L	2,(HL)	2,A	3,B	3,0	3,D	3,E	3,H	3,L	3,(HL)	3,A
A	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES	RES
D	4,D			4,E		4,L	4,(NL)		3,D					DEQ		DES
Б	6 B	60		6 E	6 H	61		64	7 B				7 H	71	7 (HL)	
С	SFT	SFT	SFT	SFT	SET	SFT	SFT	SFT	SFT	SFT	SFT	SFT	SET	SFT	SFT	SET
Ũ	0.B	0.C	0.D	0.E	0.H	0.L	0.(HL)	0.A	1.B	1.C	1.D	1.E	1.H	1.L	1.(HL)	1.A
D	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET
	2,B	2,C	2,D	2,E	2,H	2,L	2,(HL)	2,A	3,B	3,C	3,D	3,E	3,H	3,L	3,(HL)	3,A
Е	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET
	4,B	4,C	4,D	4,E	4,H	4,L	4,(HL)	4,A	5,B	5,C	5,D	5,E	5,H	5,L	5,(HL)	5,A
F	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET	SET
	6,B	6,C	6,D	6,E	6,H	6,L	6,(HL)	6,A	7,B	7,C	7,D	7,E	7,H	7,L	7,(HL)	7,A

Notes: n=8-bit data; Mmn=16- or 24-bit addr or data; d=8-bit two's-complement displacement.



Table 107. Op Code Map—Second Op Code After 0DDh



	Lower Nibble (Hex)															
	0	1	2	3	4	5	6	7	8	9	А	В	С	D	Е	F
0								LD		ADD						LD
								BC,		IX,BC						(IX+d),
								(IX+d)		400						BC
1																
								DE, (IX±d)		IA,DE						
2			ID	INC	INC	DEC						DEC	INC	DEC		
2		IX.	(Mmn).	IX	IXH	IXH	IXH.n	(IX+d)			IX.	IX		IXI	IXI.n	(IX+d).
		Mmn	IX					((Mmn)				.,,	HL
3		LD IY,			INC	DEC	LD (IX	LD IX,		ADD	(LD	LD
•		(IX+d)			(IX+d)	(IX+d)	+d),n	(IX+d)		IX,SP					(IX+d),	(IX+d),
															IY	IX
4					LD	LD	LD B,						LD	LD	LD C,	
					B,IXH	B,IXL	(IX+d)						C,IXH	C,IXL	(IX+d)	
5					LD	LD	LD D,						LD	LD	LD E,	
~					D,IXH	D,IXL	(IX+d)						E,IXH	E,IXL	(IX+d)	
6		LD					LD H,								LD L,	
	IXH,B	IXH,C	IXH,D	IXH,E			(1X+a)	IXH,A	IXL,B	IXL,C	IXL,D	IXL,E		IXL,IX	(IX+a)	IXL,A
⇒ 7		П														
(He)	(IX+d)	(IX+d)	(IX+d)	(IX+d)	(IX+d)	(IX+d)		(IX+d)							(IX+d)	
bble	(л.с.а), В	C	D	E	Η	L		Α					7 (,17 (1 1	/ (,I/ (E	(17() (1)	
ا ق ا					ADD	ADD	ADD						ADC	ADC	ADC	
npp					A,IXH	A,IXL	А,						A,IXH	A,IXL	А,	
							(IX+d)								(IX+d)	
9					SUB	SUB	SUB						SBC	SBC	SBC	
					A,IXH	A,IXL	А,						A,IXH	A,IXL	А,	
							(IX+d)						VOD	VOD	(IX+d)	
A							AND						XOR	XOR	XOR	
					A,IXH	A,IXL	A,						A,IXH	A,IXL	A,	
B					OR	OR							CP	CP		
D							(IX+d)								(X+d)	
С					, ,,,, , , , , ,	, ,,,, (_	(1)(1)(4)					Table	, ,,,, , , , , ,	, i,i, i	(1)(1)(4)	
•												110				
D																
Е		POP		EX		PUSH				JP						
		IX		(SP),I		IX				(IX)						
-				Х						1.5						
F										LD						
										SP,IX						

PS006614-1208

Op-Code Map



Table 108. Op Code Map—Second Op Code After 0EDh



	Lower Nibble (Hex)																
	_	0	1	2	3	4	5	6	7	8	9	Α	В	С	D	Е	F
	0	IN0	OUT0	LEA	LEA	TST			LD	IN0	OUT0			TST			LD
		B,(n)	(n),B	BC,	BC,	A,B			BC,	C,(n)	(n),C			A,C			(HL),
	1	IN0	ΟΠΤΟ		IFA	TST				IN0	ΟΠΤΟ			TST			
	'	D.(n)	(n).D	DE.	DE.	A.D			DE.	E.(n)	(n).E			A.E).
		~ /	()/	IX+d	IY+d	,			(HL)	~ /	× //			,			DE
	2	IN0	OUT0	LEA	LEA	TST			LD HL,	IN0	OUT0			TST			LD
		H,(n)	(n),H	HL	HL	A,H			(HL)	L,(n)	(n),L			A,L			(HL),
	2		או ח ו		,IY+d	тет			או ח ו	INIO				тет			HL
	S		LDTT, (HL)		ILEA II IV+d				(HL)	Δ (n)	(n) Δ						
			(11)	,iX 'u	,11 'u	∽,(ייב)			(11)	Λ,(Π)	(11),7			<i></i> ,,,		Υ Υ	IX
	4	IN	OUT	SBC	LD	NEG	RETN	IM 0	LD	IN	OUT	ADC	LD	MLT	RETI		LD
		B,(BC)	(BC),B	HL,BC	(Mmn)				I,A	C,(C)	(C),C	HL,BC	BC,	BC			R,A
					,								(Mmn)				
	-	INI		000	BC			11.1.1		INI						IM O	
	э	IIN D (BC)	(BC) D		(Mmn)					E (C)	(C) F						
ě		в,(во)	(00),0	112,02		IY+d	IX+d		73,1	∟,(∪)	(∪),∟	,	(Mmn)				73,13
Ξ.					DE		-						· /				
ble	6	IBN	OUT	SBC	LD	TST	PEA	PEA	RRD	IN	OUT	ADC	LD	MLT	LD	LD	RLD
ÏZ		H,(C)	(BC),H	HL,HL	(Mmn)	A,n	IX+d	IY+d		L,(C)	(C),L	HL,HL	HL,	HL	MB,A	A,MB	
e					, LII								(Mmn)				
dd	7			SBC		TSTIO		SLP		IN	OUT	ADC	ID	ΜГΤ	STMIX	RSMI	
	'			HL.SP	(Mmn)	n		0LI		A.(C)	(C).A	HL.SP	SP.	SP	O I WII/	X	
				, -	, ,					<i>X</i> - <i>Y</i>	(- <i>)</i> ,	, -	(Mmn)				
					SP												
	8			INIM	OTIM	INI2						INDM	OTDM	IND2			
	9			INIMR	ΟΤΙΜ	INI2R						INDM	OTDM	IND2R			
	Ū				R							R	R				
	A	LDI	CPI	INI	OUTI	OUTI2				LDD	CPD	IND	OUTD	OUTD			
	_				OTID	OTIOD					0000		OTOD	2			
	в	LDIR	CPIR	INIR	OUR	OTIZR				LDDR	CPDR	INDR	OIDR				
	C.													г			
	5																
	D																
	F																
	-																



Table 109. Op Code Map—Second Op Code After 0FDh



Notes: n=8-bit data; Mmn=16- or 24-bit addr or data; d=8-bit two's-complement displacement.





Table 110. Op Code Map—Fourth Byte After 0DDh, 0CBh, and dd

Lower Nibble of 4th Byte



Notes: d=8-bit two's complement displacement.

Legend





Table 111. Op Code Map—Fourth Byte After 0FDh, 0CBh, and dd



Notes: d=8-bit two's-complement displacement.

Legend



Crystal Oscillator

The eZ80190 device features an on-chip crystal oscillator that supplies clocks to the internal eZ80[®] CPU core, to peripherals, and to the external pin. The clock circuitry uses the three dedicated pins X_{IN} , X_{OUT} , and PHI.

The external clock/oscillator (X_{IN}) input features two clock-generation options. X_{IN} may be used to interface the internal oscillator to an external oscillator (see Figure 35). Typical circuit parameters are C1 = C2 = 10 pF and R = 1 M Ω using a parallel resonant crystal.

 X_{IN} can also accept a CMOS-level clock input. The oscillator output (X_{OUT}) connects the internal crystal oscillator to an external crystal oscillator. If an external clock is used, X_{OUT} should be left unconnected. The PHI pin, which drives the high-speed system clock, may be used to synchronize other peripherals to the eZ80190 device system clock.



Note: *These values are typical values only. Actual values must be tuned for the crystal and the frequency of operation.

Figure 35. Crystal Oscillator

zilog

Electrical Characteristics

Absolute Maximum Ratings

Stresses greater than those listed in Table 112 may cause permanent damage to the device. These ratings are stress ratings only. Operation of the device at any condition outside those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability. For improved reliability, unused inputs should be tied to one of the supply voltages $(V_{DD} \text{ or } V_{SS})$.

Table 112. Absolute Maximum Ratings

Parameter	Min	Max	Units	Notes
Ambient temperature under bias	-40	+105	С	1
Storage temperature	-65	+150	С	
Voltage on any pin with respect to V _{SS}	-0.3	+6.0	V	2
Voltage on V_{DD} pin with respect to V_{SS}	-0.3	+6.0	V	
Total power dissipation		520	mW	
Maximum current out of V _{SS}		145	mA	
Maximum current into V _{DD}		145	mA	
Maximum output current from active output pin	-8	+8	mA	

Notes:

1. Operating temperature is listed in Table 113.

2. This voltage applies to all pins except where otherwise noted.



181

DC Characteristics

Table 113 lists the Direct Current characteristics of the eZ80190 device.

Table 113. DC Characteristics

		Stand Temperatu = 0 °C to	dard Ire Range 5 70 °C	Exter Temperatu = -40 °Cte	nded ire Range o 105 °C		
Symbol	Parameter	Min	Мах	Min	Max	Units	Conditions
V _{DD}	Supply Voltage	3.0	3.6	3.0	3.6	V	
V _{IL}	Low Level Input Voltage	-0.3	0.8V	-0.3	0.8V	V	
V _{IH}	High Level Input Voltage	$0.7 \mathrm{xV}_{\mathrm{DD}}$	5.5	$0.7 \mathrm{xV}_{\mathrm{DD}}$	5.5	V	
V _{OL}	Low Level Output Voltage		0.4		0.4	V	V _{DD} = 3.0 V; I _{OL} = 1 mA
V _{OH}	High Level Output Voltage	2.4		2.4		V	V _{DD} = 3.0 V; I _{OH} = ,— mA
I _{IL}	Input Leakage Current	-10	+10	-10	+10	μA	V _{DD} = 3.6 V; V _{IN} = V _{DD} or V _{SS} *
I _{TL}	Tri-State Leakage Current	-10	+10	-10	+10	μA	V _{DD} = 3.6 V
* This co	ndition excludes the Z	DA and ZCL p	oins, when a	driven Low, du	ue to the pre	sence of	on-chip pull-ups.

In the following pages, Figure 36 displays the typical current consumption of the eZ80190 device versus the number of WAIT states while operating 25 °C, 3.3 V, and with either a 1 MHz or 5 MHz system clock. Figure 37 displays the typical current consumption of the eZ80190 device versus the number of WAIT states while operating 25 °C, 3.3 V, and with either a 20 MHz or 50 MHz system clock. Figure 38 displays the typical current consumption of the eZ80190 device versus the system clock frequency while operating 25 °C, 3.3 V, and with either a 20 MHz or 50 MHz states. Figure 39 displays the typical current consumption of the eZ80190 device versus the system clock frequency while operating 25 °C, 3.3 V, and using 0, 2, or 7 WAIT states. Figure 39 displays the typical current consumption of the eZ80190 device versus the system clock frequency while operating at 3.3 V, 7 WAIT states, and with either a 5 MHz, 20 MHz, or 50 MHz system clock.

zilog ₁₈₂



ICC vs. WAIT States (Typical @ 3.3V, 25C)

ICC vs WAIT States (Typical @ 3.3V, 25C)



Figure 37. I_{CC} vs. WAIT2

zilog ₁₈₃



ICC vs. Frequency (Typical @3.3V, 25C)

Figure 38. I_{CC} vs. Frequency

ICC vs. Temp (Typical @ 3.3V, 7 WAIT States)



Figure 39. I_{CC} vs. Temperature

zilog

184

AC Characteristics

The section provides information on the Alternating Current (AC) characteristics and timing of the eZ80190 device. All AC timing information assumes a standard load of 50 pF on all outputs.

Table 114. AC Characteristics

	_	$T_{A} = 0^{0}C$	to 70 ⁰ C	$T_{A} = -40^{0}$	C to 105 ⁰ C				
Symbol	Parameter	Min	Max	Min	Мах	Units	Conditions		
T _{XIN}	System Clock Cycle Time	20		20		ns	V _{DD} = 3.0–3.6 V		
T _{XINH}	System Clock High Time	8		8		ns	V _{DD} = 3.0–3.6 V; T _{CLK} = 20 ns		
T _{XINL}	System Clock Low Time	8		8		ns	V _{DD} = 3.0–3.6 V; T _{CLK} = 20 ns		
T _{XINR}	System Clock Rise Time		2		2	ns	V _{DD} = 3.0–3.6 V; T _{CLK} = 20ns		
T _{XINF}	System Clock Fall Time		2		2	ns	V _{DD} = 3.0–3.6 V; T _{CLK} = 20 ns		



External Memory Read Timing

Figure 40 and Table 115 display the timing for external READs.



Figure 40. External Memory Read Timing

Table 115. External Read Timing

		Delay	y (ns)
Parameter	Description	Min.	Max.
T ₁	Clock Rise to ADDR Valid Delay	_	10.2
T ₂	Clock Rise to ADDR Hold Time	1.6	_
T ₃	Clock Rise to Output DATA Valid Delay	0.0	
T ₄	DATA Hold Time from Clock Rise	5.0	_
T ₅	Clock Rise to CSx Assertion Delay	3.0	10.5
T ₆	Clock Rise to CSx Deassertion Delay	3.0	9.7
T ₇	Clock Rise to MREQ Assertion Delay	2.8	9.6

zilog

186

Table 115. External Read Timing (Continued)

		Delay (ns)				
Parameter	Description	Min.	Max.			
T ₈	Clock Rise to MREQ Deassertion Delay	1.6	6.9			
T ₉	Clock Rise to RD Assertion Delay	3.0	9.8			
T ₁₀	Clock Rise to RD Deassertion Delay	2.5	7.1			

External Memory Write Timing

Figure 41 and Table 116 on page 187 display the timing for external writes.





zilog[°]

		Dela	y (ns)
Parameter	Description	Min.	Max.
T ₁	Clock Rise to ADDR Valid Delay		10.2
T ₂	Clock Rise to ADDR Hold Time	1.6	_
T ₃	Clock Rise to Output DATA Valid Delay		10.2
T ₄	DATA Hold Time from Clock Rise	5.0	_
T ₅	Clock Rise to $\overline{\text{CSx}}$ Assertion Delay	3.0	10.5
Т ₆	Clock Rise to $\overline{\text{CSx}}$ Deassertion Delay	3.0	9.7
T ₇	Clock Rise to MREQ Assertion Delay	2.8	9.6
T ₈	Clock Rise to MREQ Deassertion Delay	1.6	6.9
T ₉	Clock Fall to WR Assertion Delay	1.5	3.9
T ₁₀	Clock Rise to WR Deassertion Delay*	1.4	4.1

Table 116. External Write Timing

* At the conclusion of a write cycle, deassertion of WR always occurs before any change to ADDR, DATA, CSx, or MREQ.

187



External I/O Read Timing

Figure 42 and Table 117 display the timing for external I/O reads.



Figure 42. External I/O Read Timing

		Delay	/ (ns)
Parameter	Description	Min	Мах
T ₁	Clock Rise to ADDR Valid Delay	_	10.2
T ₂	Clock Rise to ADDR Hold Time	1.6	
T ₃	Input DATA Valid to Clock Rise Setup Time	0.0	
T ₄	DATA Hold Time from Clock Rise	5.0	_
T ₅	Clock Rise to CSx Assertion Delay	3.0	10.5
T ₆	Clock Rise to CSx Deassertion Delay	3.0	9.7
T ₇	Clock Rise to IORQ Assertion Delay	2.1	10.3
Т ₈	Clock Rise to IORQ Deassertion Delay	4.1	7.9

Table 117. External I/O Read Timing

zilog[°]

189

Table 117. External I/O Read Timing (Continued)

			Delay (ns)	
Parameter	Description	Min	Мах	
T ₉	Clock Rise to RD Assertion Delay	3.0	9.8	
T ₁₀	Clock Rise to RD Deassertion Delay	2.5	7.1	

External I/O Write Timing

Figure 43 and Table 118 display the timing for external I/O writes.



Figure 43. External I/O Write Timing

zilog[°]

190

		Delay (ns)	
Parameter	Description	Min	Max
T ₁	Clock Rise to ADDR Valid Delay	_	10.2
T ₂	Clock Rise to ADDR Hold Time	1.6	_
T ₃	Clock Rise to Output DATA Valid Delay	_	10.2
T ₄	DATA Hold Time from Clock Rise	5.0	_
T ₅	Clock Rise to CSx Assertion Delay	3.0	10.5
T ₆	Clock Rise to CSx Deassertion Delay	3.0	9.7
T ₇	Clock Rise to IORQ Assertion Delay	2.1	10.3
T ₈	Clock Rise to IORQ Deassertion Delay	4.1	7.9
T ₉	Clock Fall to WR Assertion Delay	1.5	3.9
T ₁₀	Clock Rise to \overline{WR} Deassertion Delay*	1.4	4.1

Table 118. External I/O Write Timing

* At the conclusion of a write cycle, deassertion of WR always occurs before any change to ADDR, DATA, CSx, or IORQ.

Wait State Timing for Read Operations

Figure 44 on page 191 displays the extension of the memory access signals using a single WAIT state for a read operation. The $\overline{\text{WAIT}}$ signal is not delivered to a pin on the eZ80190 device. It is displayed here for informational purposes only.





Figure 44. Wait State Timing for Read Operations

Wait State Timing for Write Operations

Figure 45 on page 192 displays the extension of the memory access signals using a single WAIT state for a write operation. The $\overline{\text{WAIT}}$ signal is not delivered to a pin on the eZ80190 device and is displayed here for informational purposes only.





Figure 45. Wait State Timing for Write Operations

General-Purpose I/O Port Input Sample Timing

Figure 46 on page 193 displays timing of the GPIO input sampling. The input value on a GPIO port pin is sampled on the rising edge of the system clock. The port value is then available to the CPU on the second rising clock edge following the change of the port value.



Figure 46. Port Input Sample Timing

General-Purpose I/O Port Output Timing

Figure 47 and Table 119 on page 194 display the timing of the GPIO outputs.



Figure 47. GPIO Port Output Timing

eZ80190

193

Product Specification

zilog



194

Table 119. GPIO Port Output Timing

		Delay (ns)	
Parameter	Description	Min	Max
T ₁	Clock Rise to Port Output Valid Delay		8.9
T ₂	Clock Rise to Port Output Hold Time	1.8	

External Bus Acknowledge Timing

Table 120 lists bus acknowledge timing details.

Table 120. Bus Acknowledge Timing

		Delay (ns)	
Parameter	Description	Min	Max
T ₁	Clock Rise to BUSACKN Assertion Delay	_	6.5
T ₂	Clock Rise to BUSACKN Deassertion Delay	2.3	

External System Clock Driver Timing

Table 121 lists timing information for the PHI pin. The PHI pin allows external peripherals to synchronize with the internal system clock driver on the eZ80190 device.

Table 121. PHI System Clock Timing

		Del	Delay (ns)	
Parameter	Description	Min	Max	
T ₁	Clock Rise to PHI Rise	1.8	4.0	
T ₂	Clock Fall to PHI Fall	2.0	4.8	





Packaging

Figure 48 displays the 100-pin LQFP (also called the VQFP) package for the eZ80190 device.



Figure 48. 100-Pin LQFP Package



Ordering Information

Order eZ80190 from Zilog[®], using the following part numbers. For more information regarding ordering, consult your local Zilog sales office. Zilog website <u>www.zilog.com</u>, lists all regional offices and provides additional eZ80190 product information.

Table 122 lists a part number, a product specification index code, and a brief description of each eZ80190 part.

Table 122. Ordering Information

Part	PSI	Description
eZ80190	eZ80190AZ050SC	100-pin LQFP, 50MHz, Standard Temperature
	eZ80190AZ050EC	100-pin LQFP, 50MHz, Extended Temperature
eZ80190 Development Kit	eZ8019000100ZCO	Complete Development Kit

The latest information regarding software and other product updates is available for down-load at <u>www.zilog.com</u>.

Part Number Description

Zilog part numbers consist of a number of components, as indicated in the following examples:

Zilog Base Product	S		
eZ80 [®]	eZ80 CPU Zilog prefix		
190	Product N	umber	
AZ	Package		
050	Speed		
S or E	Temperature		
С	Environmental Flow		
Package		AZ = LQFP (also called the VQFP)	
Speed		050 = 50 MHz	
Standard Temperature		S = 0 °C to +70 °C	
Extended Temperature		E = -40 °C to +105 °C	
Environmental Flow		C = Plastic Standard	

zilog ₁₉₇

Example. Part number eZ80190AZ050SC is an eZ80[®] CPU product in an LQFP package, operating with a 50 MHz external clock frequency over a 0 °C to +70 °C temperature range, and built using the Plastic Standard environmental flow.

Zilog

Index

Numerics

100-Pin LQFP Package 195
100-pin LQFP package 4
16-bit divisor count 64, 65
16-bit downcounter 31, 63
16-bit reload register 31
16-bit start value 33
16-MB linear addressing 30
24-bit address bus 23
24-bit address mode 30
4-bit clock prescaler 31

Α

Absolute Maximum Ratings 180 Absolute maximum ratings 180 AC Characteristics 184 ACK 94, 98, 99, 100, 101, 102, 105, 107, 109, 110 ACK bit 103 Acknowledge 94 Acknowledge, I2C 94 ADDR06 ADDR16 **ADDR10**7 **ADDR11** 8 **ADDR12** 8 **ADDR13** 8 **ADDR14** 8 **ADDR15** 8 **ADDR16** 9 ADDR179 **ADDR18**9 **ADDR19** 9 ADDR26 **ADDR20**9 ADDR21 10

ADDR22 10 **ADDR23** 10 ADDR36 ADDR46 ADDR56 **ADDR6** 7 **ADDR7** 7 **ADDR8** 7 ADDR97 Address Bus 6, 7, 8, 9, 10 address bus 53, 56, 57, 137 address bus, 24-bit 23 Address Match 154 address match 53, 57, 150 Address Match, ZDI Registers 155, 157, 158 Addressing 104 ADL MEMORY mode 53 ADL Mode 138, 139 ADL mode 155, 160, 165 ADL mode bit 138, 139 alternate I/O functions 45 alternate-function I/O 62 Alternatives to OTI2R and INI2R 117 AND connection 95 Arbitration 98, 99, 100, 101, 102, 109, 110 arbitration 97, 103, 106 Architectural Overview 1 Arithmetic Instructions 167 asynchronous communications protocol 67,68 asynchronous communications protocol bits 69 Asynchronous Receiver/Transmitter 62 asynchronous reception protocol 68 asynchronous serial data 14, 15, 17, 18 asynchronous transmission protocol 68 auto-address increment function 161



automatic reload 35, 36, 37, 38

В

Baud Rate Generator 63 Baud Rate Generator Functional Description 63 Baud Rate Generator output 63 bidirectional serial protocol 148 BI—see Break condition 80 **BI—see Break Indication 80** bit generation and detection, parity 67 bit generation and detection, start 67 bit generation and detection, stop 67 Bit Manipulation Instructions 167 block data transfer 140 Block Diagram 2 Block Transfer and Compare Instructions 168 Break Indication 70, 80 break signal 69, 77 **BRG** 63 BRG clock 69 BRG clock divisor ratio 65 BRG counter 63, 64, 65 BRG Divisor Latch 63, 64, 65 BRG Divisor Latch registers 63 BRG Divisor Latch Registers—High Byte 65 BRG Divisor Latch Registers—Low Byte 64 BRG divisor registers 63 BRG divisor value 63, 64, 65, 66 BRG output frequency 63 BRGx 63 BRGx DLR_H 63, 64, 65 BRGx DLR H register 74 BRGx DLR L 63, 64, 65 BRGx DLR L register 65, 73 BURST mode 140, 141, 142, 145 bus arbitration 92, 96 **Bus Arbitration Overview 92**

Bus Clock Speed 111 Bus Enable bit 107 bus request signal 22 BUSACK 20 BUSREQ 20, 22 Byte Format 93 byte transfer 92, 93, 97 byte transfer, cycle-steal 142

С

C register 116 CALC Bank 118 CALC bank 114, 116, 117, 119, 121, 124, 127, 134, 135 calc stat field 117 CGE bit 103 Characteristics, electrical Absolute maximum ratings 180 **Chip Select Registers 55** Chip Select signal 51, 56, 57 Chip Select signals, external 60 Chip Select x Lower Bound Register 55 Chip Select x Upper Bound Register 56 Chip Select/Wait State Generator block 6, 7, 8, 9, 10 Chip Select/Wait State Generator, Register Map 25 Chip Selects, external I/O 23 Chip Selects, I/O 50, 53, 55, 56 Chip Selects, Memory 50 Clear to Send 16, 19, 82 Clear to Send, Delta Status Change of 82 **Clock Divider 34** clock divider 35, 85, 111 clock divider ratio 32 clock frequency 32 clock frequency, 16X 71 clock frequency, 50-MHz external 197 clock frequency, eZ80 CPU 148 clock generator 62, 84



zilog

200

clock period 45, 46, 92 clock prescaler, 4-bit 31 clock source 63 **Clock Synchronization 95** Clock Synchronization for Handshake 97 **Clocking Overview 92** Continuous Mode 33, 34 continuous mode 31, 35, 37, 38 control register values 51, 117 **Control Transfers** 71 **CPHA** 88 CPHA—see SPI Clock Phase 85, 86, 88, 89 CPOL—see SPI Clock Polarity 85, 86, 87, 89 Crystal Oscillator 178 **CS EN 50** CS en 53, 58 **CS IO 50** CS io 53, 56, 57 **CS0** 5 CS1 5 **CS2** 5 **CS3**5 **CTS0** 19 CTS1 16 CTS—see Clear to Send 79, 82 current count value 34, 36 Customer Support 233 CYCLE-STEAL mode 140, 141, 142, 145

D

DATA bank 116, 117, 118, 119, 120, 124, 125, 127, 134, 135 data bank 114 Data Bus 10, 11 data bus 59, 113 Data Carrier Detect 16, 20, 82 Data Carrier Detect, Delta Status Change of 82 Data Ready 81 data ready interrupt, receiver 70 data ready logic, receiver 69 Data Set Ready 16, 19, 82 Data Set Ready, Delta Status Change of 82 Data Terminal Ready 16, 19, 79 Data Transfers 72 Data Validity 93 data stat field 117 **DATA0** 10 **DATA1** 10 DATA2 11 **DATA3** 11 **DATA4** 11 **DATA5** 11 **DATA6** 11 **DATA7** 11 DC Characteristics 181 **DCD0** 20 DCD1 16 DCD-see Data Carrier Detect 79, 82 DCTS—see Delta Status Change of Clear to Send 82 DDCD-see Delta Status Change of Data Carrier Detect 82 **DDSR** 82 DDSR—see Delta Status Change of Data Set Ready 82 debugging 147 Defining A New Calculation As READY 116 Defining The DATA Bank As EMPTY 116 Delta Status Change of Clear to Send 82 Delta Status Change of Data Carrier Detect 82 Delta Status Change of Data Set Ready 82 **Direct Memory Access 140** divisor count, 16-bit 64, 65 **DMA** 140 **DMA Channel Priorities 142 DMA Control Registers** 142 DMA controller 140, 141, 142, 145 DMA controllers, internal 22





DMA Controllers, Register Map 28 **DMA Interrupts** 142 DMA Programming 140 DMA Transfer Modes 141 **Document Information 197 Document Number Description** 197 downcounter 36 downcounter, 16-bit 31, 63 DR Bit 43 DR—see Data Ready 69, 81 DSP operations 121 **DSR0** 19 **DSR1**16 DSR—see Data Set Ready 79, 82 **DTR0** 19 **DTR1**16 DTR—see Data Terminal Ready 79 dual-edge-triggered interrupt mode 45, 47 dual-port MACC RAM 59, 113, 125

Ε

edge-detected interrupt 47 edge-selectable interrupt 47 edge-triggered interrupt mode 45 edge-triggered interrupt source 47 **Edge-Triggered Interrupts** 47 EI, Op Code Map 172 ENAB 107, 108 Enabling 40 Enabling And Disabling The WDT 40 Ending Program Counter 138, 139 end-of-count value, PRT 32, 33, 34, 35 **Environmental Flow 196** ERR bit 71, 80 Exchange Instructions 168 **Extended Temperature 196** External I/O Read Timing 188 External I/O Write Timing 189 **External Memory Read Timing 185** External Memory Write Timing 186

external NMI signal 12 external pull-down resistor 44 eZ80 CPU 1 eZ80 CPU Core 30 eZ80 CPU Core Features 30 eZ80 CPU Core Overview 30 eZ80 CPU instruction set 30 eZ80190 Webserver 1, 178

F

f 43, 63, 82, 107, 121 Fall Time, system clock 184 FAST mode 112 fast mode 92, 111 Features 1 FE—see Framing Error 80 FIFO mode 68, 70 Figure 36 182 four-wire interface 84 framing error 74, 80 framing error 69 full-duplex 84 full-duplex transmission 87

G

General Call Address 105 General Call address 110 general call address 92, 103, 104, 107 General-Purpose I/O Port Input Sample Timing 192 General-Purpose I/O Port Output Timing 193 General-Purpose Input/Output Ports, Register Map 24 GND 2 GPIO 43 GPIO control register bits 43 GPIO Control Registers 47



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



Interrupt Enable bit (IEN) 106 interrupt enable flag 34 Interrupt Enable Flag 1 165 Interrupt Enable Register, UART0 26 Interrupt Enable Register, UART1 27 Interrupt Enable Register, UARTx 74 Interrupt Enable, PRT 33, 34 interrupt input 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22 interrupt mode ,single-edge-triggered 47 interrupt mode, dual edge-triggered 45 interrupt mode, dual-edge-triggered 47 interrupt mode, level-sensitive 45 interrupt mode, single- or dual-edge-triggered 47 interrupt mode, single-edge-triggered 45 interrupt service request 34 interrupt service request, DMA 145 Interrupt Service Routine 137 interrupt service routine 138, 139 interrupt, edge-detected 47 interrupt, low-level 46 **IORQ** 5, 12 irq en 34, 35, 88, 89 **ISR** 137

L

least significant bit 69, 108, 121, 132, 148 least significant byte 122, 123, 125, 132, 159 left-shift 121 level-sensitive interrupt 45, 47 Level-Triggered Interrupts 46 level-trigaered interrupts 46 Line break detection and generation 67 line status error 70 line status interrupt 70, 72, 74 Load Instructions 169 Logical Instructions 169 LOOP-BACK mode 79, 82 low-level interrupt 46

LSB 138, 139 lsb 36, 38, 97, 98, 100, 103

Μ

MACC Accumulator Byte 0 Register 132 MACC Accumulator Byte 1 Register 133 MACC Accumulator Byte 2 Register 133 MACC Accumulator Byte 3 Register 134 MACC Accumulator Byte 4 Register 134 MACC Control Register 130 MACC Dual Bank Operation 117 MACC Length Register 129 MACC Overview 113 MACC RAM 125 MACC RAM Address Indexing 125 MACC RAM, dual-port 59 MACC Status Register 135 MACC x DATA Ending Address Register 128 MACC x DATA Reload Address Register 128 MACC y DATA Ending Address Register 130 MACC y DATA Reload Address Register 130 MACC AC0 28, 116, 121, 123 MACC AC0 register 132 MACC AC1 28, 116, 121, 123, 133 MACC AC2 28, 116, 121, 123, 133 MACC AC3 28, 116, 121, 122, 124, 134 MACC AC4 28, 116, 119, 120, 121, 122, 124, 134 MACC CTL 28, 116, 119, 121, 124, 131 MACC CTL register 121, 123 MACC LENGTH 28, 117, 124 MACC LENGTH register 126 MACC STAT 28, 124 MACC STAT register 117, 119, 135 MACC xEND 28, 117, 124, 126 MACC xEND register 126, 128



zilog

204

MACC xRELOAD 28, 117, 124, 126 MACC xRELOAD register 126, 128 MACC xSTART 28, 117, 124, 125 MACC xSTART register 126, 127 MACC vEND 28, 117, 124, 126, 130 MACC yEND register 130 MACC yRELOAD 28, 117, 124, 126, 130 MACC ySTART 28, 117, 124, 125 MACC vSTART register 129 Maskable interrupt 165 maskable interrupt 12 Master In Slave Out 14, 17, 84 MASTER mode 92, 103, 107, 109, 110, 111, 112 Master Mode Start bit 107 Master Mode START condition 108 Master Mode start transmit condition 108 Master Mode Stop bit 100, 107 Master Mode STOP condition 108 Master Mode STOP TRANSMIT condition 108 MASTER mode, SPI 87 Master Out Slave In 15, 18, 85 Master Receive 92, 100 Master Receive Status Codes for Data Bytes, I2C 102 Master Receive Status Codes, I2C 101 master receiver 94, 102 Master Transmit 97 master transmit 102 MASTER TRANSMIT mode 92, 97 Master Transmit Status Codes for Data Bytes, I2C 100 Master Transmit Status Codes, I2C 98 Master Transmit Status Codes, I2C 10-Bit 99 master transmitter 103 master en 88, 89 MBASE 138, 155, 160, 162, 165 MBASE offset value 53 Memory and I/O Chip Selects 50

Memory Chip Select Example 51 Memory Chip Select Operation 50 Memory Chip Select Priority 51 Memory Chip Selects 50 Memory mode suffixes 161 Memory Request 5 memory space 6, 7, 8, 9, 10, 50, 53 memory space, eZ80 CPU 113 **MISO0** 17 **MISO1** 14 MISO—see Master In Slave Out 84 Mode Fault 88 modem 69 modem control logic 69 modem control output 79 Modem Status 75 Modem status 70 modem status 72, 78 modem status input ports 79 modem status inputs 69 Modem Status Interrupt, UART 71 Modem Status Register, UART 26, 27, 81 modem status signal 16, 17, 19, 20 MODF—see Mode Fault 85, 88, 90 Module Reset 71 MOSI 15, 18, 85, 86, 88 **MOSI0** 18 **MOSI1** 15 MOSI-see Master Out Slave In 84 most significant bit 69, 84, 85, 94, 106, 108, 123, 148, 152 most significant byte 37, 38, 116, 122, 124, 125.134 MREQ 5 MSB 37, 38 msb 37, 38, 94, 131, 152 multimaster conflict 88, 90 Multiply-Accumulator 59, 115, 116, 117, 121, 124, 135 Multiply-Accumulator Basic Operation 114 Multiply-Accumulator Control And Data



zilog .

205

Registers 127 Multiply-Accumulator RAM 60 Multiply-Accumulator, Register Map 28

Ν

NACK 92, 94, 98, 99, 101, 102, 104, 107, 109, 110 **NMI** 12 NMI input 12 NMI signal 12 nmi out 40, 41 nmi out bit 40 nmi out flag 41 no error condition 72 NOISE 121 NOISE bit 132 noise bit 131 NOISE field 121 NOISE value 122 Nonmaskable Interrupt 12, 30 Nonmaskable interrupt 170 nonmaskable interrupt 39, 40, 41 Not Acknowledge 94

0

OE—see overrun error 81 On-chip peripherals 53 on-chip peripherals 23 On-chip RAM 61 on-chip RAM 50, 59, 60 On-Chip RAM Control, Register Map 25 Op Code maps 172 open-drain I/O 12, 44 open-drain output 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 44, 92 open-source I/O 44 open-source mode 44 open-source output 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22 Operating Modes 97 Operation Of The eZ80190 Webserver During ZDI Breakpoints 153 Ordering Information 195, 196 Oscillator Input 20 Oscillator Output 20 OTI2R block 115, 116 OTI2R instruction 116, 119, 124 OUT_SHIFT Function 123 OUT1—see modem control output 79, 82 OUT2—see modem control output 79, 82 overrun 67, 91 overrun condition 70, 91 overrun error 69, 74, 81

Ρ

PA0 21 PA1 21 PA2 21 PA3 21 PA4 21 PA5 21 **PA6** 22 **PA7** 22 parity 77 parity bit 69, 74, 77, 80 parity bit generation and detection 67 parity checker 69 parity error 69, 77, 80 parity generator 68, 69 Part Number Description 196 **PB0** 12 **PB1**13 **PB2** 13 **PB3** 13 **PB4** 13 **PB5**13 **PB6**13 **PB7** 14 PC0 14

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




Resetting the I2C Registers 104 RETI 138, 170, 175 RETN 170, 175 **Retrieve A Calculation 124 RI0** 20 **RI1**17 right-shift 123, 131 Ring Indicator 17, 20, 82 Ring Indicator, Trailing Edge Change on 82 rise time, RC 12 Rise Time, system clock 184 RI—see Ring Indicator 69, 79, 82 Rotate and Shift Instructions 170 RST 170, 172 rst en 36 rst en bit 37, 38 rst flag bit 40 RTS0—see Request to Send 18 RTS1—see Request to Send 15 RTS—see Request to Send 79 **RXD** 74 **RXD** input signal 72 RXD input start bit 69 **RXD signal** 69 **RxD0** 18

S

RxD115

sarx_ctl 143 Schmitt trigger 12 Schmitt Trigger Input 12 SCK 85 SCK0 18 SCK1 15 SCK—see Serial Clock 84 SCL 92, 93, 94, 95, 96, 97, 110 SCL0 17 SCL1 14 SDA 92, 93, 94, 96, 103 SDA0 18 **SDA1**15 selectable taps 31 serial bus 91 serial bus, I2C 62 serial bus, SPI 90 Serial Clock 85, 92 Serial Clock, I2C 14, 17 Serial Clock, SPI 15, 18 serial communication controllers 62, 63, 64 Serial Data 92 serial data 84, 85 serial data, asynchronous 14, 15, 17, 18 Serial Data, I2C 15, 18 serial transmit/receive 68 Setting Timer Duration 32 Setting Up A New Calculation 124 Signal Direction 5 Single Pass Mode 32 single pass mode 31, 34, 35 single-edge-triggered interrupt mode 45, 47 SLAVE mode 84, 88, 92, 103, 105, 107, 110 Slave Receive 92, 103 SLAVE RECEIVE mode 96 slave receiver 94, 97 Slave Select 16, 19, 85 Slave Transmit 92, 102 Slave Transmit mode 107 slave transmitter 94, 100 Sleep instruction 167 SLP 167, 170, 175 Software Control of the MACC 115 SPI block 84 SPI Clock Phase 85, 86 **SPI Clock Polarity 86 SPI Control Register 89** SPI Flags 88 **SPI Functional Description** 86 SPI interrupt 88 SPI MASTER mode 87 SPI Receive Buffer Register 91 SPI Receive Buffer register 89

eZ80190

Product Specification

zilog

eZ80190 Product Specification



SPI Registers 89 SPI Serial Clock 15, 18 SPI serial clock 85 SPI Signals 84 SPI slave 91 SPI slave device 14, 15, 17, 18, 85 SPI SLAVE mode 88 SPI Status Register 90 SPI Status register 88, 89 SPI Transmit Shift Register 90 SPI Transmit Shift register 88 spi en 88, 89 SPIF 87, 88, 91 spiF 90 SPIx CTL 87, 88 SPIx RBR 89, 91 SPIx SR 85, 87, 88 SPIx TSR 88, 89, 90, 91 **SPR 83** SS 85, 86, 88, 89 SS0 19 SS1 16 SS—see Slave Select 84 STA bit 97, 98, 99, 100, 101, 102, 104, 107, 108, 110 standard load 184 standard mode 92 Standard Temperature 196 Standard, Plastic 197 START and STOP Conditions 93 START bit, ZDI 148 START condition 93, 96, 98, 99, 100, 102, 103, 105, 107, 108, 109, 110, 111, 112 start condition 93, 94, 96 Start Condition, ZDI 149 START signal, ZDI 149, 150 Starting Program Counter 138, 139 STOP condition 93, 94, 96, 100, 102, 103, 107, 110, 111, 112 STOP condition, Master Mode 108 STOP TRANSMIT condition, Master Mode

108 STP 104, 107, 108 STP bit 100, 101, 102, 107, 110 supply ground 44 Supply Voltage 181 supply voltage 44, 92, 180 SYNC RESET 68 system clock 45, 46, 63, 153, 179, 192 System Clock Cycle Time 184 system clock cycles 12, 40, 54 System Clock fall time 184 System Clock Frequency 32 System Clock High Time 184 System Clock Low Time 184 System Clock rise time 184 system clock, eZ80 Webserver 111, 112, 179 System Clock, internal 22 system clock, primary 63 system reset 40, 72

Т

TEMT 80 TERI—see Trailing Edge Change on Ring Indicator 82 **TEST** 17 test mode 17 **THRE 80** THRE bit 68 thre bit 77 time-out period 32 Time-Out Period Selection, WDT 40 time-out period, PRT 32, 34 time-out period, WDT 39, 40 Timer Control Register 35 Timer Data High Byte Register 36 Timer Data Low Byte Register 36 timer interrupt flag 34 **Timer Interrupts 34** Timer Reload High Byte Register 38

eZ80190 **Product Specification**



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

eZ80190 **Product Specification**



WDT time-out period 40, 41 WDT time-out RESET indicator flag 39 WDT CTL register 40 wdt en bit 40 wdt period 40, 41 Webserver ID High Byte Register 163 Webserver ID Low Byte Register 162 Webserver ID Revision Register 163 **WR** 5 WRITE bit 97, 109, 110 write bit 98, 103 Write Collision 88 write collision 87, 90 www.zilog.com 147

Х

X data 125 x DATA 127, 128 x data 125, 129 XIN 20, 178 XOUT 20, 178

Υ

y DATA 129, 130 y data 125, 129

Ζ

Z180 CPU 30 **Z80** 1 Z80 CPU 30 Z80 MEMORY mode 164 Z80 Memory mode 162 Z80 MEMORY mode page 53 Z80 Mode 138, 155, 160 Z80 mode 53, 165 Z80-compatible mode 30 ZCL 12, 148, 149, 150, 151, 156 ZDA 12, 148, 149, 150, 156 ZDI 147, 148

ZDI Address Match Registers 155 ZDI block 147 ZDI Block Read 153 **ZDI Block WRITE 151** ZDI break 153, 155, 157 **ZDI Break Control Register** 156 **ZDI Break Control register** 155 ZDI break mode 165 ZDI breakpoint 153 ZDI Breakpoints 153 ZDI Clock 12 ZDI Clock and Data Conventions 148 ZDI Data 12 ZDI Data pin 148 ZDI data transfer 150 **ZDI Interface** 148 **ZDI Overview** 147 ZDI Read Memory Data Value Register 165 ZDI Read Only Registers 154 **ZDI Read Operations** 152 ZDI Read Register Low, High, and Upper 165 ZDI Read/Write Control Register 159 ZDI Register Addressing 150 **ZDI Register Definitions** 155 **ZDI registers** 150 ZDI Single-Bit Byte Separator 150 ZDI Single-Byte Read 152 **ZDI Single-Byte WRITE 151** ZDI START bit 148 ZDI Start Condition 149 ZDI START signal 149, 150 ZDI Status Register 164 ZDI Write Data Registers 159 **ZDI Write Memory Register** 162 ZDI Write Only Registers 153 **ZDI Write Operations 151** Zilog Debug Interface 12, 30, 147 Zilog Developer Studio 147, 196 **ZPAK** 147 **ZPAK emulator** 148



Customer Support

For answers to technical questions about the product, documentation, or any other issues with Zilog's offerings, please visit Zilog's Knowledge Base at http://www.zilog.com/kb.

For any comments, detail technical questions, or reporting problems, please visit Zilog's Technical Support at <u>http://support.zilog.com</u>.

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

ZiLOG:

EZ80190AZ050EG EZ80190AZ050SG