LogiCORE IP I/O Module (v1.00.a)

DS866 April 24, 2012 **Product Specification**

Introduction

The LogiCORE™ I/O Module is a highly integrated and light-weight implementation of a standard set of peripherals.

The I/O Module is a standalone version of the tightly coupled I/O Module included in the LogiCORE MicroBlaze™ Micro Controller System (MCS). Using the I/O Module, a system equivalent to MicroBlaze MCS can be design using the ISE® Design Suite Embedded Edition.

The I/O Module connects to MicroBlaze through the lmb_v10 bus.

Features

- LMB v1.0 bus interfaces to communicate with MicroBlaze
- I/O Bus
- Interrupt Controller with fast interrupt mode support
- UART
- Fixed Interval Timers
- Programmable Interval Timers
- General Purpose Inputs
- General Purpose Outputs

LogiCORE Facts					
Core Specifics					
Supported Device Family(1)	Spartan®-3, Spartan-6, Virtex®-4, Virtex-5, Virtex-6, Virtex-7, Kintex™-7, Artix™-7, Zynq™-7000				
Supported User Interfaces	Local Memory Bus (LMB), Dynamic Reconfiguration Port (DRP)				
Resources(2)					
Configuration	**LUTs**	**FFs**	**DSP Slices**	**Block RAMs**	
Minimum	40	75	0	0	
Maximum	530	1014	0	0	
Provided with Core					
Documentation	Product Specification				
Design Files	VHDL				
Example Design	Not Provided				
Test Bench	Not Provided				
Constraints File	Not Provided				
Supported SW Driver(3)	Standalone				
Tested Design Tools					
Design Entry Tools	Xilinx Platform Studio (XPS) 14.1				
Simulation(4)	Mentor Graphics ModelSim				
Synthesis Tools	ISE 14.1				
Support					
Provided by Xilinx @ www.xilinx.com/support					

1. For a complete listing of supported families, see the release notes for this core.
2. Resources listed here are for Virtex-6 devices. For more complete device performance numbers, see Table 36.
3. Standalone driver details can be found in the EDK or SDK directory (*<install_directory>*/doc/usenglish/xilinx_drivers.htm)
4. For the supported versions of the tools, see the ISE Design Suite 14: Release Notes Guide.

Functional Description

The I/O Module is a light-weight implementation of a set of standard I/O functions commonly used in a MicroBlaze processor sub-system. The input/output signals of the I/O Module are shown in Figure 1. The detailed list of signals are listed and described in Table 4. See the description of LMB Signals in the MicroBlaze Bus Interfaces chapter in the *MicroBlaze Processor Reference Guide* [Ref 1].

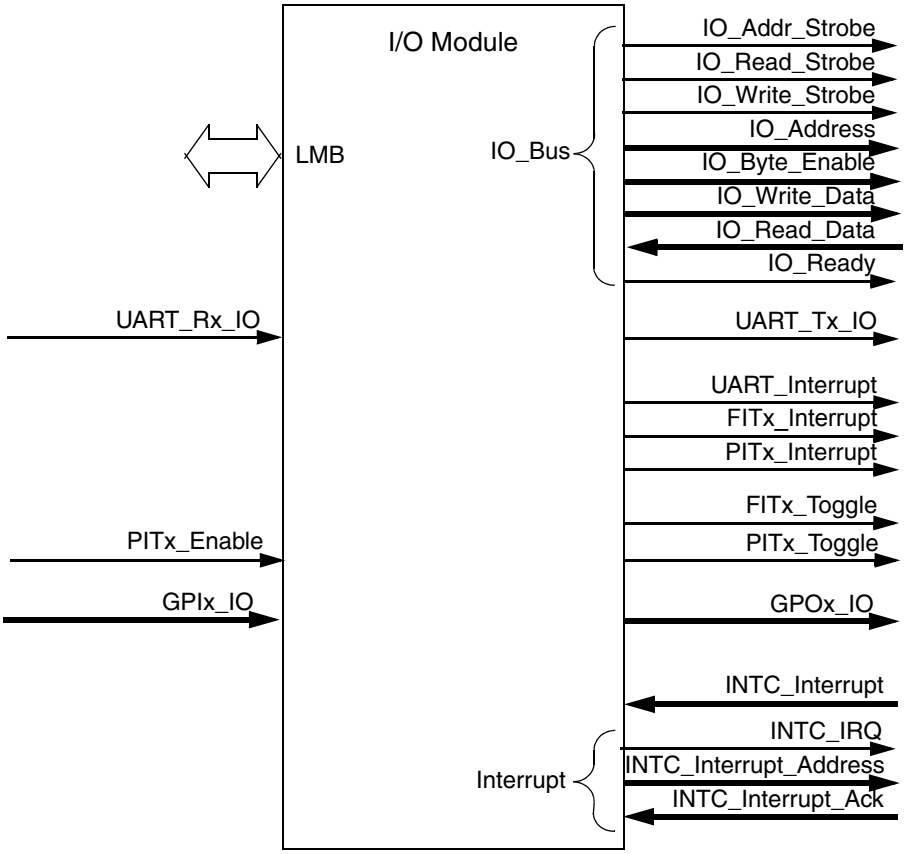

Figure 1: **I/O Module Block Diagram**

In a MicroBlaze system the I/O Module would typically be connected according to Figure 2.

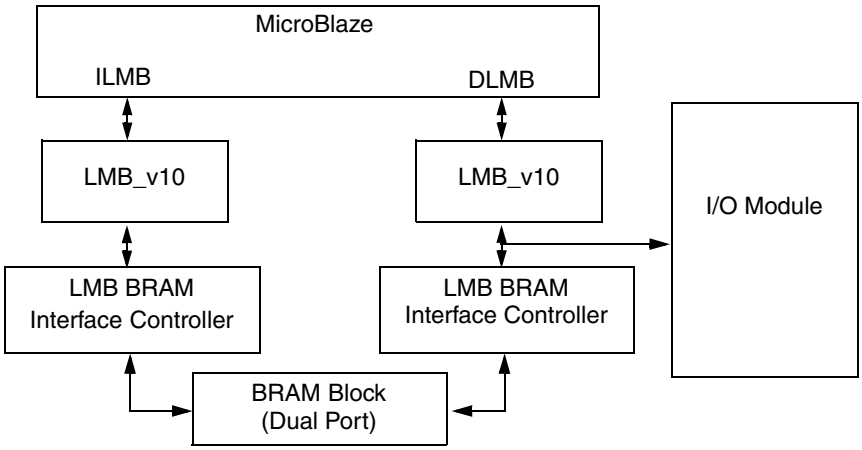

Figure 2: **Typical MicroBlaze System**

I/O Bus

The I/O Bus provides a simple bus for accessing to external modules using MicroBlaze Load/Store instructions. The I/O Bus is mapped at address C_IO_BASEADDR-C_IO_HIGHADDR in the MicroBlaze memory space, with the I/O Bus address directly reflecting the byte address used by MicroBlaze Load/Store instructions. I/O Bus data is 32-bit wide, with byte enables to write byte and half-word data.

The I/O Bus has a ready handshake to handle different waitstate needs, from IO_Ready asserted the cycle after the IO_Addr_Strobe is asserted to as many cycles as needed. There is no time-out on the I/O Bus and MicroBlaze is stalled until IO_Ready is asserted. IO_Address, IO_Byte_Enable, IO_Write_Data, IO_Read_Strobe, IO_Write_Strobe are only valid when IO_Addr_Strobe is asserted. For read access IO_Read_Data is sampled at the rising Clk edge, when the slave has asserted IO_Ready.

I/O Bus read and write transactions can be found in the two following timing diagrams in Figure 3 and Figure 4.

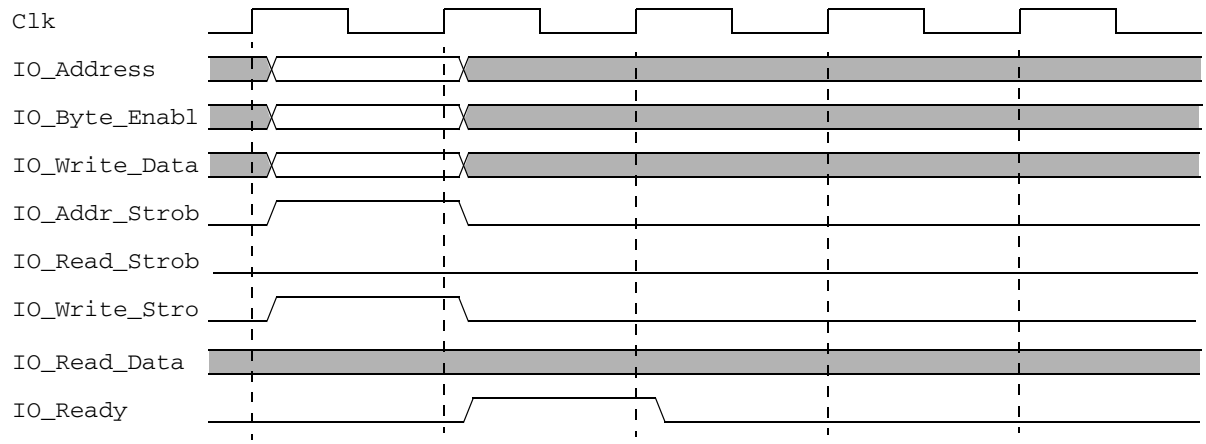

Figure 3: **I/O Bus Write**

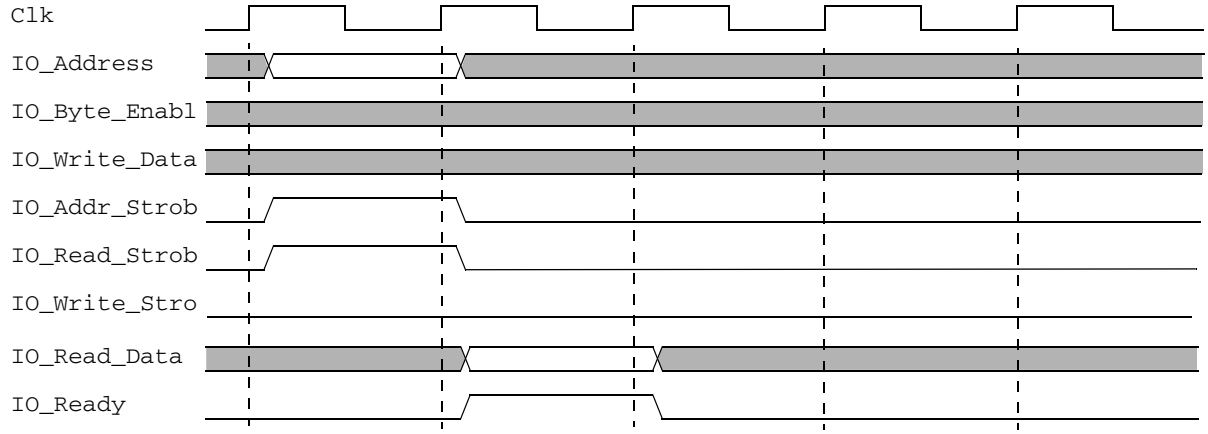

Figure 4: **I/O Bus Read**

The byte enable signals indicate which byte lanes of the data bus contain valid data. Valid values for IO_Byte_Enable are shown in Table 1. The IO_Byte_Enable signal should be used instead of the two least significant bits of the IO_Address to decode byte and halfword accesses, to ensure that byte and halfword accesses are correctly decoded independent of MicroBlaze endianess.

Table 1: **Valid Values for IO_Byte_Enable[3:0]**

IO_Byte_Enable	IO_Data_Write and IO_Data_Read Byte Lanes Used			
[3:0]	[31:24]	[23:16]	[15:8]	[7:0]
0001				●
0010			●	
0100		●		
1000	●			
0011			●	●
1100	●	●		
1111	●	●	●	●

The I/O Bus is fully compatible with the Xilinx Dynamic Reconfiguration Port.(DRP). This configuration port supports partial dynamic reconfiguration of functional blocks, such as CMTs, clock management, XADC, serial transceivers, and the PCIe® block.

The nominal connection of the I/O Bus to the DRP is illustrated in Table 2.

Table 2: **Mapping of the I/O Bus to the Dynamic Reconfiguration Port**

MicroBlaze MCS Signal	DRP Signal	Note
Clk	DCLK	
IO_Addr_Strobe	DEN	
IO_Read_Strobe	-	Not used by DRP
IO_Write_Strobe	DWE	
IO_Address[m+2:2]	DADDR[m:0]	Uses 32-bit word access for DRP
IO_Byte_Enable	-	Only 32-bit word accesses used for DRP
IO_Write_Data[n:0]	DI[n:0]	Data width depends on DRP (n < 32)
IO_Read_Data[n:0]	DO[n:0]	Data width depends on DRP (n < 32)
IO_Ready	DRDY	

For a detailed description of the DRP, see the *7 Series FPGAs Configuration User Guide* [Ref 2].

UART

The Universal Asynchronous Receiver Transmitter (UART) interface provides the controller interface for asynchronous serial data transfers. Features supported include:

- One transmit and one receive channel (full duplex)
- Configurable number of data bits in a character (5-8)
- Configurable parity bit (odd or even)
- Configurable baud rate

The UART performs parallel-to-serial conversion on characters received through LMB and serial-to-parallel conversion on characters received from a serial peripheral. The UART is capable of transmitting and receiving 8, 7, 6 or 5-bit characters, with 1-stop bit and odd, even or no parity. The UART can transmit and receive independently.

The device can be configured and its status can be monitored via the internal register set. The UART also asserts the UART_Interrupt output when the receiver becomes non-empty, when the transmitter becomes empty or when an error condition has occurred. The individual interrupt events are connected to the Interrupt Controller of the I/O Module and can be used to assert the INTC_IRQ output signal.

Fixed Interval Timer, FIT

The Fixed Interval Timer generates a strobe (interrupt) signal at fixed intervals. The Fixed Interval Timer asserts the output signal FITx_Interrupt one clock cycle every C_FITx_NO_CLOCKS. Operation begins immediately after FPGA configuration and the clock is running. The FITx_Toggle output signal is toggled each time FITx_Interrupt is asserted, creating a 50% duty cycle output with twice the FITx_Interrupt period. Using the parameter C_FITx_INTERRUPT, the FIT can be connected to the Interrupt Controller of the I/O Module and used for generating interrupts every time the strobe occurs.

Programmable Interval Timer, PIT

The Programmable Interval Timer, PIT, has a configurable width from 1 to 32. The PIT operation and period are controlled by software.

The PITx_Interrupt output signal is asserted one clock cycle when the timer lapses. The timer can be used in continuous mode, where the timer reloads automatically when it lapses. In continuous mode, the period between two PITx_Interrupt assertions is the value in PITx Preload Register + 2 count events.

The PIT can also be used in one-shot mode, where the timer stops when it has reached zero. The timer is implemented by means of a counter that is pre-loaded with the timer value and then decremented. When the counter reaches zero, the timer lapses, and the interrupt signal is generated. The timer starts counting when it is enabled by setting the EN bit in the PITx Control Register.

The PITx_Toggle output signal is toggled each time PITx_Interrupt is asserted, creating a 50% duty cycle output with twice the PITx_Interrupt period when the timer is operated in continuous mode.

The value of the counter that implements the timer can be read by software if the C_PITx_Readable parameter is enabled.The PIT can have a pre-scaler connected from any FITx, PITx, or External. The pre-scaler is selected by the C_PITx_PRESCALER parameter. The PIT has no pre-scaler by default. If External is selected the input signal PITx_Enable is used as pre-scaler. Selecting External as pre-scaler can also be used to measure the width in clock cycles of a signal connected to the PITx_Enable input.

Using the parameter C_PITx_INTERRUPT, the PIT can be connected to the Interrupt Controller of the I/O Module and used for generating interrupts every time it lapses.

General Purpose Output, GPO

The General Purpose Output, GPO, drives I/O Module GPO output signals defined by the value of the GPOx register, programmable from software. The width of the GPOx is defined by the C_GPOx_SIZE and the initial value is defined by the parameter C_GPOx_INIT. When the GPOx register is written, the value of the GPOx output signals will change accordingly.

General Purpose Input, GPI

The General Purpose Input, GPI, makes it possible for software to sample the value of the I/O Module GPI input signals by reading the GPIx register. The width of GPIx is defined by the parameter C_GPIx_SIZE.

Interrupt Controller INTC

The Interrupt Controller handles both I/O module internal interrupt events and external ones. The internal interrupt events originate from the UART and the Fixed or Programmable Interval Timers. For an internal interrupt to be generated on the INTC_IRQ output, the corresponding I/O Module parameter needs to be set, e.g. C_UART_RX_INTERRUPT=1, and that particular interrupt needs to be enabled in the Interrupt Enable Register.

The Interrupt Controller supports up to 16 external interrupts using the INTC_Interrupt inputs. The number of external interrupts is defined by the parameter C_INTC_INTR_SIZE. The external interrupt signals can be individually configured as either edge or level sensitive by the C_INTC_LEVEL_EDGE parameter. The polarity of the external interrupt signals can be individually configured to be either active high (rising edge) or low (falling edge) by the C_INTC_POSITIVE parameter. Interrupt events for external interrupt sources are generated according to Table 3.

Table 3: **Interrupt Event Generation**

C_INTC_LEVEL_EDGE(x)	C_INTC_POSITIVE(x)	INTC_Interrupt(x) Input
0	0	0
0	1	1
1	0	1 -> 0
1	1	0 -> 1
0	0	0

The current status of all interrupt sources can be read from the Interrupt Status Register. The current status of all enabled interrupts can be read from the Interrupt Pending Register.

An interrupt is cleared in both the Interrupt Status and Interrupt Pending Registers by writing to the Interrupt Acknowledge Register, with bits set corresponding to the interrupts that should be cleared.

Either normal or fast interrupt mode can be used, based on latency requirement. Fast interrupt mode is available when the parameter C_INTC_HAS_FAST is set, and is enabled for an interrupt by setting the corresponding bit in the Interrupt Mode Register (IRQ_MODE). In this case, the Interrupt Controller drives the interrupt vector address of the highest priority interrupt on the INTC_Interrupt_Address port, along with INTC_IRQ. The generated interrupt is cleared based on acknowledge received from the processor via the INTC_Interrupt_Ack port. The processor sends 0b01 on this port when the interrupt is being acknowledged by the processor (that is, when branching to the interrupt service routine), sends 0b10 when executing a return from interrupt instruction in the interrupt service routine, and sends 0b11 when interrupts are re-enabled. The bit in IRQ_STATUS corresponding to the interrupt is cleared when 0b10 or 0b11 is seen on the port. The interrupt vector address for each interrupt is stored in the corresponding IRQ_VECTOR register.

I/O Module I/O Signals

The I/O ports and signals for the I/O Module are listed and described in Table 4.

Table 4: **I/O Module I/O Signals**

Port Name	MSB:LSB	I/O	Description
LMB Signals			
LMB_ABus	0:C_LMB_AWIDTH-1	I	LMB Address Bus
LMB_WriteDBus	0:C_LMB_DWIDTH-1	I	LMB Write Data Bus
LMB_ReadStrobe		I	LMB Read Strobe
LMB_AddrStrobe		I	LMB Address Strobe
LMB_WriteStrobe		I	LMB Write Strobe
LMB_BE	0:C_LMB_DWIDTH/8-1	I	LMB Byte Enable Bus
SI_DBus	0:C_LMB_DWIDTH-1	O	LMB Read Data Bus
SI_Ready		O	LMB Data Ready
SI_Wait		O	LMB Wait
SI_CE		O	LMB Correctable Error
SI_UE		O	LMB Uncorrectable Error
I/O Bus Signals			
IO_Addr_Strobe		O	Address strobe signals valid I/O Bus output signals
IO_Read_Strobe		O	I/O Bus access is a read
IO_Write_Strobe		O	I/O Bus access is a write
IO_Address	31:0	O	Address for access
IO_Byte_Enable	3:0	O	Byte enables for access
IO_Write_Data	31:0	O	Data to write for I/O Bus write access
IO_Read_Data	31:0	I	Read data for I/O Bus read access
IO_Ready		I	Ready handshake to end I/O Bus access
UART Signals			
UART_Rx_IO		I	Receive Data
UART_Tx_IO		O	Transmit Data
UART_Interrupt		O	UART Interrupt
FIT Signals			
FITx_Interrupt(1)		O	FITx timer lapsed
FITx_Toggle(1)		O	Inverted FITx_Toggle when FITx timer lapses
PIT Signals			
PITx_Enable(1)		I	PITx count enable when C_PITx_PRESCALER = External
PITx_Interrupt(1)		O	PITx timer lapsed
PITx_Toggle(1)		O	Inverted PITx_Toggle when PITx lapses
GPO Signals			
GPOx(1)	[C_GPOx_SIZE - 1]:0	O	GPOx Output

Table 4: **I/O Module I/O Signals *(Cont'd)***

Port Name	MSB:LSB	I/O	Description
GPI Signals			
GPIx[1]	[C_GPIx_SIZE - 1]:0	I	GPIx Input
INTC Signals			
INTC_Interrupt	0:[C_INTC_INTR_SIZE - 1]	I	External interrupt inputs
INTC_IRQ		O	Interrupt Output
INTC_Interrupt_ Address	[C_INTC_ADDR_WIDTH-1]:0	O	Interrupt Address Output
INTC_Interrupt_Ack	1:0	I	Interrupt Acknowledge Input

1. x = 1, 2, 3 or 4

I/O Module Parameters

To allow the user to obtain an I/O Module that is uniquely tailored a specific system, certain features can be parameterized in the I/O module design. This allows the user to configure a design that only utilizes the resources required by the system, and operates with the best possible performance. The features that can be parameterized in I/O Module designs are shown in Table 5.

Table 5: **I/O Module Parameters**

Parameter Name	Feature/Description	Allowable Values	Default Value	VHDL Type
C_FAMILY[1]	FPGA Architecture	Supported architectures	"virtex5"	string
C_FREQ[1]	Frequency of CLK input		100000000	integer
C_INSTANCE[1]	Instance name	Any legal VHDL string	"iomodule"	string
C_BASEADDR	LMB I/O Module Register Base Address	Valid Address Range[2]	0xFFFFFFFF	std_logic_vector
C_HIGHADDR	LMB I/O Module Register High Address	Valid Address Range[2]	0x00000000	std_logic_vector
C_MASK	LMB I/O Module Register Address Space Decode Mask	Valid decode mask[5]	0x00800000	std_logic_vector
C_IO_HIGHADDR	LMB I/O Module I/O Bus Base Address	Valid Address Range[2]	0xFFFFFFFF	std_logic_vector
C_IO_LOWADDR	LMB I/O Module I/O Bus Address	Valid Address Range[2]	0x00000000	std_logic_vector
C_IO_MASK	LMB I/O Module I/O Bus Address Space Decode Mask	Valid decode mask[5]	0x00800000	std_logic_vector
C_LMB_AWIDTH	LMB Address Bus Width	32	32	integer
C_LMB_DWIDTH	LMB Data Bus Width	32	32	integer
I/O Bus Parameter				
C_USE_IO_BUS	Use I/O Bus	0 = Not Used 1 = Used	0	integer

Table 5: **I/O Module Parameters *(Cont'd)***

Parameter Name	Feature/Description	Allowable Values	Default Value	VHDL Type
UART Parameters				
C_USE_UART_RX	Use UART Receive	0 = Not Used 1 = Used	0	integer
C_USE_UART_TX	Use UART Transmit	0 = Not Used 1 = Used	0	integer
C_UART_BAUDRATE	Baud rate of the UART in bits per second	integer (e.g. 115200)	9600	integer
C_UART_DATA_BITS	The number of data bits in the serial frame	5 - 8	8	integer
C_UART_USE_PARITY	Determines whether parity is used or not	0 = No Parity 1 = Use Parity	0	integer
C_UART_ODD_PARITY	If parity is used, determines whether parity is odd or even	0 = Even Parity 1 = Odd Parity	0	integer
C_UART_RX_INTERRUPT	Use UART RX Interrupt in INTC	0 = Not Used 1 = Used	0	integer
C_UART_TX_INTERRUPT	Use UART TX Interrupt in INTC	0 = Not Used 1 = Used	0	integer
C_UART_ERROR_INTERRUPT	Use UART ERROR Interrupt in INTC	0 = Not Used 1 = Used	0	integer
FIT Parameters				
C_USE_FITx[(2)]	Enable implementation of FIT	0 = Not Used 1 = Used	0	integer
C_FITx_No_CLOCKS[(2)]	The number of clock cycles between strobes	>2	6216	integer
C_FITx_INTERRUPT[(2)]	Use FITx_Interrupt in INTC	0 = Not Used 1 = Used	0	integer
PIT Parameters				
C_USE_PITx[(2)]	Enable implementation of PIT	0 = Not Used 1 = Used	0	integer
C_PITx_SIZE[(2)]	Size of PITx counter	1 - 32	1	integer
C_PITx_READABLE[(2)]	Make PITx counter software readable	0 = Not SW readable 1 = SW readable	1	integer
C_PITx_PRESCALER[(2)(3)]	Select PITx prescaler	0 = No prescaler 1 = FIT1 2 = FIT2 3 = FIT3 4 = FIT4 5 = PIT1 6 = PIT2 7 = PIT3 8 = PIT4 9 = External	0	integer
C_PITx_INTERRUPT[(2)]	Use PITx_Interrupt in INTC	0 = Not Used 1 = Used	0	integer

Table 5: **I/O Module Parameters *(Cont'd)***

Parameter Name	Feature/Description	Allowable Values	Default Value	VHDL Type
GPO Parameters				
C_USE_GPOx[2]	Use GPOx	0 = Not Used 1 = Used	0	integer
C_GPOx_SIZE[2]	Size of GPOx	1 - 32	32	integer
C_GPOx_INIT[2]	Initial value for GPOx	Fit Range (31:0)	all zeros	std_logic_vector
GPI Parameters				
C_USE_GPIx[2]	Use GPIx	0 = Not Used 1 = Used	0	integer
C_GPIx_SIZE[2]	Size of GPIx	1 - 32	32	integer
INTC Parameters				
C_INTC_USE_EXT_INTR	Use I/O Module external interrupt inputs	0 = Not Used 1 = Used	0	integer
C_INTC_INTR_SIZE	Number of external interrupt inputs used	1 - 16	1	integer
C_INTC_LEVEL_EDGE	Level or edge triggered for each external interrupt	For each bit: 0 = Level 1 = Edge	level	std_logic_vector
C_INTC_POSITIVE	Polarity for each external interrupt	For each bit: 0 = active low 1 =active high	active high	std_logic_vector
C_INTC_HAS_FAST	Use fast interrupt mode	0 = Not Used 1 = Used	0	integer
C_INTC_ADDR_WIDTH	Interrupt Address width	5 - 32	32	integer

1. Values automatically populated by tool.
2. x =1, 2, 3 or 4.
3. Selecting PIT prescaler the same as PITx is illegal, e.g. PIT2 cannot be prescaler to itself.
4. The range specified by BASEADDR and HIGHADDR must comprise a complete, contiguous power-of-two range, such that range = 2^n, and the n least significant bits of BASEADDR must be zero.
5. The decode mask determines which bits are used by the LMB decode logic to decode a valid access to LMB.

Parameter - Port Dependencies

The width of many of the I/O Module signals depends on design parameters. The dependencies between the design parameters and I/O signals are shown in Table 6.

Table 6: **Parameter-Port Dependencies**

Parameter Name	Ports (Port width depends on parameter)
C_INTC_INTR_SIZE	INTC_Interrupt
C_INTC_ADDR_WIDTH	INTC_Interrupt_Address
C_GPO1_SIZE	GPO1
C_GPO2_SIZE	GPO2
C_GPO3_SIZE	GPO3
C_GPO4_SIZE	GPO4
C_GPI1_SIZE	GPI1
C_GPI2_SIZE	GPI2
C_GPI3_SIZE	GPI3
C_GPI4_SIZE	GPI4

I/O Module Register Descriptions

Table 7: **I/O Module Register Address Map**

Base Address + Offset (hex)	Register	Access Type	Description
C_BASEADDR + 0x0	UART_RX	R	UART Receive Data Register
C_BASEADDR + 0x4	UART_TX	W	UART Transmit Data Register
C_BASEADDR + 0x8	UART_STATUS	R	UART Status Register
C_BASEADDR + 0xC	IRQ_MODE	W	Interrupt Mode Register
C_BASEADDR + 0x10	GPO1	W	General Purpose Output 1 Register
C_BASEADDR + 0x14	GPO2	W	General Purpose Output 2 Register
C_BASEADDR + 0x18	GPO3	W	General Purpose Output 3 Register
C_BASEADDR + 0x1C	GPO4	W	General Purpose Output 4 Register
C_BASEADDR + 0x20	GPI1	R	General Purpose Input 1 Register
C_BASEADDR + 0x24	GPI2	R	General Purpose Input 2 Register
C_BASEADDR + 0x28	GPI3	R	General Purpose Input 3 Register
C_BASEADDR + 0x2C	GPI4	R	General Purpose Input 4 Register
C_BASEADDR + 0x30	IRQ_STATUS	R	Interrupt Status Register
C_BASEADDR + 0x34	IRQ_PENDING	R	Pending Interrupt Register
C_BASEADDR + 0x38	IRQ_ENABLE	W	Interrupt Enable Register
C_BASEADDR + 0x3C	IRQ_ACK	W	Interrupt Acknowledge Register
C_BASEADDR + 0x40	PIT1_PRELOAD	W	PIT1 Preload Register
C_BASEADDR + 0x44	PIT1_COUNTER	R	PIT1 Counter Register
C_BASEADDR + 0x48	PIT1_CONTROL	W	PIT1 Control Register
C_BASEADDR + 0x4C	Reserved		

Table 7: **I/O Module Register Address Map *(Cont'd)***

Base Address + Offset (hex)	Register	Access Type	Description
C_BASEADDR + 0x50	PIT2_PRELOAD	W	PIT2 Preload Register
C_BASEADDR + 0x54	PIT2_COUNTER	R	PIT2 Counter Register
C_BASEADDR + 0x58	PIT2_CONTROL	W	PIT2 Control Register
C_BASEADDR + 0x5C	Reserved		
C_BASEADDR + 0x60	PIT3_PRELOAD	W	PIT3 Preload Register
C_BASEADDR + 0x64	PIT3_COUNTER	R	PIT3 Counter Register
C_BASEADDR + 0x68	PIT3_CONTROL	W	PIT3 Control Register
C_BASEADDR + 0x6C	Reserved		
C_BASEADDR + 0x70	PIT4_PRELOAD	W	PIT4 Preload Register
C_BASEADDR + 0x74	PIT4_COUNTER	R	PIT4 Counter Register
C_BASEADDR + 0x78	PIT4_CONTROL	W	PIT4 Control Register
C_BASEADDR + 0x7C	Reserved		
C_BASEADDR + 0x80 - C_BASEADDR + 0xFC	IRQ_VECTOR_0 - IRQ_VECTOR_31	W	Interrupt Address Vector Registers
(C_BASEADDR + 0x100) - C_HIGHADDR	Reserved		
C_IO_BASEADDR - C_IO_HIGHADDR	I/O Bus	RW	Mapped to I/O Bus address output IO_Address

UART Receive Data Register (UART_RX)

A register contains data received by the UART. Reading of this location will result in reading the current word from the register. When a read request is issued without having received a new character, the previously read data will be read again. This register is a read-only register. Issuing a write request to the register will do nothing but generate the write acknowledgement.

The register is implemented if C_USE_UART_RX is set to 1.

Table 8: **UART Receive Data Register (UART_RX) (C_DATA_BITS=8)**

Reserved		UART_RX	
31	8	7	0

Table 9: **UART Receive Data Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31:C_UART_DATA_BITS	-	R	0	Reserved
[C_UART_DATA_BITS-1]:0	UART_RX	R	0	UART Receive Data

UART Transmit Data Register (UART_TX)

A register contains data to be output by the UART. Data to be transmitted is written into this register. This is write only location. Issuing a read request to this register generates the read acknowledgement with zero data. Writing this register when the character has not been transmitted will overwrite previously written data, resulting in loss of data.

The register is implemented if C_USE_UART_TX is set to 1.

Table 10: **UART Transmit Data Register (UART_TX) (C_DATA_BITS=8)**

Reserved		UART_TX	
31	8	7	0

Table 11: **UART Transmit Data Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31:C_UART_DATA_BITS	-	R	0	Reserved
[C_UART_DATA_BITS-1]:0	UART_TX	R	0	UART Transmit Data

UART Status Register (UART_Status)

The UART Status Register contains the status of the receive and transmit registers, and if there are any errors. This is read only register. If a write request is issued to status register it will do nothing but generate write acknowledgement.

The register is implemented if C_USE_UART_RX or C_USE_UART_TX is set to 1.

Table 12: **UART Status Register (UART_Status)**

Reserved		UART_Status	
31	8	7	0

Table 13: **UART Status Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
7	Parity Error	R	0	Indicates that a parity error has occurred after the last time the status register was read. If the UART is configured without any parity handling, this bit is always '0'. The received character is written into the receive register. This bit is cleared when the status register is read. 0 = No parity error has occurred 1 = A parity error has occurred
6	Frame Error	R	0	Indicates that a frame error has occurred after the last time the status register was read. Frame Error is defined as detection of a stop bit with the value 0. The receive character is ignored and not written to the receive register. This bit is cleared when the status register is read. 0 = No Frame error has occurred 1 = A frame error has occurred
5	Overrun Error	R	0	Indicates that a overrun error has occurred since the last time the status register was read. Overrun occurs when a new character has been received but the receive register has not been read. The received character is ignored and not written into the receive register. This bit is cleared when the status register is read. 0 = No interrupt has occurred 1 = Interrupt has occurred
4	-	R	0	Reserved
3	Tx Used	R	0	Indicates if the transmit register is in use 0 = Transmit register is not in use 1= Transmit register is in use
2	-	R	0	Reserved
1	-	R	0	Reserved
0	Rx Valid Data	R	0	Indicates if the receive register has valid data 0 = Receive register is empty 1 = Receive register has valid data

General Purpose Output x Register (GPOx) (x = 1, 2, 3 or 4)

This register holds the value that will be driven to the corresponding bits in the I/O Module GPOx port output signals. All bits in the register are updated when the register is written.

This register is not implemented if the value of C_USE_GPOx is 0.

Table 14: **General Purpose Output x Register (GPOx)**

Reserved		GPOx	
31	C_GPOx_SIZE	C_GPOx_SIZE-1	0

Table 15: **General Purpose Output x Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31:C_GPOx_SIZE	-	-	-	Reserved
[C_GPOx_SIZE-1]:0	GPOx	W	0	Register holds data driven to corresponding bits in the GPO port

General Purpose Input x Register (GPIx) (x=1, 2, 3 or 4)

This register reads the value that is input on the corresponding I/O Module GPIx port input signal bits.

This register is not implemented if the value of C_USE_GPIx is 0.

Table 16: **General Purpose Input x Register (GPIx)**

Reserved		GPIx	
31	C_GPIx_SIZE	C_GPIx_SIZE-1	0

Table 17: **General Purpose Input x Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31:C_GPIx_SIZE	-	R	0	Reserved
[C_GPIx_SIZE-1]:0	GPIx	R	0	Register reads value input on the I/O Module GPIx port input signals

Interrupt Status Register (IRQ_STATUS)

The Interrupt Status Register holds information on interrupt events that have occurred. The register is read-only and the IRQ_ACK register should be used to clear individual interrupts.

Table 18: **Interrupt Status Register (IRQ_STATUS)**

Reserved		INTC_Interrupt		Reserved		Internal Interrupts	
31	C_INTC_EXT_INTR+16	C_INTC_EXT_INTR+15	16	15	11	10	0

Table 19: **Interrupt Status Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31:[C_INTC_EXT_INTR + 16]	-	R	0	Reserved
[C_INTC_EXT_INTR+15]:16	INTC_Interrupt	R	0	I/O Module external interrupt input signal INTC_Interrupt [C_INTC_EXT_INTR-1:0] mapped to corresponding bit positions in IRQ_STATUS
15:11	-	R	0	Reserved
10	FIT4	R	0	FIT4 strobe
9	FIT3	R	0	FIT3 strobe
8	FIT2	R	0	FIT2 strobe
7	FIT1	R	0	FIT1 strobe
6	PIT4	R	0	PIT4 lapsed
5	PIT3	R	0	PIT3 lapsed
4	PIT2	R	0	PIT2 lapsed
3	PIT1	R	0	PIT1 lapsed
2	UART_RX	R	0	UART Received Data
1	UART_TX	R	0	UART Transmitted Data
0	UART_ERR	R	0	UART Error

Interrupt Pending Register (IRQ_PENDING)

The Interrupt Pending Register holds information on enabled interrupt events that have occurred. IRQ_PENDING is the contents of IRQ_STATUS bit-wised masked with the IRQ_ENABLE register. The register is read-only and the IRQ_ACK register should be used to clear individual interrupts.

Table 20: **Interrupt Pending Register (IRQ_PENDING)**

Reserved		INTC_Interrupt		Reserved		Internal Interrupts	
31	C_INTC_EXT_INTR+16	C_INTC_EXT_INTR+15	16	15	11	10	0

Table 21: **Interrupt Pending Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31:[C_INTC_EXT_INTR+16]	-	R	0	Reserved
[C_INTC_EXT_INTR+15]:16	INTC_Interrupt	R	0	I/O Module external interrupt input signal INTC_Interrupt [C_INTC_EXT_INTR-1:0] mapped to corresponding bit positions in IRQ_STATUS
15:11	-	R	0	Reserved
10	FIT4	R	0	FIT4 strobe
9	FIT3	R	0	FIT3 strobe
8	FIT2	R	0	FIT2 strobe
7	FIT1	R	0	FIT1 strobe
6	PIT4	R	0	PIT4 lapsed
5	PIT3	R	0	PIT3 lapsed
4	PIT2	R	0	PIT2 lapsed
3	PIT1	R	0	PIT1 lapsed
2	UART_RX	R	0	UART Received Data
1	UART_TX	R	0	UART Transmitted Data
0	UART_ERR	R	0	UART Error

Interrupt Enable Register (IRQ_ENABLE)

The Interrupt Enable Register enables assertion of the I/O Module interrupt output signal INTC_IRQ by individual interrupt sources. The contents of this register is also used to mask the value of the IRQ_STATUS register when registering enabled interrupts in the IRQ_PENDING register.

Table 22: **Interrupt Enable Register (IRQ_ENABLE)**

Reserved		INTC_Interrupt		Reserved		Internal Interrupts	
31	C_INTC_EXT_INTR+16	C_INTC_EXT_INTR+15	16	15	11	10	0

Table 23: **Interrupt Enable Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31:[C_INTC_EXT_INTR+16]	-	-	0	Reserved
[C_INTC_EXT_INTR+15]:16	INTC_Interrupt	W	0	Enable I/O Module external interrupt input signal INTC_Interrupt(16-C_INTC_EXT_INTR)
15 - 11	-	-	0	Reserved
10	FIT4	W	0	FIT4 interrupt enabled
9	FIT3	W	0	FIT3 interrupt enabled
8	FIT2	W	0	FIT2 interrupt enabled
7	FIT1	W	0	FIT1 interrupt enabled
6	PIT4	W	0	PIT4 interrupt enabled
5	PIT3	W	0	PIT3 interrupt enabled
4	PIT2	W	0	PIT2 interrupt enabled
3	PIT1	W	0	PIT1 interrupt enabled
2	UART_RX	W	0	UART Received Data interrupt enabled
1	UART_TX	W	0	UART Transmitted Data interrupt enabled
0	UART_ERR	W	0	UART Error interrupt enabled

Interrupt Acknowledge Register (IRQ_ACK)

This register is used as a command register for clearing individual interrupts in IRQ_STATUS and IRQ_PENDING registers. All bits written '1' will clear the corresponding bits in the IRQ_STATUS and IRQ_PENDING registers. The register is write-only.

Table 24: **Interrupt Acknowledge Register (IRQ_ACK)**

IRQ_ACK	
31	0

Table 25: **Interrupt Acknowledge Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31:0	IRQ_ACK	W	0	All bit position written with 1 will clear corresponding bits in both the IRQ_STATUS and the IRQ_PENDING registers

Interrupt Mode Register (IRQ_MODE)

This register is used to define which interrupts use fast interrupt mode. All bits written '1' will use fast interrupt mode. The register is write-only.

The register is only implemented when fast interrupt mode is enabled, by setting C_INTC_HAS_FAST to 1.

Table 26: **Interrupt Mode Register (IRQ_MODE)**

IRQ_MODE	
31	0

Table 27: **Interrupt Mode Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31:0	IRQ_MODE	W	0	All bit position written with 1 will use fast interrupt mode

Interrupt Address Vector Registers (IRQ_VECTOR_0 - IRQ_VECTOR_31)

These 32 registers are used as Interrupt Address Vector for the corresponding interrupt bit. The content is sent to the processor on the INTC_Interrupt_Address port when the interrupt occurs. The registers are write-only.

The two least significant bits and the most significant bits greater than or equal to C_INTC_ADDR_WIDTH (if any) of each register are fixed to 0.

For reserved interrupt bits (11-15), and unused external interrupts (greater than C_INTC_EXT_INTR+15), writing to the corresponding register has no effect.

The registers are only implemented when fast interrupt mode is enabled, by setting C_INTC_HAS_FAST to 1.

Table 28: **Interrupt Address Vector Register (IRQ_VECTOR_x)**

0		IRQ_VECTOR_x		0	
31	C_INTC_ADDR_WIDTH	C_INTC_ADDR_WIDTH-1	2	1	0

Table 29: **Interrupt Address Vector Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31:0	IRQ_VECTOR	W	0x10	The Interrupt Address Vector for the corresponding interrupt.

PITx Preload Register (PITx_PRELOAD) (x = 1, 2, 3 or 4)

The value written to this register determines the period between two consecutive PITx_Interrupt events. The period will be the value written to the register + 2 count events.

The register is implemented if C_USE_PITx is 1.

Table 30: **PITx Preload Register (PITx_PRELOAD)**

Reserved		PITx_PRELOAD	
31	C_PITx_SIZE	C_PITx_SIZE-1	0

Table 31: **PITx Preload Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31:C_PITx_SIZE	-	-	-	Reserved
[C_PITx_SIZE-1]:0	PITx_PRELOAD	W	0	Register holds the timer period

PITx Counter Register (PITx_COUNTER) (x = 1, 2, 3 or 4)

When reading this register the obtained data will be a sample of the current counter value.

The register is implemented if C_USE_PITx is 1 and C_PITx_READABLE is 1.

Table 32: **PITx Counter Register (PITx_COUNTER)**

Reserved		PITx_PRELOAD	
31	C_PITx_SIZE	C_PITx_SIZE-1	31

Table 33: **PITx Counter Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31:C_PITx_SIZE	-	-	-	Reserved
[C_PITx_SIZE-1]:0	PITx_COUNTER	R	0	PITx counter value at time of read

PITx Control Register (PITx_CONTROL) (x=1, 2, 3 or 4)

The EN bit in this register enables/disables counting. The PRELOAD bit determines if the counting is continuous with automatic reload of the PITx_PRELOAD value when lapsing (PITx_COUNTER = 0) or if the counting is stopped after counting the number of cycles defined in PITx_PRELOAD.

The register is implemented if C_USE_PITx is 1.

Table 34: **PITx Control Register (PITx_CONTROL)**

Reserved		RELOAD	EN
31	2	1	0

Table 35: **PITx Control Register Bit Definitions**

Bit(s)	Name	Core Access	Reset Value	Description
31:2	-	-	0	Reserved
1	PRELOAD	W	0	0 = Counter counts PITx_PRELOAD value cycles and the stops 1 = Counter value is automatically reloaded with the PITx_PRELOAD value when counter lapses
0	EN	W	0	0 = Counting Disabled 1 = Counter Enabled

Design Implementation

Target Technology

The target technology is an FPGA listed in the Supported Device Family(1) field of the LogiCORE Facts table.

Device Utilization and Performance Benchmarks

Because the MicroBlaze MCS is a module that is used together with other parts of the design in the FPGA, the utilization and timing numbers reported in this section are just estimates, and the actual utilization of FPGA resources and timing of the MicroBlaze MCS design will vary from the results reported here. All parameters not given in the table below have their default values.

Table 36: **Performance and Resource Utilization Benchmarks on Virtex-6 (xc6vlx240t-1-ff1156)**

Parameter Values (other parameters at default value)														Device Resources	
C_USE_UART_RX	C_USE_UART_TX	C_INTC_USE_EXT_INTR	C_INTC_INTR_SIZE	C_USE_FIT1	C_FIT1_No_CLOCKS	C_USE_PIT1	C_PIT1_SIZE	C_USE_GPI1	C_GPI1_SIZE	C_USE_GPO1	C_GPO1_SIZE	C_USE_IO_BUS	C_INTC_HAS_FAST	LUTs	Flip-Flops
1	1	0	0	0	0	0	0	0	0	0	0	0	0	40	75
1	1	1	5	0	0	0	0	0	0	0	0	0	0	69	110
1	1	1	5	0	0	0	0	0	0	0	0	0	1	118	173
1	1	1	5	1	65000	0	0	0	0	0	0	0	0	75	122
1	1	1	5	1	65000	1	32	0	0	0	0	0	0	121	216
1	1	1	5	1	65000	1	32	1	32	1	32	0	0	121	280
1	1	1	5	1	65000	1	32	1	32	1	32	1	0	119	361

LMB Timing

See the MicroBlaze Bus Interfaces chapter in the *MicroBlaze Processor Reference Guide* [Ref 1] for details on the transaction signaling.

Support

Xilinx provides technical support for this LogiCORE product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support of product if implemented in devices that are not defined in the documentation, if customized beyond that allowed in the product documentation, or if changes are made to any section of the design labeled *DO NOT MODIFY*.

Ordering Information

This Xilinx LogiCORE IP module is provided at no additional cost with the Xilinx ISE® Design Suite Embedded Edition software under the terms of the Xilinx End User License. The core is generated using the Xilinx ISE Embedded Edition software (EDK).

Information about this and other Xilinx LogiCORE IP modules is available at the Xilinx Intellectual Property page. For information on pricing and availability of other Xilinx LogiCORE modules and software, please contact your local Xilinx sales representative.

Reference Documents

The following reference documents are available online:

1. MicroBlaze Processor Reference Guide (UG081)
2. 7 Series FPGAs Configuration User Guide (UG470)

Revision History

The following table shows the revision history for this document:

Date	Version	Description of Revisions
04/24/12	1.0	Initial Xilinx release.

Notice of Disclaimer

Automotive Applications Disclaimer