SPI GPIO Expander

December 2010 **Reference Design RD1073**

Introduction

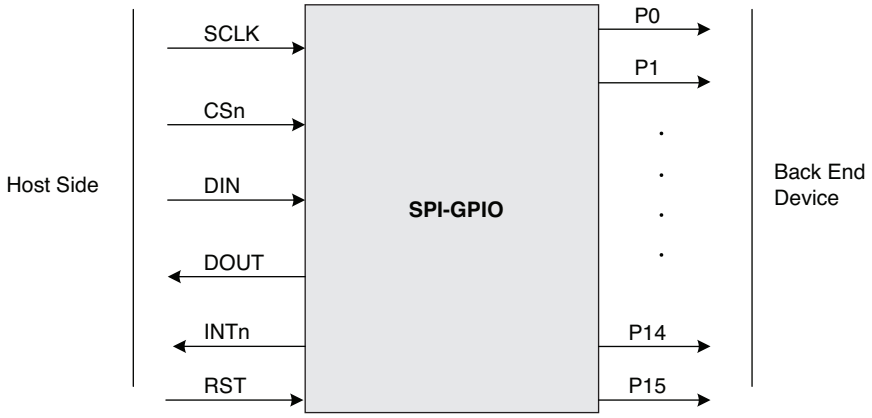
Microprocessors often have a limited number of general purpose I/O (GPIO) ports. This approach helps reduce pin count and shrink package size. I/O expanders, or port expanders, provide I/O expansion capabilities for microprocessors. They allow designers to save the GPIO ports on the microprocessor for other critical tasks. There are many generic I/O expander devices available. Most of them use low pin count protocols, such as I^2C or SPI, as the interface to the host. This design provides a programmable solution for serial expansion of GPIOs. It uses a Serial Peripheral Interface (SPI) as the interface between the microprocessor and the GPIOs. The design provides additional control and monitoring capabilities for the microprocessor when it does not have sufficient GPIOs to do the job.

Features

- SPI-compatible serial interface to the host
- 16 GPIOs can be configured as inputs or outputs
- GPIOs configured as inputs can cause an interrupt request to the host
- Interrupts can be masked if necessary
- All GPIOs are configured as inputs at hardware reset
- 16 GPIOs can be accessed individually at once; some combinations of four or eight GPIOs can be accessed as a group; all 16 GPIOs can be accessed as a group.

Interface

Figure 1. SPI to GPIO Interface



Functional Description

This design provides up to 16 ports, P0 to P15, controlled through the SPI-compatible serial interface. Each port is individually user-configurable as either a logic input or a logic output. A power-on reset or reset signal initializes the 16 ports as inputs. This design consists of a no-op register, a 16-bit configuration register, a 16-bit mask register, a 16-bit input register and a 16-bit output register. The 24-bit shift register controls the serial-in and serial-out of the data to and from the host. The interrupt generation block generates an interrupt signal to the host when one of the input ports changes its logic state. Figure 2 is the functional block diagram.

Figure 2. Functional Block Diagram

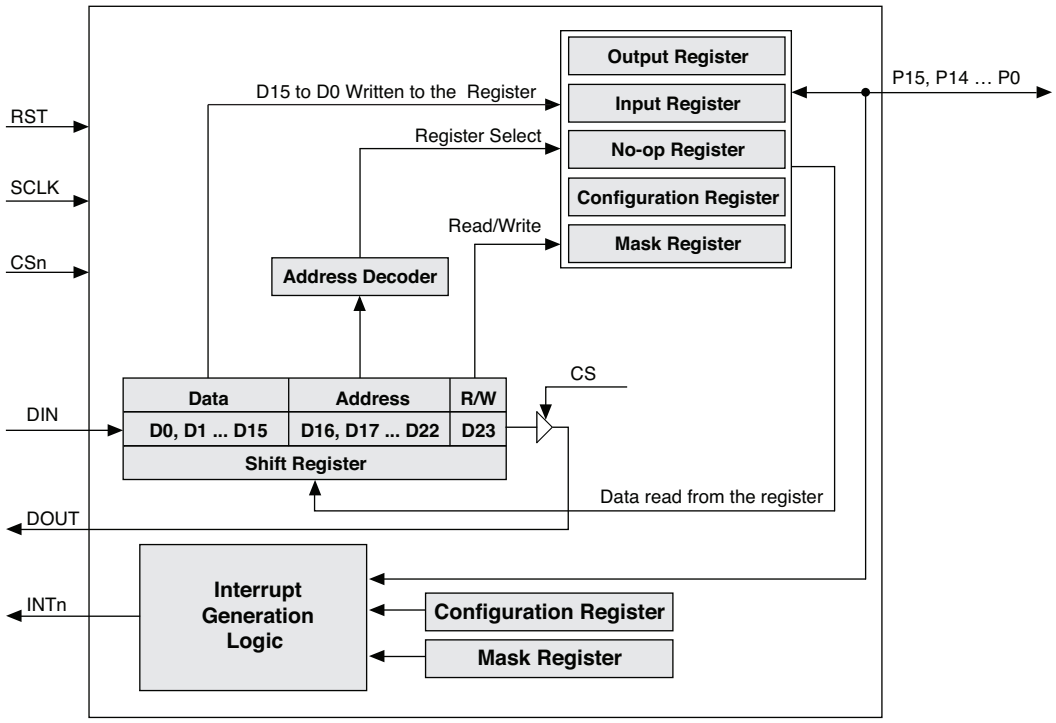


Table 1. SPI GPIO Expander I/O Interface Descriptions

Signal Name	Direction	Active State	Definition
Host Interface			
SCLK	Input	N/A	4-wire interface serial clock input port.
CSn	Input	Low	4-wire interface chip-select input, active low.
DIN	Input	N/A	4-wire interface serial data input port.
DOUT	Output	Low, High, HiZ	4-wire interface serial data output port.
RST	Input	High	CPLD rest signal.
INTn	Output	Low	Interrupt signal to the host, becomes active when one of the input GPIOs changes state.
Back End Interface			
P0 to P15	Output, Input	Low, High, HiZ	GPIO Ports 0 to 15.

Communication with the Host

This design communicates with the host through the SPI-compatible 4-wire serial interface. DIN is sampled on the rising edge of SCLK when the CSn pin is low. DOUT provides a copy of the data that was received 23.5 clocks earlier at the DIN pin or outputs internal register data upon request when CSn is low. When CSn is high, this design ignores all activity on SCLK and DIN. DOUT must be high impedance when CSn is high.

Data Format from the Host

Between taking the CSn pin low and taking it back high, the host should clock at least 24 bits into DIN. Only the last 24 bits will be retained. Bit D23 is the first in and bit D0 is the last in. Bit D23 is a command bit. Its logic 0 indicates a WRITE from the host to a register and logic 1 indicates a READ from a register by the host. Bits D22-D16 are defined as the address bits. The address bits indicate the address of the register that the host plans to access in this operation. Bits D15-D0 are the data bits the host writes to the register which is selected by the address bits.

Register Definitions

Registers in this design include a no-op register, a 16-bit configuration register, a 16-bit mask register, a 16-bit output register and a 16-bit input register.

The no-op register indicates no operation. This register can only be written by the host and its address is 0x00. Writing to this register does not change other register states or port states but will shift out the data in the D15 to D0 positions of the shift register.

The configuration register is used to configure the directions of the ports. This is a 16-bit register where each bit corresponds to a port. Bit 0 corresponds to P0, bit 1 corresponds to P1, etc. Setting the bit in the respective configure register enables the corresponding port as an input. In the same way, clearing the bit in the configuration register enables the corresponding port as an output.

Table 2. Configuration Register Definitions

Register	Address	Width	Access	Reset Value	Bit Definition
Configuration register	0x01	16	R/W	0xffff	'1' configures the port as an input; '0' configures the port as an output.

The mask register is used to mask the interrupt. This is a 16-bit wide register and each bit corresponds to a port. Bit 0 corresponds to P0, bit 1 corresponds to P1, etc. Setting the bit in the mask register will mask the interrupt generation of a specific port. Clearing the bit in the mask register enables the interrupt generation from that port.

Table 3. Mask Register Definitions

Register	Address	Width	Access	Reset Value	Bit Definition
Mask register	0x02	16	R/W	0xffff	'1' masks interrupt generation; '0' enables interrupt generation.

The output register is a write-only register. It sets the logic levels of the output ports defined by the configuration register. This is a 16-bit register and each bit corresponds to a port. Several addressing methods for output registers are available. As shown in Table 4, any port can be read individually. Some combinations of four or eight ports can be read simultaneously. All 16 ports can be read together.

Table 4. Output Register Address Map Definitions

Register	Access	Address	Reset Value
Output register port 0 (Data bit D0. Data bits D1-D15 are ignored).	W	0x03	0x1
Output register port 1 (Data bit D1. Data bits D0 and D1-D15 are ignored).	W	0x04	0x1
Output register port 2 (Data bit D2. Data bits D0-D1 and D3-D15 are ignored).	W	0x05	0x1
Output register port 3 (Data bit D3. Data bits D0-D2 and D4-D15 are ignored).	W	0x06	0x1
Output register port 4 (Data bit D4. Data bits D0-D3 and D5-D15 are ignored).	W	0x07	0x1
Output register port 5 (Data bit D5. Data bits D0-D4 and D6-D15 are ignored).	W	0x08	0x1
Output register port 6 (Data bit D6. Data bits D0-D5 and D7-D15 are ignored).	W	0x09	0x1
Output register port 7 (Data bit D7. Data bits D0-D6 and D8-D15 are ignored).	W	0x0A	0x1

Table 4. Output Register Address Map Definitions (Continued)

Register	Access	Address	Reset Value
Output register port 8 (Data bit D8. Data bits D0-D7 and D9-D15 are ignored).	W	0x0B	0x1
Output register port 9 (Data bit D9. Data bits D0-D8 and D10-D15 are ignored).	W	0x0C	0x1
Output register port 10 (Data bit D10. Data bits D0-D9 and D11-D15 are ignored).	W	0x0D	0x1
Output register port 11 (Data bit D11. Data bits D0-D10 and D12-D15 are ignored).	W	0x0E	0x1
Output register port 12 (Data bit D12. Data bits D0-D11 and D13-D15 are ignored).	W	0x0F	0x1
Output register port 13 (Data bit D13. Data bits D0-D12 and D14-D15 are ignored).	W	0x10	0x1
Output register port 14 (Data bit D14. Data bits D0-D13 and D15 are ignored).	W	0x11	0x1
Output register port 15 (Data bit D15. Data bits D0-D14 are ignored).	W	0x12	0x1
Output register ports 0-3 (Data bits D0-D3. Data bits D4-D15 are ignored).	W	0x13	0xf
Output register ports 4-7 (Data bits D4-D7. Data bits D0-D3 and D8-D15 are ignored).	W	0x14	0xf
Output register ports 8-11 (Data bits D8-D11. Data bits D0-D7 and D12-D15 are ignored).	W	0x15	0xf
Output register ports 12-15 (Data bits D12-D15. Data bits D0-D11 are ignored).	W	0x16	0xf
Output register ports 0-7 (Data bits D0-D7. Data bits D8-D15 are ignored).	W	0x17	0xff
Output register ports 8-15 (Data bits D8-D15. Data bits D0-D7 are ignored).	W	0x18	0xff
Output register ports 0-15.	W	0x19	0xffff

The input register is a read-only register. It reflects the incoming logic levels of the ports, regardless of whether the port is defined as an input or an output. This is a 16-bit register and each bit corresponds to a port. As with the output register, several addressing methods are available for input registers. Any port can be read individually. Some combinations of four or eight ports can be read simultaneously. All 16 ports can be read together. Table 5 shows the input register addressing methods.

Table 5. Definition of Input Register Addressing Map

Register	Access	Address	Reset Value
Input register port 0 (Data bit D0. Data bits D1-D15 are not changed).	R	0x03	0x0
Input register port 1 (Data bit D1. Data bit D0 and data bits D1-D15 are not changed).	R	0x04	0x0
Input register port 2 (Data bit D2. Data bits D0-D1 and D3-D15 are not changed).	R	0x05	0x0
Input register port 3 (Data bit D3. Data bits D0-D2 and D4-D15 are not changed).	R	0x06	0x0
Input register port 4 (Data bit D4. Data bits D0-D3 and D5-D15 are not changed).	R	0x07	0x0
Input register port 5 (Data bit D5. Data bits D0-D4 and D6-D15 are not changed).	R	0x08	0x0

Table 5. Definition of Input Register Addressing Map (Continued)

Register	Access	Address	Reset Value
Input register port 6 (Data bit D6. Data bits D0-D5 and D7-D15 are not changed).	R	0x09	0x0
Input register port 7 (Data bit D7. Data bits D0-D6 and D8-D15 are not changed).	R	0x0A	0x0
Input register port 8 (Data bit D8. Data bits D0-D7 and D9-D15 are not changed).	R	0x0B	0x0
Input register port 9 (Data bit D9. Data bits D0-D8 and D10-D15 are not changed).	R	0x0C	0x0
Input register port 10 (Data bit D10. Data bits D0-D9 and D11-D15 are not changed).	R	0x0D	0x0
Input register port 11 (Data bit D11. Data bits D0-D10 and D12-D15 are not changed).	R	0x0E	0x0
Input register port 12 (Data bit D12. Data bits D0-D11 and D13-D15 are not changed).	R	0x0F	0x0
Input register port 13 (Data bit D13. Data bits D0-D12 and D14-D15 are not changed).	R	0x10	0x0
Input register port 14 (Data bit D14. Data bits D0-D13 and D15 are not changed).	R	0x11	0x0
Input register port 15 (Data bit D15. Data bits D0-D14 are not changed).	R	0x12	0x0
Input register ports 0-3 (Data bits D0-D3. Data bits D4-D15 are not changed).	R	0x13	0x0
Input register ports 4-7 (Data bits D4-D7. Data bits D0-D3 and D8-D15 are not changed).	R	0x14	0x0
Input register ports 8-11 (Data bits D8-D11. Data bits D0-D7 and D12-D15 are not changed).	R	0x15	0x0
Input register ports 12-15 (Data bits D12-D15. Data bits D0-D11 are not changed).	R	0x16	0x0
Input register ports 0-7 (Data bits D0-D7. Data bits D8-D15 are not changed).	R	0x17	0x00
Input register ports 8-15 (Data bits D8-D15. Data bits D0-D7 are not changed).	R	0x18	0x00
Input register ports 0-15	R	0x19	0x0000

Host Writing to Registers

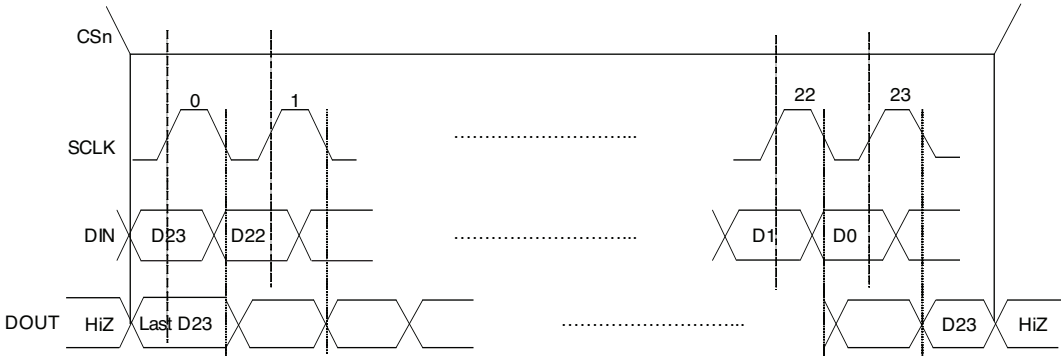
The no-op register, configuration register, mask register and output register can be written by the host. When CSn is low, it shifts in the DIN data on the rising edge of SCLK. When CSn goes high, the 24 bits in the shift register are then decoded and executed.

Use the following sequence for the host to write to a register:

1. Take SCLK low.
2. Take CSn low.
3. Clock 24 bits of data into the DIN with D23 first and D0 last (bit D23 is low, indicating the host is writing to the register; bits D22-D16 indicate the register address; bits D15 to D0 are the data written to the register).
4. Take SCLK low.
5. Take CSn high.

Figure 3 shows the relative timing of a WRITE from the host.

Figure 3. Writing a Register



Host Reading from Registers

The configuration register, mask register and input register can be read by the host.

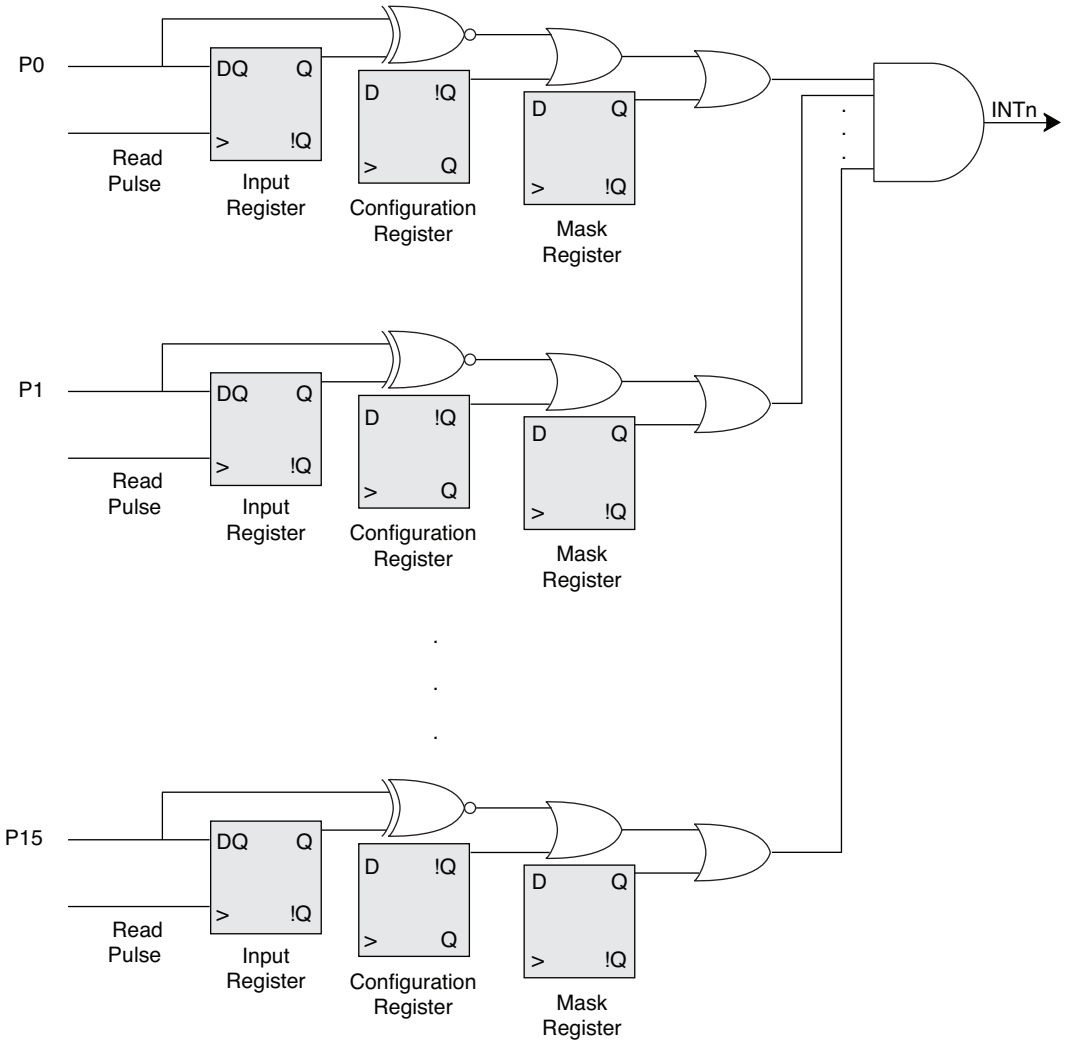
Use the following sequence for the host to read from a register:

1. Take SCLK low.
2. Take CSn low.
3. Clock 24 bits of data into DIN with D23 first, D0 last (bit D23 is high, indicating the host is reading the register; bits D22-D16 indicate the register address to be read; the values of bits D15 to D0 are ignored at this stage and can be any value).
4. Take SCLK low.
5. Take CSn high. When CSn goes high, positions D15-D0 in the shift register are loaded with the register data addressed by bits D22-D16.
6. Issue a write no-op register command, and examine the bitstream at DOUT. Data bits D15-D0 are the contents of the register to be read.

Interrupt Generation

The interrupt output signal is activated when one of the input ports changes states and when the corresponding bit in the mask register is not set (logic 0). A port defined as an output port cannot generate an interrupt. This design cannot identify the port that causes the interrupt if more than one port is generating an interrupt at the same time. The interrupt deactivates when the input returns to its previous state or the input register is read by the host. Figure 4 shows the interrupt generation logic. In Figure 4, the read pulse is generated when the host reads the input register.

Figure 4. Interrupt Generation Logic



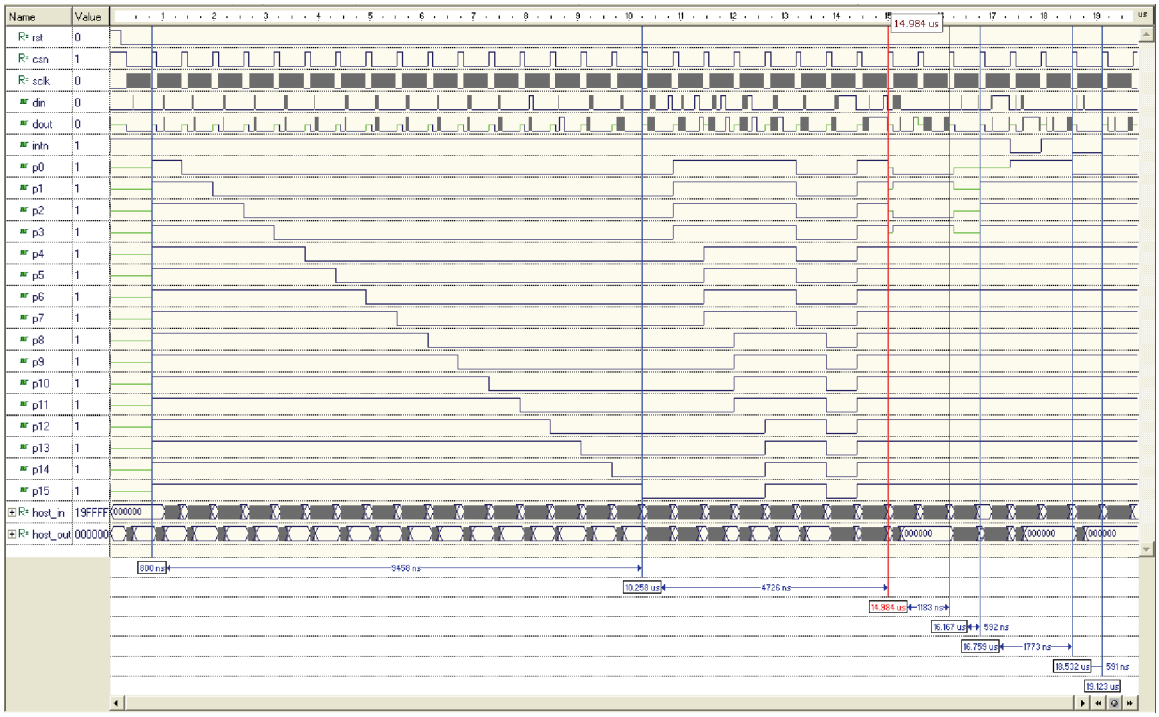
Test Bench Description

After reset, the 16 GPIOs are configured as input, the initial value of the output register is 0xff and the initial value of the mask register is 0xff. The following simulation scenario represents the most common READ/WRITE operations. All the events correspond to the timings shown in Figure 5.

1. The host writes 0x0000 to the configuration register to make all the GPIOs configured as output. Since the initial value of the output register is 0xff, the value of all GPIOs is at logic 1 as shown at 800 ns.
2. The host writes 0x0 to address 0x3 through 0x12 in turn to change ports P0-P15 output logic level from 1 to 0 (as shown between 1391ns to 10.258 µs).
3. The host writes 0xf to address 0x13 to change ports P0-P3 output logic to 1 at the same time as shown at 10.849 µs. Then it does similar WRITE operations to the other 4-port groups using the appropriate addresses. As a result, the output logic level changes from 0 to 1, four ports at a time.
4. The host writes 0x00 to address 0x17 to change the port P0-P7 output logic levels back to 0 as shown at 13.213 µs. Similar WRITE operations are performed to the other 8-port groups as shown in the simulation.

5. The host writes 0xffff to address 0x19 to make the port P0-P15 output logic levels back to 1 as shown at 14.395 µs.
6. The host then writes 0x000f to the configuration register to make ports P0-P3 as input and other ports as output (as shown at 14.984 µs).
7. The host reads address 0x13 to check if the read function is correct (as shown at 16.167 µs).
8. The host writes 0x0001 to the configuration register to make port P0 an input and other ports outputs (as shown at 16.759 µs).
9. The test bench changes the value of port P0 to activate the interrupt signal and then deactivates the interrupt signal by the host reading back the input register (as shown at 18.532 µs and 19.123 µs).

Figure 5. Simulation Results



Implementation

Table 6. Performance and Resource Utilization1

Device Family	Language	Speed Grade	Utilization	f_{MAX} (MHz)	I/Os	Architecture Resources
MachXO™ [1]	Verilog	-3	194 LUTs	>40	22	N/A
ispMACH® 4000ZE[2]	Verilog	-5 (ns)	126 Macrocells	>40	22	N/A
Platform Manager[3]	Verilog	-3	194 LUTs	>40	22	N/A

1. Performance and utilization characteristics are generated using LCMXO256C-3T100C with Lattice Diamond™ 1.1 and ispLEVER® 8.1 SP1 software. When using this design in a different device, density, speed or grade, performance and utilization may vary.
2. Performance and utilization characteristics are generated using LC4256ZE-5TN100C with Lattice ispLEVER Classic 1.4 software. When using this design in a different device, density, speed or grade, performance and utilization may vary.
3. Performance and utilization characteristics are generated using LPTM10-12107-3FTG208CES with Lattice ispLEVER 8.1 SP1 software. When using this design in a different device, density, speed or grade, performance and utilization may vary.

Technical Support Assistance

Hotline: 1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
February 2010	01.0	Initial release.
December 2010	01.1	Added support for Platform Manager device family.