Lab 4 - Interrupt-Driven Input Port

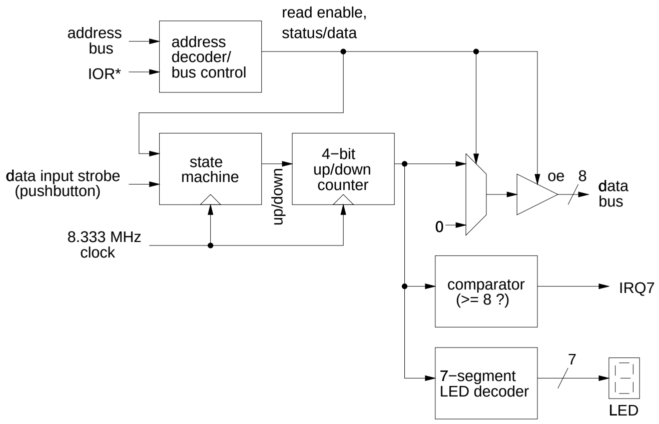
Introduction

In this lab you will design an interrupt-driven input port in VHDL and implement it on the FPGA in the lab.

You will also write a program in 8086 assembly language that initializes an interrupt vector and provides an interrupt service routine for the port.

Hardware Description

The following diagram shows a possible implementation of the input port:



Note that the port is not connected to a real data source – this is to simplify the design. The port has an active-low data input strobe that indicates that a new value is to be read by the port. When the strobe is asseted the ficticious input value is stored in a simulated 15-entry FIFO (first-in first-out) buffer. The port uses a 4-bit counter to keep track of the number of values in the buffer although it does not implement the buffer itself. The port generates an interrupt by asserting the PC-104 bus interrupt request number 7 (IRQ 7) when the number of stored values reaches 8.

The port has a read-only status register at I/O address 220H and a read-only data register at address 221H. The value read from the register at 220H should be the number of values currently in the FIFO. The value read from the register at 221H should be the oldest value in the FIFO. Since we are not actually going to implement the FIFO, reads from the data register should return 0. However, your port should *automatically* decrement the count of the number of stored values whenever the data register is read.

To help verify correct operation of the port, the current count should be displayed on an LED in hex (0-F).

Your design *must* be synchronous and use only the PC-104 bus clock, SYSCLK. Note that there will be *several* rising edges of the SYSCLK signal during each data input strobe and during each read cycle.

The table below shows the new connections between the PC-104 signals and the FPGA (note the correction for IOR*). The rows are numbered starting at 1 for the top (unmarked) row.

	Left Board		Right Board	
Row	PC-104 Signal	FPGA Pin	PC-104 Signal	FPGA Pin
6			IOR*	158
7			SYSCLK	211
8			IRQ7	109

Address Decoder

The address decoder asserts the data bus tri-state buffer output enable for I/O reads at address 220H or 221H. It sets the value of the status/data signal depending on the register that is being accessed.

Counter

The counter keeps track of the number of values in the fictitious FIFO buffer. Its control input makes the counter count up, down, or hold its value on each rising edge of the clock.

State Machine

The state machine controls the counter. The counter should be incremented only on the *first* clock edge after the data input strobe is asserted and only on the *first* clock edge of a data register read cycle. Your design should correctly handle the case where the data input strobe is asserted at the same time that the data register is read. The state machine will need to "remember" the previous values of the data input strobe and the data register read enable signals to detect the first clock edge.

Pre-Lab Assignment

Before the lab you must write and assemble the program described below. You must also design the circuit and test it by simulating its operation. The TA will ask to see your assembler and VHDL code and the simulation waveforms at the start of the lab.

Assembly Language Program

Write an 8088 assembly-language program that does the following:

- initializes the interrupt vector for IRQ 7 to point to an interrupt service routine included within your program
- enables interrupt number 7 in the PICs interrupt mask register
- executes an infinite loop (a JMP instruction to itself)

The ISR, included as part of your program, should do the following:

- save any registers it will modify
- read the status register to check if there are any values stored in the FIFO
- if yes, read the data register and repeat the previous step
- re-enable interrupts by issuing the EOI instruction to the PIC
- restore any saved registers
- return from the ISR

Refer to lecture 10 for details on the locations of interrupt vectors, 8086 interrupt service routines and the operation of the 8259 PIC.

VHDL Description

Write, compile and test by simulation a VHDL description for the circuit described above. Use the following tests to help verify its correct operation:

- Apply three clock pulses while the data input strobe (pushbutton) is asserted. The LED output should change from 0 to 1 on the first clock edge and and remain at the encoded value for 1.
- Read from the status register. The data bus should read 01H.
- Apply 16 more clock cycles with the pushbutton input alternately asserted and not asserted. The LED output should read the encoded value for 9.
- Apply the address of the data register to the address bus and assert IOR*. Apply three clock pulses. The LED output should count down to, and remain at, the encoded value for 8. The IRQ7 output should be asserted (high).
- Read from the data port again. The LED output should read 7 and the IRQ7 output should be low.

Print and Copy Files

Save the files *projectname*.`asm` (assembly language source code), *projectname*.`com` (DOS executable), *projectname*.`acf` (device and pin assignments), *projectname*.`vhd` (VHDL code), and *projectname*.`scf` (test waveforms) to a floppy disk to bring it with you to the lab. Print out the assembler and VHDL code and the simulator output waveforms.

Lab Procedure

Connect the PC-104 address and data bus signals, IOR*, SYSCLK, and IRQ7 to the FPGA pins on the interconnect board as described in the previous lab. Double-check your connections and turn on the power.

Compile your VHDL code if you haven't already done so, and configure the FPGA as described in the previous lab.

Press the button and observe the LED display. It should count up on each button press and wrap around back to zero when it reaches F.

Assemble and link your assembly code if you haven't already done so.

Run the Windows Hyperterm program. Reset the SBC and download your program.

Run your program on the SBC.

Press the button and observe the LED display. If your device and program are working properly, the LED will count up to 7 and then get reset to 0 on the next button push as the ISR reads the FIFO contents.

When your device is working properly ask the TA to check your work. He will make sure your device works as required and ask you one or two questions to verify your understanding of the material.

Report

Submit a short report with a written description of your circuit. Include a listing of your assembly-language program, the VHDL code and a printout of the simulation waveforms that demonstrate correct operation of your device.