ECE480/580 Digital Systems Design
Project 1: Microprocessor Design

1. INTRODUCTION

In this project, you will use the Quartus software package to design, simulate and test a simple microprocessor design. The requirements for this project consist of completing the QUARTUS design and printing the VHDL files, functional simulation waveforms and laboratory report. Additionally, each design will require appropriate information obtained from various compilation reports and netlist viewers. All simulations should be done using the ModelSim-Altera Simulator. You may use VHDL testbenches or ModelSim DO files to control the simulation.

2. DESIGN

Microprocessor Design: Design a microprocessor design so that it uses the following instruction format:

OPCODE (5 bits)	REG (1 bit)	ADDRESS (10 bits)

The opcode field is to support 16 instructions. The instructions to support should include the following:

Instruction	Function	Opcode
ADD R[REG], ADDRESS	R[REG] ← R[REG] + contents of ADDRESS	00000
STR R[REG], ADDRESS	ADDRESS ← R[REG]	00001
LDR R[REG], ADDRESS	R[REG] ← contents of ADDRESS	00010
JMP ADDRESS	PC <- ADDRESS	00011
JN R[REG], ADDRESS	IF R[REG]<0 THEN PC ← ADDRESS	00100
SUB R[REG], ADDRESS	R[REG] ← R[REG] - contents of ADDRESS	00101
INC R[REG]	R[REG] ← R[REG] + 1	00110
OR R[REG], ADDRESS	R[REG] ← R[REG] OR contents of ADDRESS	00111
AND R[REG], ADDRESS	R[REG] ← R[REG] AND contents of ADDRESS	01000
NOT R[REG]	R[REG] ← ¬R[REG]	01001
JP R[REG], ADDRESS	IF R[REG]>0 THEN PC ← ADDRESS	01010
JZ R[REG], ADDRESS	IF R[REG]=0 THEN PC ← ADDRESS	01011
SHL R[REG]	R[REG] ← R[REG] shifted left by one	01100
SHR R[REG]	R[REG] ← R[REG] shifted right by one	01101
IN R[REG]	R[REG][7..0] ← SW[7..0]	01110
OUT R[REG]	LEDR[7..0] ← R[REG][7..0]	01111
USER DEFINED	USER DEFINED	10000-11111

The memory for the microprocessor is to be 1024x16 holding both instructions and data. There are to be a minimum of two 16-bit general purpose registers: R[0] and R[1]. The register accessed by a particular instruction is determined by the REG field. Define any additional registers required in implementing your microprocessor.

In addition to the 16 predefined instructions, you are to define an additional 16 instructions of your choice. It is strongly suggested that at least two of the additional instructions provide support for indirect memory access (reads and writes) where the content of one of the general purpose registers is assumed to hold a memory address. Your additional instructions may redefine the use of the REG and ADDRESS fields as needed.

There are to be two clock sources for your microprocessor. If **SW[9]**=0 then the clock source for the microprocessor is to be a 12.5 MHz clock created from a clock divider component. If **SW[9]**=1, the clock source is to be the **KEY[0]** pushbutton. The **KEY[0]** pushbutton is basically used as a single step clock source. Multiplexing a single step and free running clock should be extremely valuable for debugging purposes. **KEY[1]** should be used as a reset for your microprocessor. Use the clock multiplexer example given on the class website for glitch-free multiplexing of the two clock sources. You should also debounce the **KEY[0]** pushbutton clock source using an appropriate VHDL debounce component.

Create a SignalTap II Logic Analyzer instance and verify your design as it operates. Display sufficient signals to verify your design. At a minimum, these should include the active clock, the program counter, registers R[0] and R[1], instruction register, data to/from memory, and any other important defined register or control signal. Use the 50 MHz clock source as the sampling clock for the logic analyzer.

Create an assembly language program that sorts a list of data values in memory using a bubble sort algorithm. The list should contain 100 data elements to be sorted. Use the In-System Memory Content Editor to verify the results of your sort routine. Create a second assembly language program that implements a rotating lights display to the LEDS **LEDR[7..0]**. The lights should rotate left if **SW[0]**=0 and right if **SW[0]**=1. The lights should change at an approximate rate of 10 Hz. You will need to create a software delay loop in your assembly language program to accomplish this. Your microprocessor must also be capable of executing any assembly language program I provide. Therefore, you should test the operation of your design thoroughly.

View the compilation report files (*.rpt) that are generated when you compile your design for the Cyclone V device. The **Fitter Report File** (*.fit.rpt) summarizes the logic utilization, memory block, and interconnect usage for your design. In your lab report, summarize logic utilization, memory, and interconnect usage for your design. The “Fitter Resource Usage Summary” section in the fitter report file should contain all the necessary information. This information can also be found in the **Compilation Report** using **`Processing->Compilation Report->Fitter->Summary`**.

Include RTL Viewer (**`Tools->NetList Viewers->RTL Viewer`**) and Technology Map Viewer (**`Tools-> NetList Viewers->Technology Map Viewer`**) schematics in your lab report.

A ModelSim simulation should clearly demonstrate all instructions the microprocessor is capable of performing. You must use either VHDL testbenches or ModelSim DO files for controlling your simulation. The testbenches or DO files must be included in your lab report.

Download your design to the Altera development board and test. Note that there will be two *.mif files associated with your design (one for each assembly language program) but only one of them will be used at any given time. You may compile your design, specifying one of the *.mif files. The *.mif file can be downloaded to your design at runtime using the In-System Memory Content Editor so you do not have to recompile your design if you wish to change executing programs.