Lab 6 – Arithmetic Unit

CS 2052 Computer Architecture

Dept. of Computer Science and Engineering, University of Moratuwa

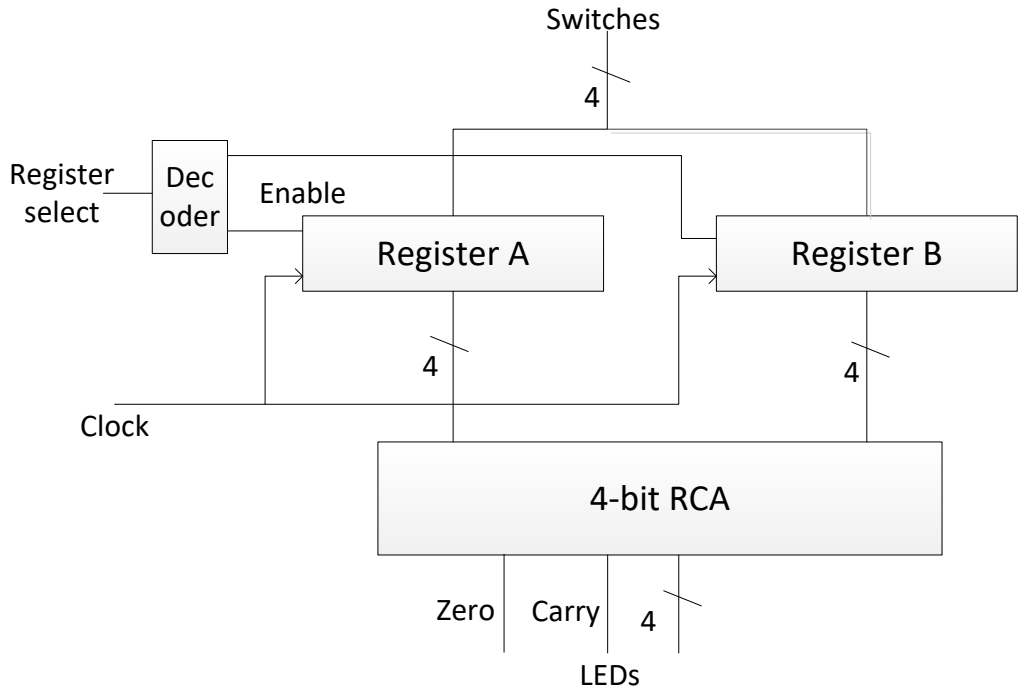
Learning Outcomes

In this lab, we will design a 4-bit arithmetic unit that can add 2 numbers stored in registers. After completing the lab, you will be able to:

- design and develop a 4-bit register
- design and develop a 4-bit arithmetic unit
- verify their functionality via simulation and on the development board

Introduction

Registers are used to store a set of bits inside a microprocessor. In this lab, we will design a 4-bit Arithmetic Unit (AU) that can add numbers stored in 2 different registers. High-level diagram of the AU is given below.



We will build a 4-bit register using D Flip Flops. Register also has an Enable input and a Clock input. 2 registers are then connected to our 4-bit Ripple Carry Adder (RCA) developed in Lab 3. This forms a simple AU.

Registers are loaded with a 4-bit binary value specified via 4 switches. Which register to load is determined by the input to the Enable pin of a register. The output of the RCA is connected to a set of LEDs. Clock input is used to synchronize the behaviour of the circuit.

Building the Circuits

Step 1: Building 4-bit Register Symbol.

Build a 4-bit register using D Flop Flops (look for a symbol named **FD**). Add **Enable** and **Clock** pins.

Create a new symbol and name it as **4_Reg**.

Step 2: Building 4-bit Arithmetic Unit.

Import RCA symbol related schematics from Lab 3.

You may either use a 1-2 decoder symbol or build it using logic gates.

Build the circuit given in the above diagram. Name the schematic file as **AU**.

Label the inputs to Registers as **D0-D3**, **RegSel**, and **Clock**. Label the outputs as **S0-S3** and **C_Out**.

Verify the functionality of the AU using the simulator. Use your index number for some of the input combinations.

Step 3: Next, we need to derive the zero flag. While the carry flag is directly available from the RCA, we need to add the necessary logic to derive the zero flag based on the output from RCA. Therefore, add relevant logic gates to detect a zero output from RCA. Label the zero output as Z**_Out**.

Step 4: Connecting inputs and outputs.

Connect switches **SW0**-**SW3** as the inputs **D0**-**D3**, **SW7** as the **RegSel** input, and **BTN0** as **Clock** input. Connect outputs **S0**-**S3** to LEDs **LD0**-**LD3**, **C_out** to **LD6, Z_out** to **LD7**.

Step 5: Test on BASYS2.

Generate the programming file (i.e., bitstream) and load it to the BASYS2 board.

Change the switches on the BASYS2 and verify the functionality of your circuit.

Demonstrate the circuit to the instructor and get the Lab Completion Log singed.

No lab report is due for this lab.

Prepared By

- Dilum Bandara, PhD – Mar 20, 2014.
- Updated on Oct 29, 2015.
- Updated on Nov 10, 2016.
- Updated on Oct 26, 2017.