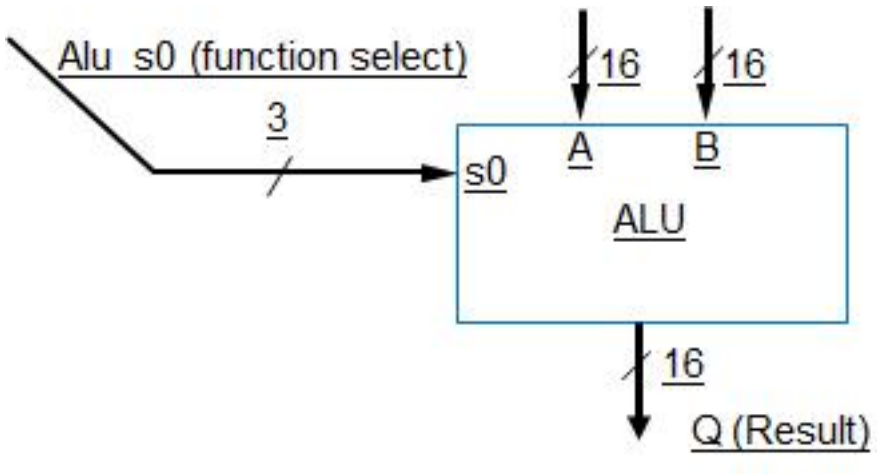
TCES 330
Programmable Processor Components

ALU

Part IV

The Arithmetic Logic Unit (ALU) is a critical component in our programmable processor design. In this exercise you will design, build, and test an 8-function ALU. The component is shown in the figure below.



Your module should contain the following code snippet and implement the eight functions listed.

```
// This ALU has eight functions:
//   if s == 0 the output is 0
//   if s == 1 the output is A + B
//   if s == 2 the output is A – B
//   if s == 3 the output is A (pass-through)
//   if s == 4 the output is A ^ B
//   if s == 5 the output is A | B
//   if s == 6 the output is A & B
//   if s == 7 the output is A + 1;
// the additional functions are for future expansion
module ALU( A, B, S, Q );
  input [2:0] Sel;     // function select
```

```
  input [15:0] A, B;   // input data
  output [15:0] Q; // ALU output (result)


// your code…
```

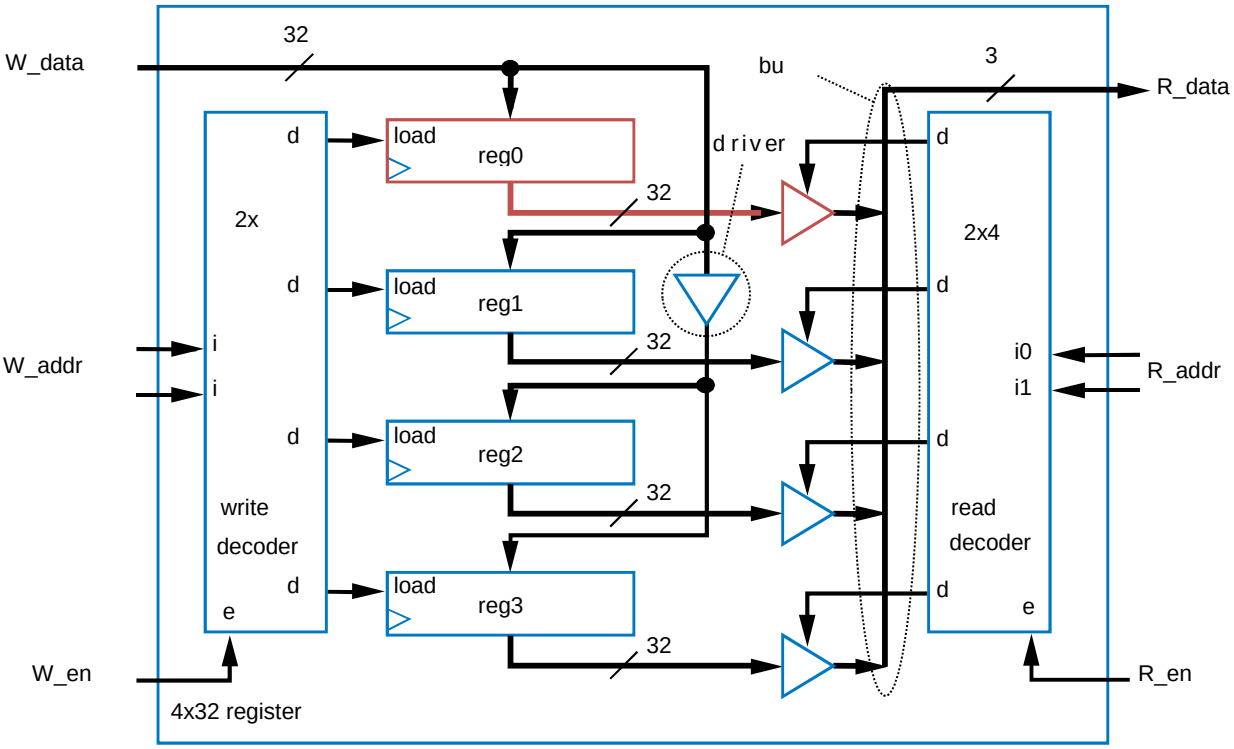
1. Create a Quartus II project for your ALU. The top-level module should instantiate your ALU and interface to the DE2 as follows. Note that each input to the ALU is 16 bits, so we can't exercise all of the input lines with switches:
    a. Use SW[2:0] for the select function, S.
    b. Use SW[6:3] for input A.
    c. Use SW[10:7] for input B.
    d. Display A on HEX0
    e. Display B on HEX1
    f. Display Q on HEX4, 5, 6 and 7.
    g. Use LEDR to display the state of the switches, SW.
2. Test your circuit on the DE2 board as much as you can to verify its correctness.
3. Write a ModelSim testbench that tests your ALU module more thoroughly.

Register File

Part V

You are to write Verilog code to implement a 16 X 16 **dual-output** register file as follows. Base your design on the figure below. The differences being:

1. Yours will have 16 16-bit registers (instead of four 32-bit registers).
2. Yours will have two independent outputs. This means you will have two R_datas, two R_addrs, and two R_ens.

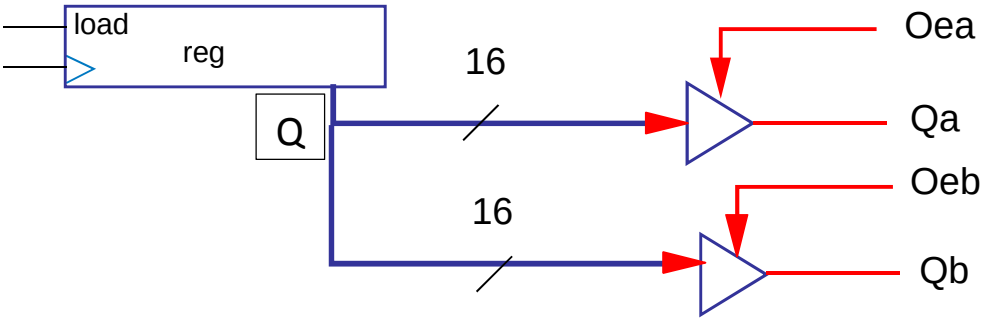


You will need to modify the code shown in class (posted on Moodle as RegisterFileBundle). The first few lines of your code should look like this:

```
// TCES 330, Spring 2014
// This is a 16 X 16 dual output register file
// MORE COMMENTS

module RegisterFile( W_data, W_addr, W_en, Ra_addr, Rb_addr, Ra_en,
                     Rb_en,  Clk, Reset, Ra_data, Rb_data );
  input [3:0] Ra_addr, Rb_addr, W_addr; // Comments to explain this
  input [15:0] W_data;
  input Ra_en, Rb_en, W_en;  // comments
  input Clk, Rst;
  output [15:0] Ra_data, Rb_data
//…
```

The dual output register with enable looks like this:



To implement the dual output feature you will need to modify the registers to have dual outputs. This means you will modify the registers shown in above to have two outputs. The first few lines of your register should look like this:

```
// TCES 330, Spring 2014
// Dual output register with Enable
// more comments…

module RegisterDualOE( I, Ld, Oea, Oeb, Clk, Rst, Qa, Qb );
  input [15:0] I;  // input data
  input Oea;       // channel a output enable
  input Oeb;       // channel b output enable
  input Ld;        // register load enable
  input Clk;       // input clock
  input Rst;       // synchronous reset signal
  output [15:0]Qa; // channel a output
  output [15:0]Qb; // channel b output
```

Testing

1. Your top level module will instantiate your register file and hook it to the switches and displays as follows. Note that we will not fully test your circuit in that we only store 6 bits of data in the 16-bit registers.
    a. Use switches 0-5 for the write data. Show these data on the corresponding red LEDs.
    b. Use switches 6-9 for the write address. Show these data on the corresponding red LEDs.
    c. Use switches 10-13 for the A-data read address. Show these data on the corresponding red LEDs.
    d. Use switches 14-17 for the B-data read address. Show these data on the corresponding red LEDs.
    e. Use HEX displays 0-3 to display the A-data.
    f. Use HEX displays 4-7 to display the B-data.
    g. Use Key 0 for the clock.
    h. Use Key 1 for a reset signal (hold Key 1 down while you produce a clock pulse). Don't forget that the output of the keys on the DE2 go LOW when you press them.
    i. You can Enable both the Write and Read sides with 1'b1s (enable all the time).
2. Use the provided testbench, testRegisterFile, in ModelSim to test your Register File more thoroughly.