# Computer Architecture Lab #1

You will design a part of **16-bit** ALU that accepts 2 16-bit input values “A” and “B” and provides 16-bit output “F”. The ALU has 4-bit selection inputs “S” (S0->3) and “Cin” input. The required part of the ALU provides a total of 12 operations specified in the following table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **S3** | **S2** | **S1** | **S0** |  |
| Part B | 0 | 1 | 0 | 0 | F = A and B |
| 0 | 1 | 0 | 1 | F = A or B |
| 0 | 1 | 1 | 0 | F = A xor B |
| 0 | 1 | 1 | 1 | F = Not A |
| Part C | 1 | 0 | 0 | 0 | F=Logic shift right A |
| 1 | 0 | 0 | 1 | F=Rotate right A |
| 1 | 0 | 1 | 0 | F=Rotate right A with Carry |
| 1 | 0 | 1 | 1 | F=Arithmetic shift right A |
| Part D | 1 | 1 | 0 | 0 | F=Logic shift left A |
| 1 | 1 | 0 | 1 | F=Rotate left A |
| 1 | 1 | 1 | 0 | F=Rotate left A with Carry |
| 1 | 1 | 1 | 1 | F = 0000 |

**Requirement:**

You will design a part of **16-bit** ALU as follows:

1. Write VHDL code for parts B, C and D in 3 separate VHDL files **Each file is named** as “partB”, “partC”, “partD” respectively.
   1. Part B ports
      1. Input => A, B, S0, S1
      2. Output => F
   2. Part C ports
      1. Inputs => A, Cin, S0, S1
      2. Output => F, Cout (the shifted/rotated bit)
   3. Part D ports
      1. Inputs => A, Cin, S0, S1
      2. Output => F, Cout (the shifted/rotated bit)
2. Compile your Code, the code should be free of errors and warnings
3. Simulate each file using the inputs stated below.
4. Integrate all 3 files in a bigger file, named “ALU”, in structural way and use 4-bit “S” selection input to choose between components.
5. Your simulation will be delivered using **DO files** only.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Operation** | **A** | **B** | **F** | **Operation** | **A** | **Cin** | **F** |
| AND | 0F0F | 000A | 000A | F=Rotate right A with Carry | 0F0F | 0 | 0787 |
| OR | 0F0F | 000A | 0F0F | F=Rotate right A with Carry | 0F0F | 1 | 8787 |
| XOR | 0F0F | 000A | 0F05 | F=Logic shift left A | 0F0F | \_ | 1E1E |
| NOT | 0F0F | \_ | F0F0 | F=Rotate left A | F0F0 | \_ | E1E1 |
| F=Logic shift right A | 0F0F | \_ | 0787 | F=Rotate left A with Carry | F0F0 | 0 | E1E0 |
| F=Rotate right A | 0F0F | \_ | 8787 | F=Rotate left A with Carry | F0F0 | 1 | E1E1 |
| F=0000 | \_ | \_ | 0000 | F=Arithmatic shift right A | F0F0 | \_ | F878 |

**N.B. you will be graded for code neatness and understanding**, Good luck

**Self-assessment**



**1.** How to describe wires in VHDL?

**2.** What hardware does the when else statement maps to?

**3.** Is VHDL case sensitive?



**Questions for Next Lab:**

**1**. If you need to use several sizes of the same component (i.e. 2-bit adder, 4- adder), is there a way to use the same entity definition for all sizes without creating single entity for each size?

**2.** What is a process?

**3.** Draw a schematic for full adder.