Week2-ripple carry adder..

The **multiplexer**, shortened to “**MUX**” or “MPX”, is a **combinational** logic **circuit** designed to switch one of several input lines through to a single common output line by the application of a control signal.

**Combinational Circuit** is the type of **circuit** in which output is independent of time and only relies on the input present at that particular instant.

On other hand **Sequential circuit** is the type of **circuit** where output not only relies on the current input but also depends on the previous output.

Mux,demux combinational

Register,counter-sequential

Ripple carry adder

4 bit

4 full adders,wires to prop carry.

Both **latches** and **flip**-**flops** are circuit elements whose output depends not only on the current inputs, but also on previous inputs and outputs. The **difference between** a **latch** and a **flip**-**flop** is that a **latch** does not have a clock signal, whereas a **flip**-**flop** always does.

Flip flop edge triggered.

The ALU (Arithmetic and Logic Unit) is the part of cpu

performs the arithmetic and logic operations which are the key computations.

Your task in this assignment is to design and simulate a 16-bit1 ALU which performs the following core operations

* Arithmetic Addition and subtraction (two’s complement)
* Logic AND along with OR operation

Register file

2 values read, one written every clock cycle.

How to construct ?

* To achieve high performance, microprocessors incorporate a register file.
* **The register file, is a bank of registers used as a temporary high-speed storage into which data is fetched from memory.**
* The register file then provides inputs to the ALU.
* As you may recall, in assignment 2 you constructed an ALU that received two 16-bit inputs and produced one 16-bit output.

**Program Counter**

The Program

Counter and its associated circuitry form a crucial part of a microprocessor.

For a microprocessor to load and execute an instruction, it first needs to fetch the instruction from the memory.

To do so, there has to be a **register in the microprocessor which stores the memory address of the next instruction to be fetched**. That register is the Program Counter (PC).

operations

* Likewise, the circuitry associated with the PC registers needs to support two operations:

(1) increment the PC contents by one,

(2) add/subtract given value to PC contents

* The PC register is of length 16-bits.
* After the reset signal is applied, its contents should be zero.
* Considering the PC module as a black box, it has the following inputs: clk, reset, inc, add, sub, offset of which only offset is a wire vector of length 16.
* The only output of pc is a 16-bit wire vector of PC register contents.

The increment and add/subtract operations each take one clock cycle, so one or add/subtract operation can be performed every clock.

Full adder code

**module fulladd(input wire a, b, cin, output wire sum, cout);**

**wire [4:0] t;**

**xor2 x0(a,b,t[0]);**

**xor2 x1(t[0],cin,sum);**

**and2 a0(a,b,t[1]);**

**and2 a1(a,cin,t[2]);**

**and2 a2(b,cin,t[3]);**

**or2 o0(t[1],t[2],t[4]);**

**or2 o1(-----------t);**

**endmodule**

**sum=a exor b exor c**

carry=ab+bc +ac

what is module?

* *Icarus Verilog*is a Verilog simulation and synthesis tool.
* GTKWave is an open-source waveform viewer that displays VCD (and other) files graphically .
* , which is used to view the simulated output of the Verilog code.
* Module are the building blocks of Verilog designs
* Describes the functionality of the design.

States the input and output ports

* module followed by circuit name and port list
* Each port is either an input or output

Always terminates with endmodule

module andgate (a, b, y);

input a, b;

output y;

assign y = a & b;

endmodule

**wire** elements are used to connect input and output ports of a module instantiation together with some other element in your design

module mux2(input wire i0,i1,j,output wire o)

assign o=(j==0)?i0:i1;

endmodule