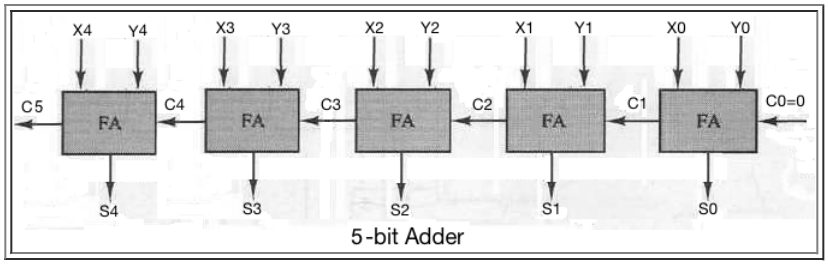
CSc 137 Verilog Programming #3: 5-Bit Adder

1. In this lab we write a Verilog program to simulate a 5-bit adder. Submit only 1 program file "Adder.v" to "3rdPrgAssig" folder located under your named folder in the dropbox host Voyager. Again, make "V2" folder (for version 2) under your named folder if you should need to re-submit (use "V3" for if needed further, etc.) Files misplaced at other locations may get points deducted.

2. A single-bit adder that takes in a carry bit is called a *Full Adder* (FA, for which module we programmed in the previous assignment). To compose a 5-bit adder, 5 FA's are linked together as shown below. The first carry-in bit **C0** is set to zero.



3. **Keyboard Input Examples**: How to write Verilog program statements to get keyboard input. Use the keyboard to enter two two-digit decimal numbers, one at a time. Each number entered is limited since there are only 5 bits to accommodate it.

4. Copy and run the demo executable:

```
atoz% cp -p ~changw/html/137/prg/3Adder/demo-a.out  .
atoz% demo-a.out
Enter X:
02
Enter Y:
05
X = 2 (00010)  Y = 5 (00101)
Result= 7 (00111)  C5 = 0

atoz% demo-a.out
Enter X:
17
Enter Y:
19
X = 17 (10001)  Y = 19 (10011)
Result= 4 (00100)  C5 = 1
```

5. The ASCII value obtained from a keyboard input must be converted, e.g., the character *'3'* will be read by the program as 51 (its ASCII table order) so subtract 48 from it to get the value it represents.

6. The skeleton of your program (5-bit adder) may look like:

```
// PUT YOUR NAME HERE
// Adder.v, 137 Verilog Programming Assignment #3
module TestMod;                      // the "main" thing
   parameter STDIN = 32'h8000_0000; // I/O address of keyboard input channel

   reg [7:0] str [1:3]; // typing in 2 chars at a time (decimal # and Enter key)
   reg [4:0] X, Y;      // 5-bit X, Y to sum
   wire [4:0] S;        // 5-bit Sum to see as result
   wire C5;             // like to know this as well from result of Adder

   instantiate the big adder module (giving X and Y as input, getting S and C5 as output)

   initial begin
```

```
         prompt for entering X: $display("Enter X: ");
         get 1st character:        --> str[1] = $fgetc(STDIN);
         get 2nd character:        --> str[2] = $fgetc(STDIN);
         and the ENTER key:        --> ...
         convert str to value for X:
            str[1] - 48 first, then times 10, then + str[2] - 48

         do the above to get input and convert it to Y

         #1; // wait until Adder gets them processed
         $display X and Y (run demo to see display format)
         and
         $display S and C5 (run demo to see display format)
      end
   endmodule

   module BigAdder(X, Y, S, C5);
      input [4:0] X, Y;   // two 5-bit input items
      output ...          // S should be similar
      output ...          // another output for a different size

      ...                 // declare temporary wires

      ... (get an instance of a full adder, C0 is 0)
      ... (get another full adder...)
      ... (get another full adder...)
      ... (get another full adder...)
      ... (get another full adder...)
   endmodule

   module FullAdderMod(...); // single-bit adder module
      ... code the full adder (like in previous assignment) ...
   endmodule

   module MajorityMod(...); // carry-bit generator module
      ... code the full adder (like in previous assignment) ...
   endmodule

   module ParityMod(...); // sum-bit generator module
      ... code the full adder (like in previous assignment) ...
   endmodule
```

- A List of Useful Linux and vi Commands
- Access Dropbox from a Linux Shell