Answers to ENGN1630 Second Optional Problem Set – Fall 2014

1.)
```
module p1h2ay11 (input [10:00] A, input clk, output reg SIGOUT);

wire [10:0] sum;
wire [11:0] cy;

reg [10:0] accum;
genvar i;

generate
for (i = 0; i < 11; i = i + 1) begin: AddTree
      full_add(A[i], accum[i], cy[i], sum[i], cy[i + 1]);
      end
endgenerate

assign cy[0] = 1'b0;

always @ (posedge clk)
begin
accum <= sum[10:0];
SIGOUT <= SIGOUT^cy[11];
end

endmodule

module full_add (input wire a, b, cin, output wire s, cout);

assign s = a^b^cin;
assign cout = (a && b) || (a && cin) || (b && cin);

endmodule
```

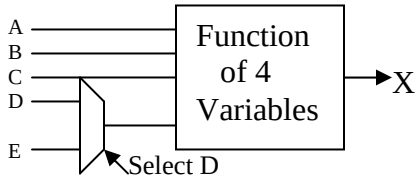
This implementation uses structural components to avoid having to write both the sum and carry expressions repeatedly. An alternate to the generate loop would be to write structural lines of the form:

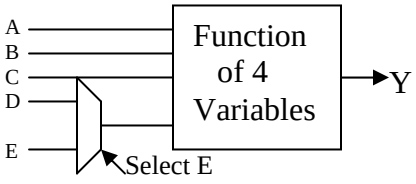
```
   full_add U1 (A[0], accum[0], 1'b0, sum[0], cy[1]);
```

2.) The first constraint is that each CLB can have only 5 inputs and 2 outputs. One way to get this is:

CLB #	GATE #
1	1,6
2	5,4
3	2,3

The division is forced because each gate of the original design has 3 inputs. Only pairs of gates sharing an input can fit within the limit of 5 inputs. For gate # 3, first consult fig. 6 of the Xilinx data. The combinational form for CLB #3 has to be "FG Mode" because there are to be two combinational outputs. The block is purely combinational, so the outputs are set as X = F and Y = G. These set up a configuration with the inputs split as:

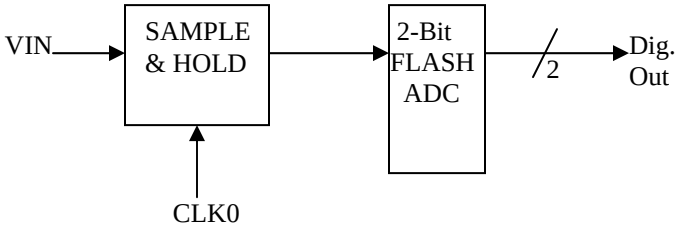




Suppose one connects A as the $\overline{R}$ of the flip-flop, B as the output of gate 4, C as unused, D as the D-input of the flip-flop, E from the Q output of the flip-flop, X as the gate3 output, and Y as the $\overline{Q}$ output of the flip-flop. While there are other possibilities from rearranging the inputs, this arrangement requires the CLB to be programmed for $F = \overline{A \cdot B \cdot D}$ and $G = \overline{A \cdot B \cdot E}$.

3.1) $D_0 = B_0 \cdot \overline{B_1} + B_2$ and $D_1 = B_1$.

3.2) Contents of the last subranging block, BLOCK0:



There is no DAC or analog amplifier needed because this is the end of the chain.

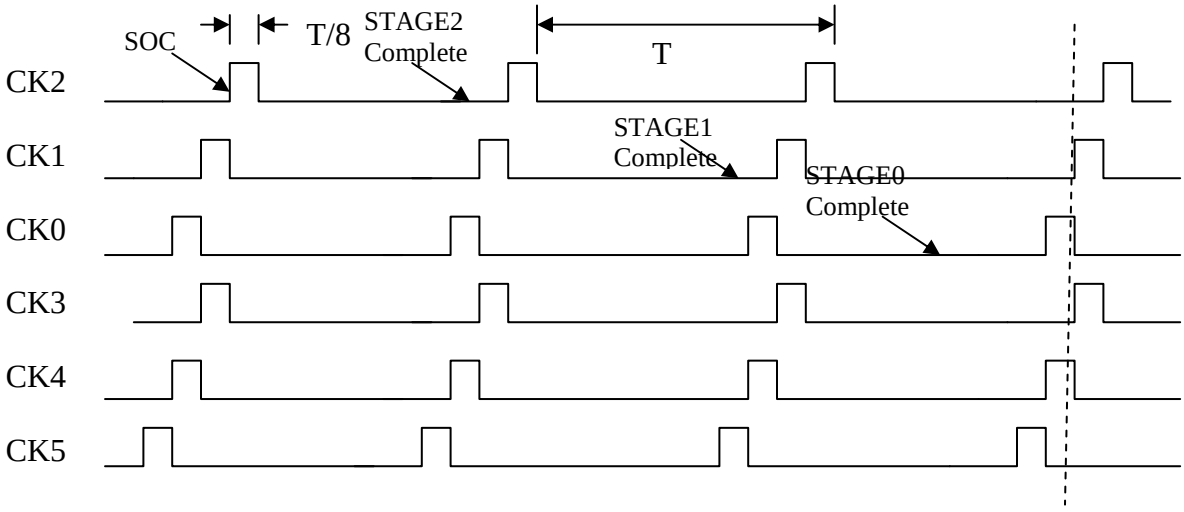
3.3) Each DAC subtraction section reduces the range of the input signal by a factor of 4 which the amplifier must make up to give the next section a full input signal. Thus the required gain is X4 (2-Bits or 2^2 or 12 DB).

3.4) The pulse width of these signals sets the sampling time of the first sample and hold circuit. Its width is determined by the need for that sample and hold to settle to full LSB accuracy. The time between pulses is long enough for the ADC/DAC and analog difference amplifier to settle before the second stage sampling begins. That sampling must be finished before a new sample can start at VIN. Therefore $\boxed{T_{TOTAL} = 2T_{S\&H} + T_{ADC/DAC-SETTLE}}$

3.5) The number of delay stages in the shift registers are:

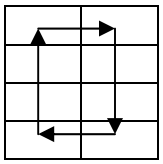
In SR2: 2 stages
In SR1: 1 stage

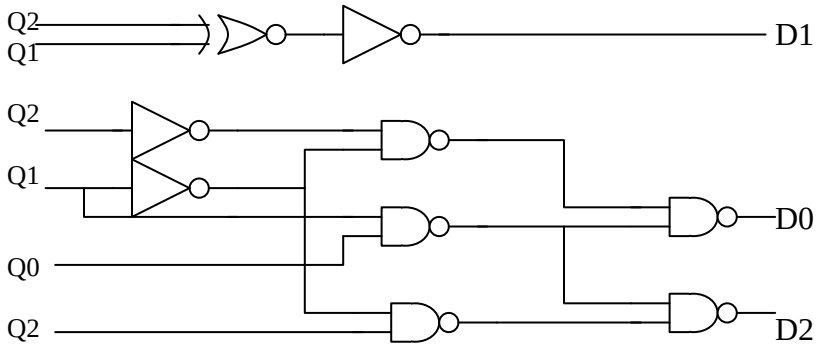
3.6)



4.) Transistors M5 and M6 are a transmission gate which is closed when A is LOW (and $\overline{A}$ from the inverter on the left is HIGH). This TG connects B to the output. If A is HIGH, transistors 1 and 2 function as an inverter between B and the output. (When A is LOW, transistors 1 and 2 cannot be turned on by B because the source voltages are incorrect. Transistor 2 will have its source at VDD but it is an N-channel device. Similarly, transistor 1 is a P-channel device with its source at GND when A is LOW.) Thus: $OUT = \overline{A} \cdot B + A \cdot \overline{B} = A \oplus B$. This is a very poor implementation of the XOR gate because the output is not fully restored to VDD and GND levels. The output signal is almost always taken from one of the two input signals and so the speed and logic levels will degrade in successive stages of this type.

5.) This is a straightforward counter design. One way to find the sequence is to traverse a 3 variable K-map clockwise through the 8 boxes. This leads to:





6.) The current drawn by the NAND gate when the ENABLE signal is HIGH is I = P/V = .005/5 = 1 ma. The portion attributable to C_L is

$$I = \frac{P_L}{V_{DD}} = \frac{C_L f_C V_{DD}^2}{V_{DD}} = C_L f_C V_{DD} = 4 \cdot 10^{-11} 2.5 \cdot 10^6 5 = .0005 ma.$$

The NAND gate alone, therefore, would draw 1ma – 0.5 ma = 0.5 ma.