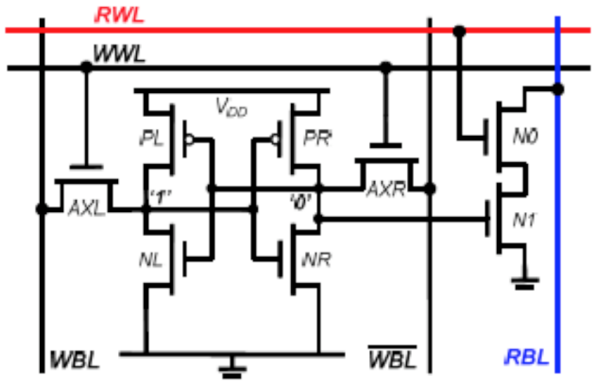
EECS 151/251A Homework 8

Due Wednesday, March 21st, 2018

Problem 1: 8T SRAM Cells

Consider the 8T SRAM cell given below. With this design, there is a Write Word Line (WWL) that is used to write the values of Write Bit Line (WBL) and $\overline{WBL}$ into the cell, and a separate Read Word Line (RWL) that is used to read the content of the cell on the Read Bit Line (RBL).



a) Determine which transistors are involved in a Write operation, and comment on their relative sizing.

> Same as 6T SRAM cell, AXL and AXR need to be stronger than PL and PR.

b) For the same cell, determine which transistors are involved in a Read operation, and comment on their relative sizing.

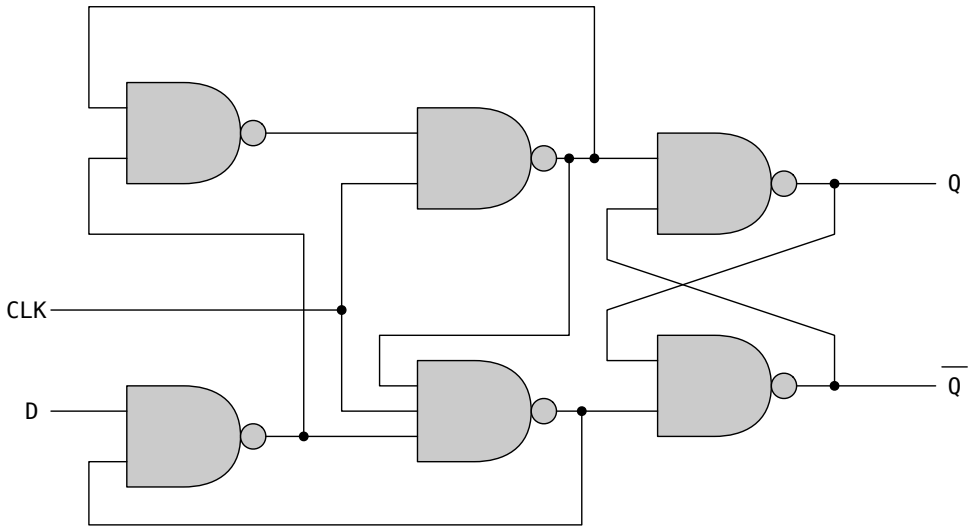
> Read is done through N0 and N1; no sizing constraints for stability.

c) Compare this structure with the 6T SRAM cell. What are the advantages and disadvantages?

Decoupled read and write operations, so less constraints on cell sizing. Also, now the cell has a separate read and write port (1R1W). Explanation: In a normal 6T SRAM cell, the pull down (PD) must be stronger than the access transistor/pass gate (PG) which must be stronger than the pull up (PU). PD stronger than PG is required for read stability, and PG stronger than PU is required for writaeability. Therefore changing PG will trade off read stability for writeability. By not using any of these transistors for reads, this trade-off disappears. In this case, both the PD and PU can be minimum sized. Note that this advantage disappears if the cells are physically interleaved (i.e., column multiplexing is used), because the non-written columns will experience reads and could have read stability issues.

Problem 2: Flip Flops

Consider the circuit below:



Part a

What type of circuit is this (flip flop, latch, combinational, other)?

It is a flip flop, since the feedback mechanism of the NAND gates causes a small pulse to set the pins in the output SR latch stage immediately following the rising edge of the clock.

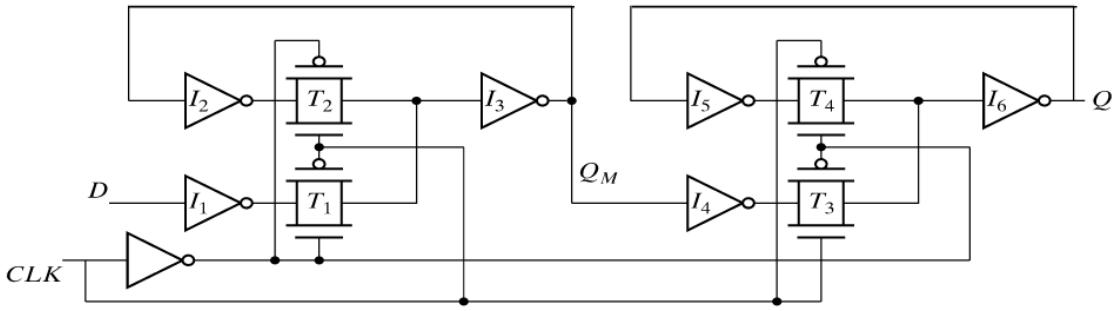
Part b

If the circuit is a flip flop, is it a positive or negative edge triggered flip flop? If the circuit is a latch, is it a transparent-high or transparent-low latch?

It is positive-edge triggered.

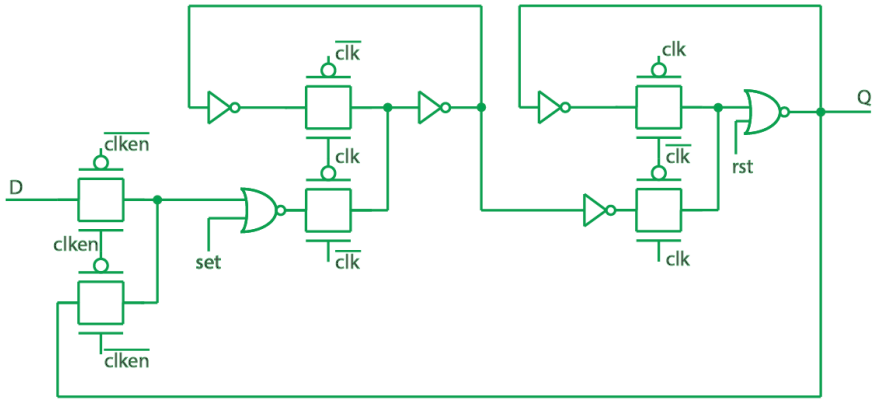
Problem 3: Register Design

Pictured here is the master-slave register circuit presented in lecture:



How would you modify the circuit to add the following features: an asynchronous reset, a synchronous set and an enable signal? Note that when the enable signal is low, the output of the register should return its previous state regardless of any clock transitions. Show the modifications you would make for each feature. Try to minimize the total number of transistors. You may use any simple 2 input gates and NMOS or PMOS transistors, and you may replace the inverters in the flip flop circuit, as long as you maintain the desired functionality.

Solution:



Problem 4: Opcode Decoder

You are given an ALU with the following opcodes:

```
ADD    0
SUB    1
AND    2
OR     3
XOR    4
XNOR   5
SLL    6
SRL    7
SRA    8
SLT    9
SLTU   10
PASSB  11 (pass B to output)
```

Part a

Write Verilog for the opcode decoder that generates the ALU opcode from the following instruction opcodes. The input to your module will be the 4-bit instruction opcode and the output will be the 4-bit ALU opcode. Branch instructions (`B*`) should be ADDs (since you will be adding an offset to the PC). Jump instructions (JMP) should pass a single input to the output.

```
BNE   0
BLT   1
BGT   2
JMP   3
ADD   4
SUB   5
AND   6
OR    7
XOR   8
XNOR  9
SLL   10
SRL   11
SRA   12
SLT   13
SLTU  14
```

Solution:

```
module alu_decoder(
  input [3:0] op,
  output reg [3:0] alu_op
);

always @(*) begin
    case(op)
    0: alu_op = 0;  // ADD
    1: alu_op = 0;  // ADD
    2: alu_op = 0;  // ADD
    3: alu_op = 11; // PASSB
    4: alu_op = 0;  // ADD
    5: alu_op = 1;  // SUB
    6: alu_op = 2;  // AND
    7: alu_op = 3;  // OR
    8: alu_op = 4;  // XOR
    9: alu_op = 5;  // XNOR
    10: alu_op = 6; // SLL
    11: alu_op = 7; // SRL
    12: alu_op = 8; // SRA
    13: alu_op = 9; // SLT
    14: alu_op = 10; // SLTU
    default: alu_op = 0; // ADD
    endcase
end

endmodule
```

Can you think of a better way to assign ALU opcodes? Explain why or why not, and include examples.

Rearrange the ALU opcodes so that they align with the instruction opcodes:

```
ADD    4
SUB    5
AND    6
OR     7
XOR    8
XNOR   9
SLL    10
SRL    11
SRA    12
SLT    13
SLTU   14
PASSB  3 (pass B to output)
```

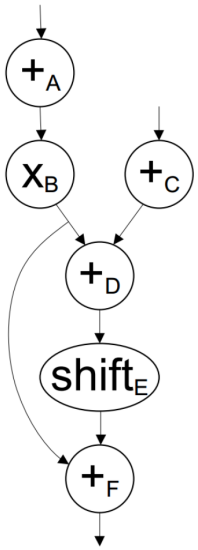
Part c

How could you implement PASSB without a dedicated PASSB instruction?

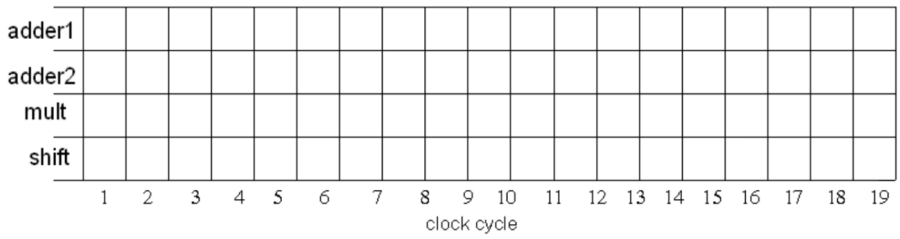
There are a number of ways to do this, but an example would be an add instruction with the A operand tied to 0.

Problem 5: Resource Utilization Chart

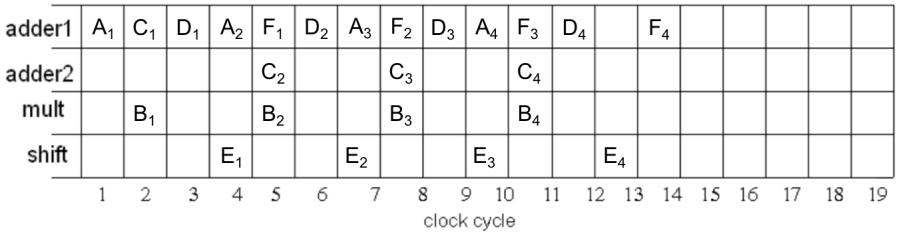
You are given a datapath that has four computation units; two adders, a multiplier, and a shifter. Each unit requires an entire clock cycle (minus flip-flop overheads) to complete its operation and is followed by a register to hold its output. You can think of all the computation units as being in parallel, where you can use them all simultaneously on any given cycle. Assume the only memory available are the registers at the output of each unit, so if the result of a unit cannot be immediately used by the next unit, you have to stall. The graph below represents an iterative operation to be completed on the datapath. Each node is labeled with the name of the computation unit that it requires plus a unique letter identifying the node. Note that there is no feedback (or loop carry dependence) in this computation.



Fill in the following resource utilization chart to show how to complete four iterations of the loop in the minimum number of cycles. Use subscripts (1, 2, 3, and 4) to indicate the iteration number. For instance, "C2" indicates node C of iteration 2.



Solution:



Problem 6: DDCA Exercise 8.12

(This is repeated. Submit your answer to Problem 4 from Homework 7 here.)

You are building an instruction cache for a MIPS processor. It has a total capacity of $4\text{C} = 2^{c+2}$. It is $N = 2^n$-way set-associative ($N \geq 8$), with a block size of $b = 2^{b'}$ bytes ($b \geq 8$). Give your answers to the following questions in terms of these parameters:

(a) Which bits of the address are used to select a word within a block?

(b) Which bits of the address are used to select the set within the cache?

(c) How many bits are in each tag?

(d) How many tag bits are in the entire cache?