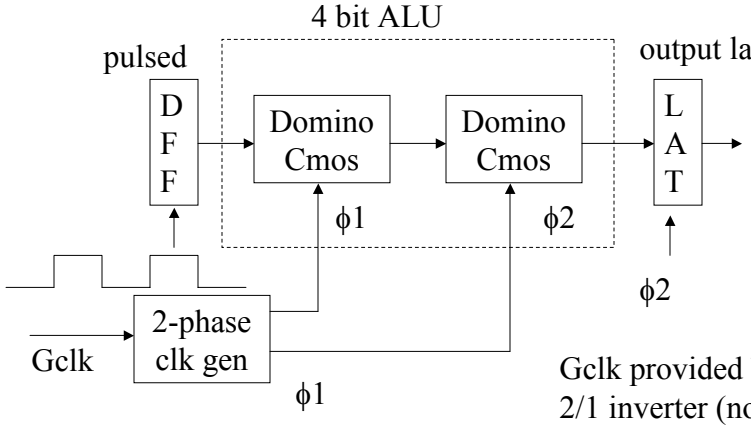
Homework #4

- All Spice problems in this homework are to be done for technologies
  - tsmc_0_18.model
  - Vdd = 2.5 V, default temp
  - I will provide a testbench for your circuit that will supply input waveforms plus a global clock.
  - You can work in teams of 2 if desired.
  - Will be worth twice that of previous assignments.
  - Do not use any GEO parameters – the default value will use worst case assumptions for source/drain capacitance
  - Use node names of VDD, GND for all power supply, ground connections in your subcircuits.

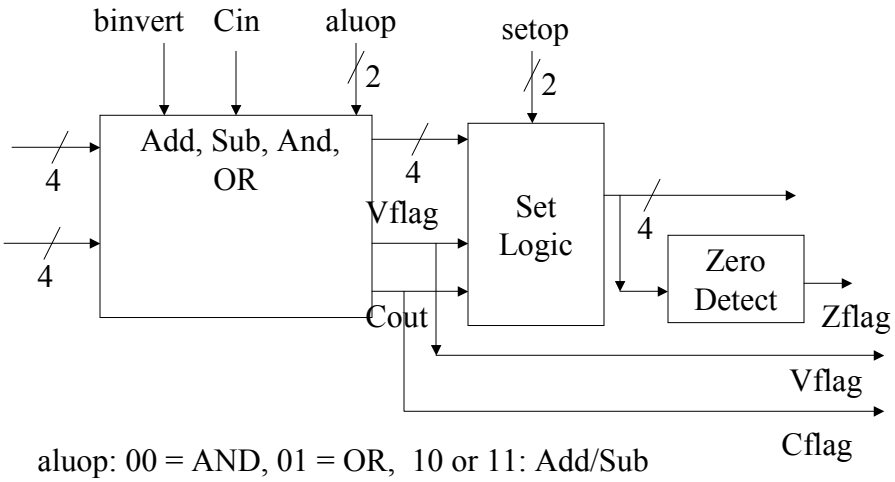
4-Bit ALU in Domino Logic

Goal: create a 4-bit ALU that uses a 2-phase domino logic pipeline.



Gclk provided by a 10X 2/1 inverter (not infinite drive)

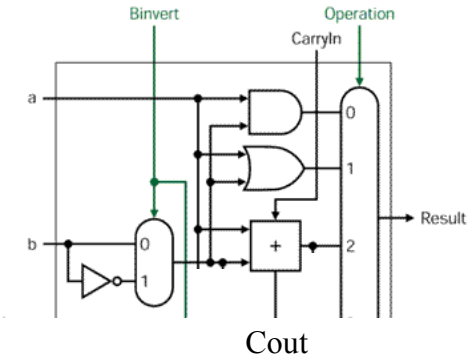
ALU Structure



aluop: 00 = AND, 01 = OR, 10 or 11: Add/Sub

setop: 00= slt, 01= sltu, 10 or 11: pass thru

Add/Sub/And/Or Cell



Basic cell

Operations: AND, OR, ADD, SUB

If SUB, then Carryin LSB = 1, Binvert = 1.

Overflow Flag Logic

Overflow logic depends on whether doing an addition or subtraction:
if (addition) overflow = (A*msb* and B*msb* and (not Smsb)) or
((not A*msb*) and (not B*msb*) and Smsb)

i.e. For addition, if sign bits of operands are the same, but the result sign bit is different, then OVERFLOW has occurred. Smsb is the most significant bit of the result.

If (subtraction) OF = (Amsb and (not Bmsb) and (not Smsb)) or
((not Amsb) and Bmsb and Smsb)

Note: In all cases, Binvert = 1 for subtraction, Binvert = 0 for add

Set Logic

SLT, SLTU : output is non-zero if A < B; ALU always does a SUB operation, SET logic will output a zero or non zero value based on flags.

if SLT (signed comparison) then
Let Nf = sign bit of result, OF = overflag
Result (LSB) = ((not OF) and Nf) or (OF and (not Nf))
Result other bits = 0.

If SLTU (unsigned comparison) then
Result (LSB) = not(CarryFlag)
Result other bits = 0.

Note that a Set operation always produces as a result either ‘1’ (‘0001’) or ‘0’ (‘0000’).

Zero Detect

Zero Detect outputs a ‘1’ if result is zero, else outputs a ‘0’.

Can be folded into SET logic if desired.

Some Challenges

The logic I have given simply specifies in a simple manner the functionality to be achieved.

Feel free to redesign the logic in a more efficient manner to fit domino logic.

It will be your decision as to what will be in phase 1 and phase 2 of the domino logic block.

You are responsible for generating your 2 phase clocks.

The testbench will only provide uncomplemented inputs, you MUST HAVE DFFS ON ALL INPUTS. You will probably need to provide both complemented and uncomplemented inputs to your circuits.

You must use domino logic for your logic approach.

Rankings of Designs

I will rank the designs via the Power-Delay-Product (PDP). The PDP is an accepted metric for measuring the speed-efficiency of a design.

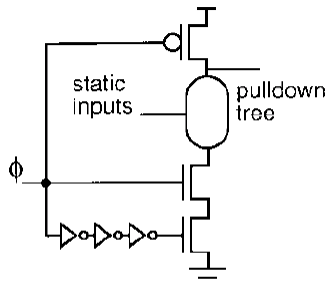
For a given suite of test vectors, I will measure the average power per clock period and multiply this by clock period. Lower PDP values are good, the design with the lowest PDP will have a ranking of '1'.

You can lower your PDP by lowering your power usage and/or by reducing the clock period.

The upper third of the class will get 25 pts added to any test grade. The middle third will get 12 pts added to any test grade. The lowest third gets no extra points.

Pulsed DFF

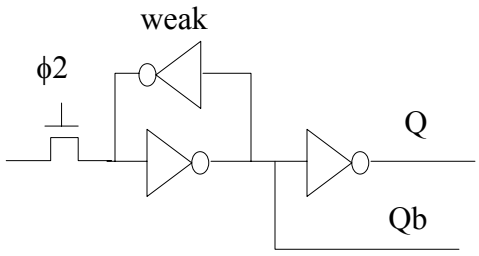
Simple Pulsed Latch



Static to Domino
(Replaces dynamic gate)

Use this design for your pulsed DFF, there is no need for weak cross coupled inverters on output.

Static Output Latch

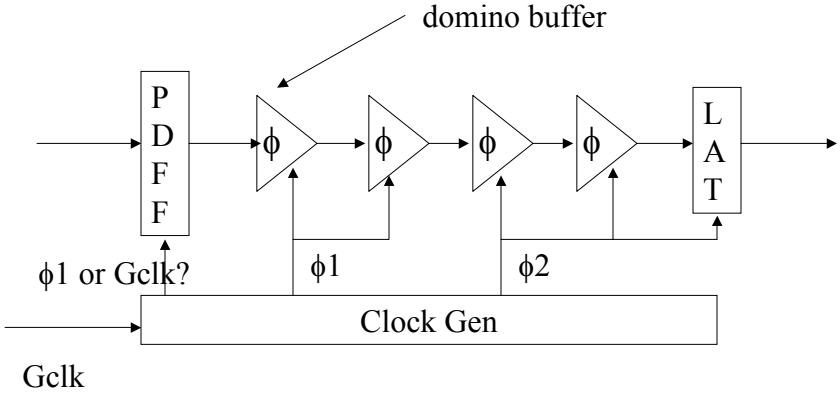


Note that you have both Q and Qb available.

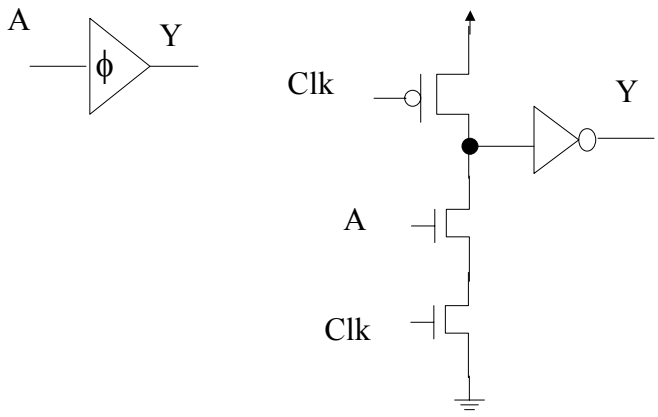
This means that your logic can implement either the true or complement version of the logic.

Debugging

Getting the clocks right is probably the most difficult challenge. It is suggested that you make a simple 1-bit data path like shown below to test your clocking strategy.



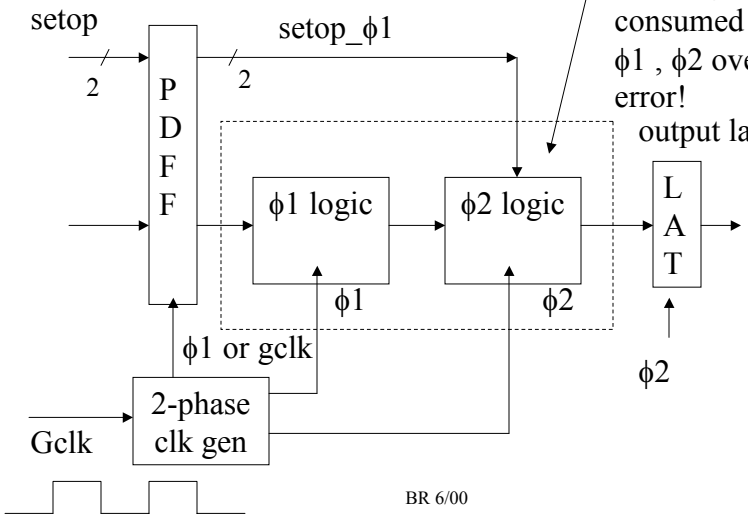
Domino Buffer



A Pitfall

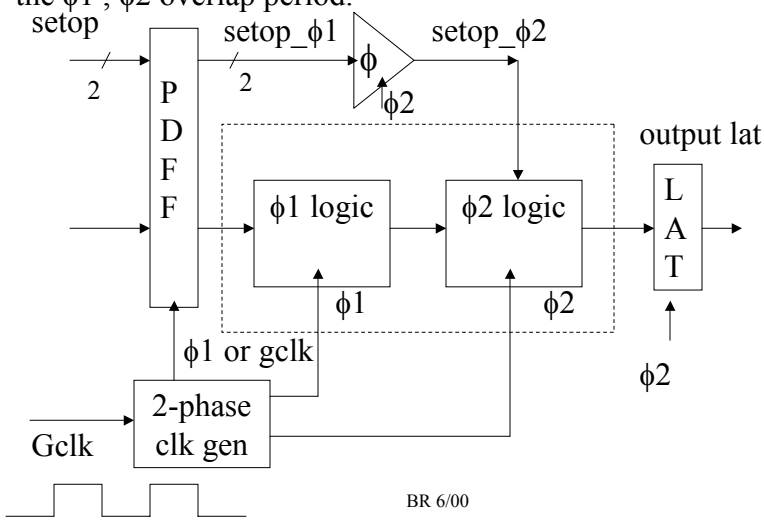
A problem! setop_ϕ1 is only valid during ϕ1 . If setup_ ϕ1 not consumed during ϕ1 , ϕ2 overlap, then error!

Assume SET logic is clocked by ϕ2



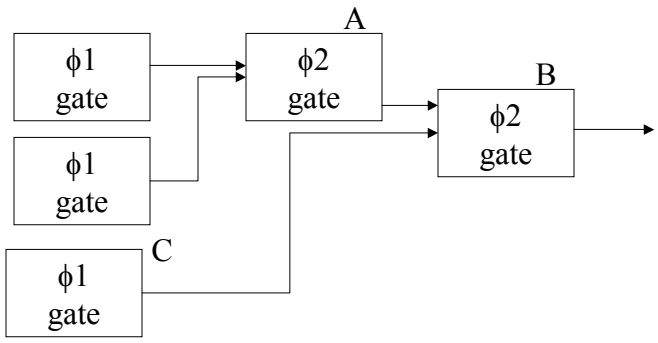
How to Fix

Control lines for ϕ2 logic should be buffered by domino buffers unless can guarantee that control line values are consumed during the ϕ1 , ϕ2 overlap period.



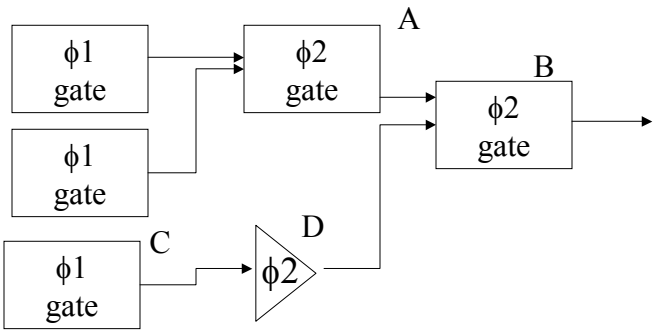
Another View of the Pitfall

You may be tempted to do something like what is shown below (a ϕ1 gate is clocked by ϕ1, a ϕ2 gate is clocked by ϕ2).



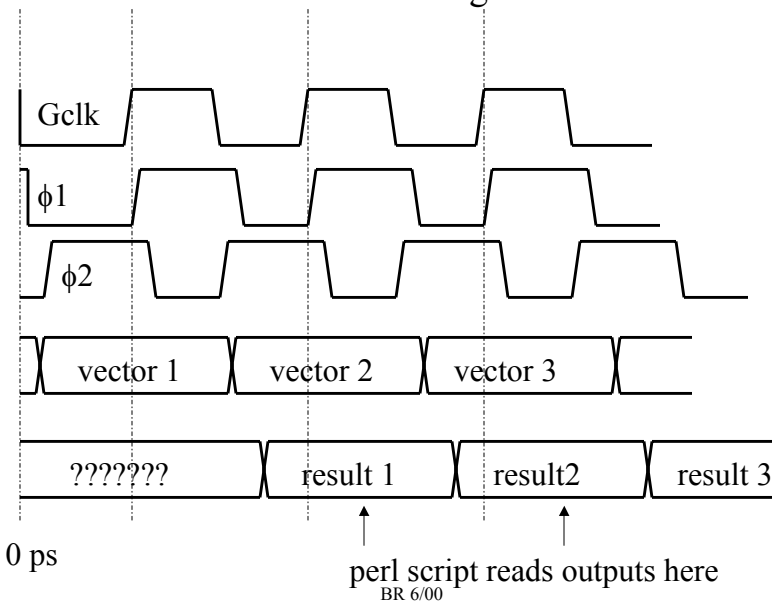
Note that gate B has inputs from both ϕ1 logic and ϕ2 logic. This means the clock overlap period must be at least as long as the gate delay of Gate A plus the length of time it takes for Gate B to consume the output of Gate C.

A Fix



Adding the domino buffer D between gates C and B now means all gates either have all ϕ1 inputs or all ϕ2 inputs. The clock overlap period only has to be long enough for gates A, D to consume their inputs.

Clocking



Report

- I want to see complete transistor-level schematics for every gate, including transistor sizes
  - I do not want Cadence files – I want to see screen captures or pictures of the schematics in your report.
- I want to see a complete schematic of the datapath at the gate level
- You need to show me the critical path in your circuit.
  - Provide data that proves this is the critical path by showing a test vector passing, then failing the simulation because the clock period was reduced and the test vector exercised this critical path (I don't need to see spice waveforms – just the clock period values, the input test vector, and the output test vector that passed/failed.)
  - For the same clock period that FAILED above because the critical path was exercised, show a test vector that passes at this clock period because is uses a short path.