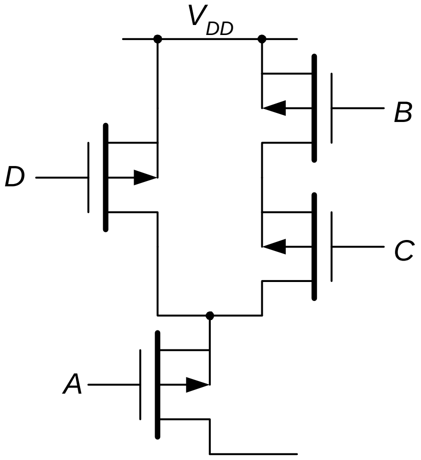
The University of Sheffield

DEPARTMENT OF ELECTRONIC AND ELECTRICAL ENGINEERING
Spring Semester 2002-2003 (2 hours), Answers to Introduction to VLSI

1. i)



(4)

ii) Looking at the pull-up network, combining the terms in series as logical AND, and in parallel as logical OR and inverting the terms corresponding to each input we have:

$Y = \overline{A}.\left(\overline{D} + \overline{B}.\overline{C}\right) = \overline{A}.\overline{D} + \overline{A}.\overline{B}.\overline{C}$ **(4)**

iii) Looking at the pull-down network, the transistor driven by *A* can be 1μm wide, the transistors driven by *B* and *C* are in series with that driven by *D* and so all of these must be 2μm wide.

Looking at the pull-up network, the transistor driven by *A* is in series with the other transistors and so, taking account of the lower mobility, the width will be 4μm. Similarly, the transistor driven by *D* will be 4μm wide. The transistors driven by *B* and *C* are in series and will be 8μm wide. **(4)**

To drive a load capacitance of 140fF requires a set of tapered buffers. Assuming that the gate is the first element of the set of buffers, the input capacitance depends on which input is being driven but the average input capacitance is 8x a minimum width transistor.

$C_{\min} = \dfrac{eW_{\min}L_{\min}}{t_{OX}} = 0.8625\,fF$, so C_{in}= 6.9fF.

The number of stages is Naperian log of the ratio between the load and the input capacitance. So, *stages* = ln(140/6.9) = 3.

This is odd but taking the first stage out, another two stages are required and if these are inverters then the output will be the same as the logic of the gate. Each inverter should have an input capacitance *e* times bigger than the previous one and so the first inverter will have an input capacitance of 18.76fF, that is the *n*FET is 7.25μm wide, the second inverter will be 51fF, that is the *n*FET is 19.7μm wide. **(8)**

2. There are a number of points that should be mentioned in the answer to this:

Risk in design – analogue and mixed-signal designs are inherently more difficult. By purchasing an existing analogue part, this risk is offset.

Analogue circuits – especially ones in which noise is an issue do not co-exist well on the same substrate as the digital section, which generates noise.

Sourcing of standard parts and reliability of supply (second sourcing, obsolescence). How guaranteed will the supply be of the part over the lifetime of the product and are any second-sourced parts entirely compatible.

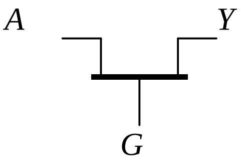
Creation and ownership of IPR and ability to develop it as technology moves on. The advantage of developing the whole product is that it can be used in derivative or subsequent products.

Cost and reliability issues. Each option will have its own associated costs and the resultant product will have have a particular reliability and lifetime. This reliability will translate into cost.

Packaging and interfaces. The design in two packages will have implications for integration at the PCB level, for example. **(8)**

Assuming that a designer person-year costs ~$500K then the cost of the digital design will be $250K whilst the cost of the mixed signal design will be $375K. Both of these costs are amortised across 200,000 parts. That is, $1.25 for the digital part and $1.875 for each mixed-signal part. Each packaged digital device costs $3 but only 70% work, giving a manufacturing cost of $4.285 per working device. Similarly, each working mixed signal device costs 5/0.65=$7.69. The cost of the analogue device, $4, must be added to these costs giving 1.25+4.285+4=$9.535 for option a). The cost of the mixed signal part will be 1.875+7.69=$9.565. There is little to choose between these two prices and the estimates are probably so crude that the difference of $0.03 is not real. In this case, the foregoing arguments must be applied. **(12)**

3. Looking at the pass transistor:



When *G* is high, the transistor conducts and, assuming that *A* is the driven node, *Y* should follow *A*. Consider the drain current for the transistor:

$$I_{DS} = b_N \cdot \left(V_{GS} - V_T - \frac{V_{DS}}{2} \right) \cdot V_{DS}$$

By differentiating the current the small signal resistance between *A* and *Y*, *r*, can be found.

$$\frac{d}{dV_{DS}} I_{DS} = \frac{1}{r} = b_N \cdot (V_{GS} - V_T - V_{DS})$$

$$r = \frac{1}{b_N \cdot (V_{GS} - V_T - V_{DS})}$$

Assume that *A* is 0V and *G* is at V_{DD}. In this case the drain current is 0 but *r* is:

$$r = \frac{1}{b_N \cdot (V_{DD} - V_T)}$$

When *A* is driven to V_{DD}, point *Y* cannot rise higher than V_{DD}-V_T without the transistor cutting off. At this point, however, the transistor is saturated because V_{DS} is greater than 0V (remember $V_{DS} > V_{GS}$-V_T) and:

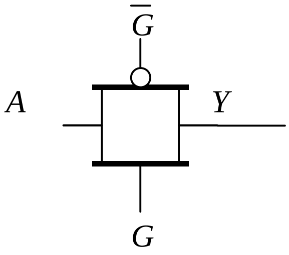
$$I_{DS} = \frac{b_N}{2} \cdot (V_{GS} - V_T)^2$$

$$\frac{d}{dV_{DS}} I_{DS} = \frac{1}{r} = 0,$$

r is infinite and I_{DS}=0. **(6)**

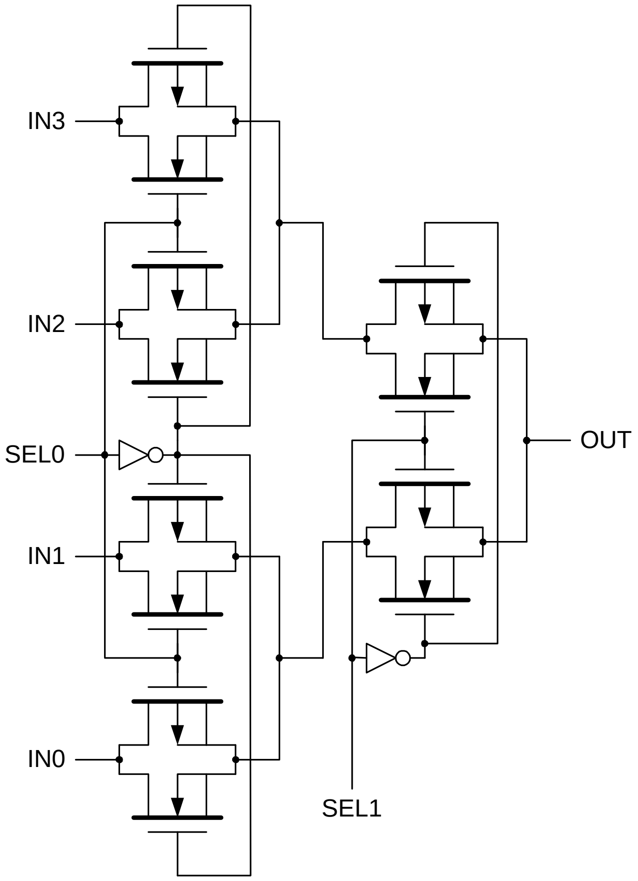
Consequently, whilst the logic 0 is well defined – it is at the right voltage and the resistance between the driven and output nodes is low. The logic 1, conversely, is degraded by V_T and the resistance between the driven and output nodes is high (this means that the node is more susceptible to noise). **(2)**

The problems of the pass transistor can countered by replacing the pass transistor by a transmission gate. The transmission gate is formed by an n and pMOSFET connected in parallel. Nominally, the devices would be sized for equal values of *b* and the gates are driven by a logic level and its inverse.

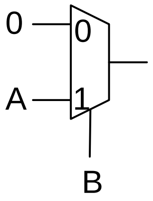


The nMOSFET defines a 0 well but 1 badly, the pMOSFET defines a 1 well and 0 badly. **(4)**

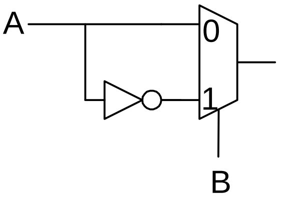
3. cont



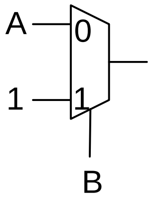
i



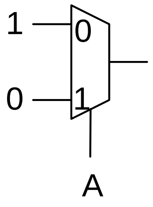
ii



iii

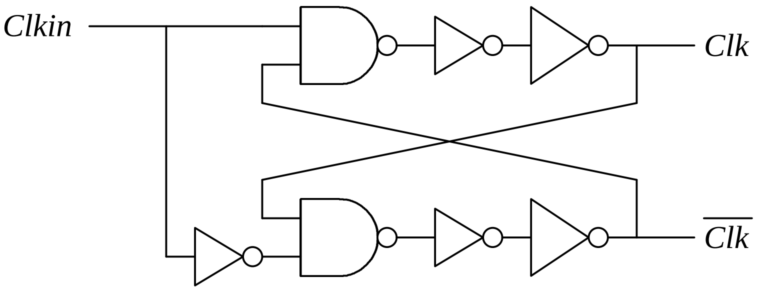


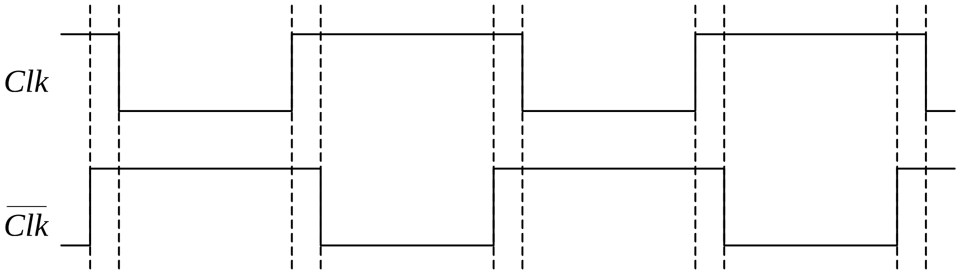
iv



(8)

4.





The output signals are driven through a set of tapered buffers and when one of these output signals first becomes 1 (the other will still be 1), it must pass through the other NAND gate changing the state to 0 and the other set of tapered buffers before the other output signal changes from 1 to 0. Thus, both outputs will be 1 for a period of time that is determined by the delay through the chains of buffers. The tapering can allow the load capacitance driven by the clocks to be managed. Naturally, such a generator would be used to supply non-overlapping clocks to more than just one flip flop. **(6)**

When *Clk* goes high, the *G* input to the first latch, the master latch, changes from low to high and traps the inverse of the value of *D* at the input in Q_m. At the same point in time, the *G* input of the second latch, the slave latch, goes low and allows the inverse of the value of Q_m to pass through to the output, *Q*. Conversely, when *Clk* goes low, the master latch becomes transparent. However, at the same instant, the slave latch freezes the inverse of the value of *Q* at *Q* thus maintaining the value of *D* sampled at the rising edge of clock in the master latch at the output. If, however, both of the *G* inputs were to go low at the same time then both latches would be transparent and changes at the input could propagate through to the output. **(6)**

One of the key advantages of this methodology is the level of automation involved. The design can be coded in an HDL at a *reasonably* abstract level and synthesised to meet constraints. Generally, the style in which HDL designs are written is called *register-transfer-level* (RTL). Where logic is coded as:
$signal\text{-}out = signal_a \{\mathrm{OP}\ signal_b\}^n$
Where OP might be logical or arithmetic and the *signals* might be logical or arithmetic quantities. Generally, it is expected that the *signals* are stored in registers and the new value of *signal-out* would be generated from data stored in registers at one rising-edge of the clock ($signal_a$, $signal_b$, etc.). The new value of *signal-out* being transferred into a register at the next rising-edge of the clock – hence the name.

4. cont For example, in VHDL:

```
process (clk)
begin
   if clk’event and clk = ’1’ then
      if reset = ’1’ then
         regs <= initialisation_values;
      else
         regs <= some_function_of_inputs_and_regs;
      end if;
   end if;
end process;
```

To implement this sensibly in a real technology requires that the clock be applied at the same instant across the entire design (subject to a small skew). The delays in the real logic and interconnect ensure that if this is the case then the input at a flip flop due the the last state of the registers will be stored in a flip flop before the new values propagate through the the input of any flip flop. Without this constraint, it would be impossible to design a clocked design reliably. The clock is distributed via a balanced tree of buffers sufficient to meet the drive requirements across the chip and with equal delay paths to meet the skew requirements. **(6)**

To make this methodology work there are some additional constraints that must be applied: there can be no feedback, do not gate clocks. This is particular the case if the design is to be testable. A typical design has a large number of internal registers with low observability and poor inherent ability to drive these registers to a defined state to observe a potential fault. The needs of test have to be built into the design process – right from the start. This will ensure that, when scan paths or BIST are incorporated, the design will be *testable*. That is, so that it can be ensured that a defined proportion of potential faults can be tested for – this is known as fault coverage. **(2)**