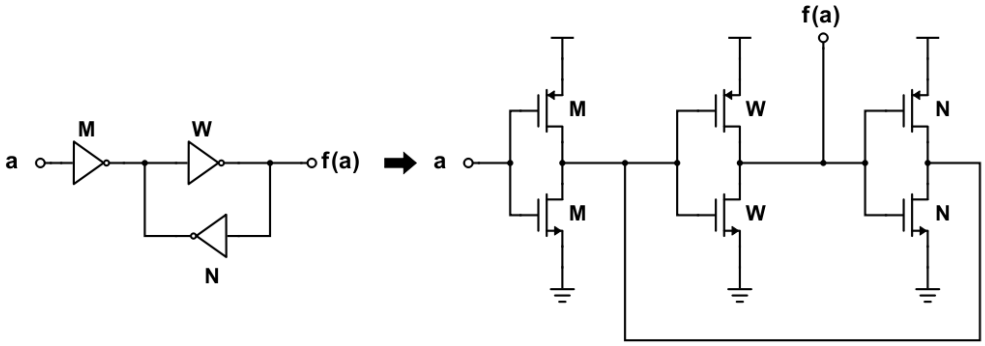
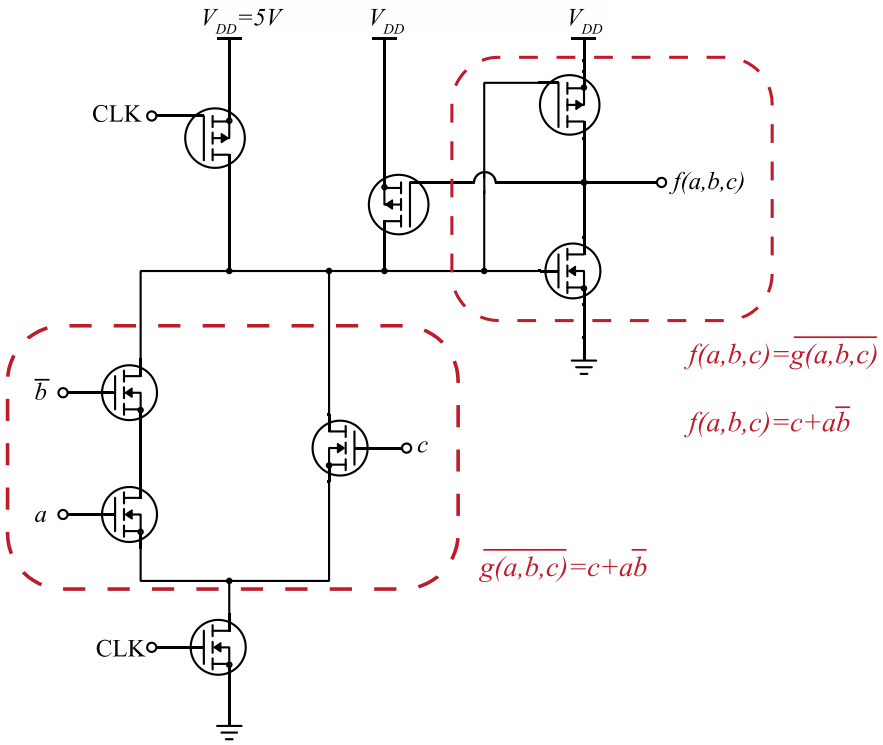
Qn 1 (a) Schematics of the non-inverting Schmitt trigger buffer is presented in the left side of below figure. Resistance of MOSFETs are inversely proportional with their aspect ratio i.e. W/L, thus the first two inverters needs to have increasing order in terms of aspect ratio. The feedback inverter needs to have the smallest possible k=k'W/L for higher performance. Therefore, the resulting aspect ratios can be seen in the right side of below figure.



Some alternative solutions:

[40%]

(b)



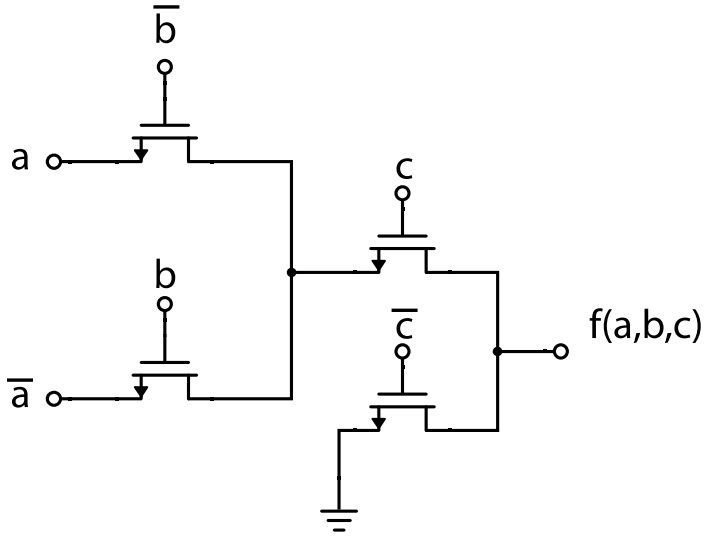
When the clock is low, pMOS on the left top is on and nMOS on the bottom connected to CLK is off. Thus, the input of the inverter on the right is high, and the output is *f(a,b,c)=0*, when the clock is low.

When the clock becomes high, then the pMOS controlled via the clock turns off and the nMOS controlled via the clock turns on. Consequently, the input of the inverter becomes $g(a,b,c) = \overline{c + a\bar{b}}$ as shown in above figure. Hence, the output is $f(a,b,c) = \overline{g(a,b,c)} = c + a\bar{b}$.
The (stable) feedback transistor is there to provide input to the inverter in case all other input paths are open, i.e., when f=1.

[30%]

(c) This logic function can be implemented with 4 transistors, but any solution lower than 8 transistors (except CLK transistors) will get full mark.

Two nMOS transistors with coupled drains on the left implement XOR operation with pass-transistors. nMOS controlled with *c* perform AND operation with the XOR input at its source. Thus, the output is "*a* XOR *b*" provided that *c=1*. Low logic output is provided with the bottom nMOS , when *c=0*, and thus ouput is provided for all possible input logic values. Hence, *f(a,b,c)= (a XOR b)c.*

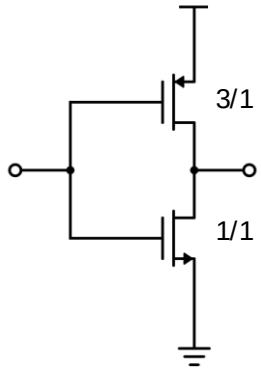


[30%]

Assessor's comments:

The question was well answered. Many provided inverting Schmitt trigger designs instead of non-inverting in part (a), and mistakenly found the function complement in part (b), while some did not include the clock signal. In the last part a common mistake was the use of input configurations with high impedance output.

Qn 2. (a) The aspect ratios are calculated to ensure $k_p=k_n$



[30%]

(b) During a single period of duration $T=1/f$, the inverter charges the load capacitor C to V_{dd} and discharges it back to zero, consuming twice the energy stored on it. Thus, the energy consumed during one period is

$$\Rightarrow \quad E = CV_{dd}^2$$

The power consumed by the inverter is then given by

$$\Rightarrow \quad P = \frac{CV_{dd}^2}{T} = CV_{dd}^2 f$$

$$P = CV_{DD}^2 f = (45 \times 10^{-15} F)(5V)^2(4.1\times 10^9 Hz) = 4.61mW$$ [20%]

(c)

- $C_P = 17.7\ fF$
- $C_{in} = 11.8\ fF$
- $C_T = \ 29.5\ fF$

$C_P = 2 \cdot (1\mu m)C_{gdn} + (1\mu m)C_{dbn} + 2 \cdot (3\mu m)C_{gdp} + (3\mu m)C_{dbp} = 17.7\text{fF}$
Multiplication by 2 due to Miller Effect

$$C_{in} = (1\mu m) \cdot C_{gdn} + (1\mu m) \cdot C_{gsn} + (3\mu m) \cdot C_{gdp} + (3\mu m) \cdot C_{gsp} = 11.8\text{fF}$$

$$C_T = C_p + C_{in} = 29.5\ fF$$

(No points deducted for not including Miller Effect.) [20%]

(d) t_{pLH} is defined as the time it takes for the inverter output to get from low logic.

For the sake of simplicity, assume that the transistor is always in the saturation mode. The average channel (pMOS source-drain) resistance, R_p, is calculated as

$R_p = \frac{V_{dd}}{I_{p,sat}}$, where $I_{p,sat} = \frac{1}{2} k_p \left(V_{dd} - \left|V_{Tp}\right|\right)^2$ is the saturation current.
The output voltage transition described by

$$V_{out}(t) = \left(1 - e^{-\frac{t}{R_p C_t}}\right) V_{dd},$$

where we have assumed that the input inversion was made at $t = 0$. Hence, the time for V_{out} to reach $V_{inv} = \frac{V_{dd}}{2}$ is given by

$$t_{pLH} \approx \ln(2)\, R_p C_T = 0.7 \frac{C_T V_{dd}}{\frac{k_P}{2}\left(V_{dd} - \left|V_{Tp}\right|\right)^2} \approx \frac{C_T V_{dd}}{k_P \left(V_{dd} - \left|V_{Tp}\right|\right)^2} \approx \frac{C_T}{k_P V_{dd}}$$

$$t_{pLH} \approx \frac{C_T V_{DD}}{k_P \left(V_{DD} - \left|V_{Tp}\right|\right)^2} \approx 26ps$$

[30%]

Assessor's comments:

Popular question very well answered. Most were able to size the transistors and calculate the power. Some had difficulty in finding the intrinsic capacitances, while many attempted to calculate the low-to-high propagation delay by considering both linear and saturation regions instead of the simplifying assumption of using only the saturation region. There were several very good answers.

3. (a) Both CPLDs and FPGAs are based on logic blocks and programmable interconnects. However, CPLD's logic blocks contain multiple macrocells (typically 4 to 20) which provide product term arrays. FPGA's logic blocks are made of logic elements (LEs), which consist of lookup tables (LUTs), registers, etc. Compared to macrocells they are much more configurable and provide several extra features to improve performance and minimize wasted logic resources. For example, the LUT is key to the creation of product functions out of combinational logic in a FPGA. The LUT replaces the product term array found in CPLDs. FPGAs use 4 or more-input LUTs to create complicated functions. [10%]

(b) Any number of 5-variable functions can be implemented by using two 4-LUTs. For example, if we cascade the two 4-LUTs by connecting the output of one 4-LUT to an input of the other, then we can realize any function of the form

$f = f_1(x_0, x_1, x_2, x_3) + x_4$

$f = f_1(x_0, x_1, x_2, x_3) \cdot x_4$.

[20%]

(c) (i)

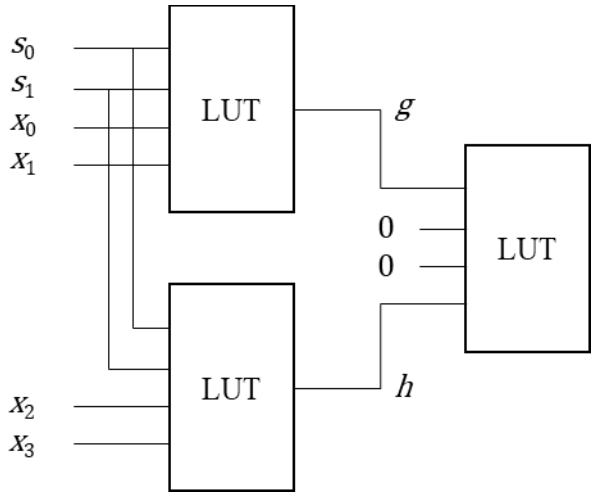
1	0	0	1	1	0	0	0	1	0	0	0	1	0	0	1

[20%]

(ii) Using the expression that defines the 4-to-1 multiplexer :

$$f = \bar{s_1}\bar{s_0}x_0 + \bar{s_1}s_0x_1 + s_1\bar{s_0}x_2 + s_1s_0x_3$$

one can write : $g = \bar{s_1}\bar{s_0}x_0 + \bar{s_1}s_0x_1$ and $h = s_1\bar{s_0}x_2 + s_1s_0x_3$



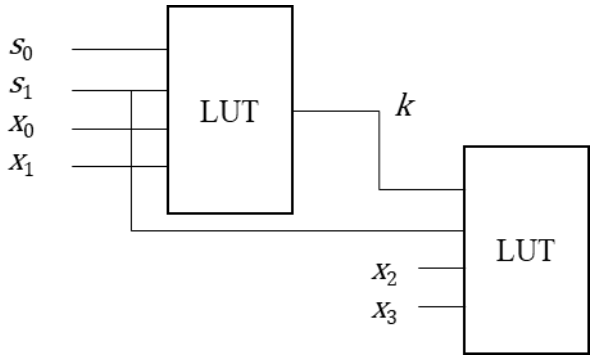
[30%]

(d) Two LUTs can be used instead of 3. E.g. the first LUT decides between x_0 and x_1, and the second LUT between x_2 and x_3, but in this case information on s_0 must be passed from the first LUT to the second LUT, i.e. term $s_0 s_1$

$$k = \bar{s_1}\bar{s_0}x_0 + \bar{s_1}s_0x_1 + s_1s_0$$

$$f = \bar{s_1}k + s_1\bar{k}x_2 + s_1kx_3$$

$$= \bar{s_1}\bar{s_0}x_0 + \bar{s_1}s_0x_1 + s_1\bar{s_0}x_2 + s_1s_0x_3$$



[20%]

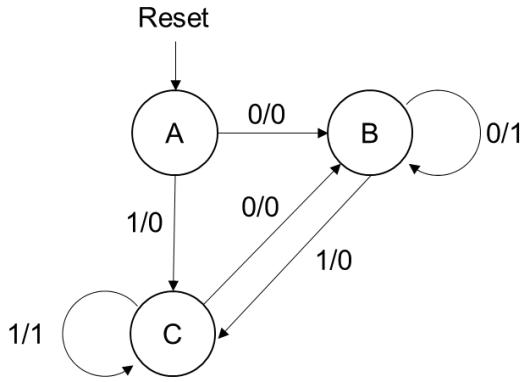
Assessor's comments:

In this question almost everyone found reasonably straightforward LUT logic implementation. Most missed key differences between CPLDs and FPGAs, and hardly anyone could find the minimum number of LUTs in part (d).

4. (a) EPROM and EEPROM are non-volatile memories, i.e. can retrieve stored information even after having been turned off and back on. This is the opposite of SRAM (volatile memory) which needs constant power to prevent data from being erased. EPROMs can be reprogrammed by using UV light, in comparison to EEPROMs, which can be electrically reprogrammed. [20%]

(b) (i) It is a Mealy configuration. The output depends on both present state (y) and primary inputs (x). This can be seen from the FSM output, i.e. the PROCESS (y, x) block in the VHDL code. [20%]

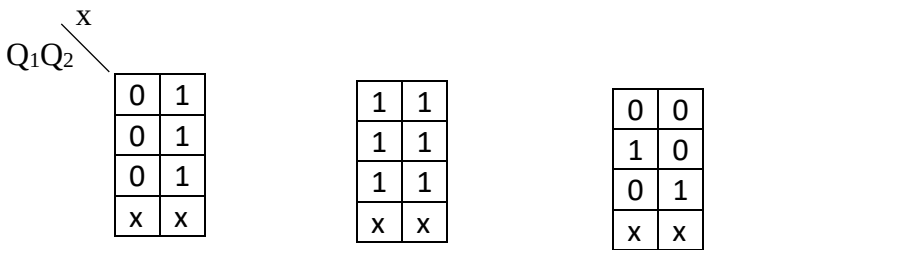
(ii) It is a sequence detector that produces z = 1 when the previous two values of x were 00 or 11; otherwise z = 0.



Present state		Next state for Q_1Q_2		Output	
	Q_1Q_2	x=0	x=1	x=0	x=1
A	00	01	11	0	0
B	01	01	11	1	0
C	11	01	11	0	1
	10	--	--	-	-

[40%]

(iii)



$D_1 = x$ $D_2 = 1$ Output = $\bar{x}\overline{Q_1}Q_2 + xQ_1$

[20%]

Assessor's comments:

The most popular question, with high marks in general. Lots of people did not know how to compare various programming technologies. The VHDL question was answered very well by most although very few were able to describe its functionality.