# EE 201 Midterm Exam 

Tufts University

7 March 2019

## SOLUTIONS

| Question | Points |
| ---: | :--- |
| Numbers | 4 |
| Boolean equations and circuits | 12 |
| Number circuits | 8 |
| What's in the box? | 4 |
| Timing | 4 |
| Building proteins | 5 |
| VHDL | 8 |
| Total | 45 |
| Bonus | 1 |

## Question 1: Numbers

(a) $[1 \mathrm{pt}]$ Write 11100011 in hex.

```
E3
```

(b) [1 pt] Write 11100011 in decimal, assuming it is an unsigned number.

$$
128+64+32+2+1=227
$$

(c) [2 pts] Write 11100011 in decimal, assuming it is a 2's complement signed number.

This is a negative number, so to get the positive value, we need to flip all the bits and add one: $00011100+1=0001110116+8+4+1=29$

So this represents -29.

## Question 2: Boolean equations and circuits

(a) [4 pts] Write a minimal boolean equation for this truth table.

| $A$ | $B$ | $C$ | $D$ | $Y$ |
| :---: | :---: | :---: | :---: | :---: |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | X |
| 1 | 0 | 0 | 0 | X |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | X |
| 1 | 1 | 1 | 1 | X |

$$
\bar{A} D+C \bar{D}
$$

(b) [4 pts] Implement the boolean equation $A B+\bar{A} C$ using only NAND gates. Assume only $A, B$, and $C$ are available, not their complements (i.e., $A$ can be an input to your circuit, but $\bar{A}$ cannot).

(c) [4 pts] Re-implement the same boolean equation using a 4:1 multiplexer and an inverter.


## Question 3: Number circuits

[8pts] Draw a circuit below that analyzes the 8 -bit signed 2 's complement input $N$ and produces three results:
(a) Output A should be high if the input number is negative, and low otherwise.
(b) Output B should be high if the input number is zero, and low otherwise.
(c) Output C should be high if the input number is greater than +31 , and low otherwise.
$N_{7}$ is the most significant bit. You may use gates with as many inputs as you need (e.g., 4-input AND, 9-input NOR).


## Question 4: What's in the box?

[5 pts] For each of the four waveforms below, identify what is in the box. It could be an SR latch, a D latch, a D flip-flop, or a single combinational logic gate (NAND, XOR, etc).

a: OR gate, b: D flip-flop, c: AND gate, d: D latch

## Question 5: Timing

[5 pts] Given the timing specifications below, what is the maximum clock speed (in Hz ) that this circuit can run at? Assume the input can keep up with whatever rate you choose. Leave your answer as a fraction if necessary.

| Gate | $t_{p d}(\mathrm{~ns})$ | $t_{c d}(\mathrm{ps})$ |
| :--- | :---: | :---: |
| 2-input XOR | 23 | 12 |
| D flip-flop Clock-Q | 15 | 8 |
| D flip-flop setup time | 12 ns |  |
| D flip-flop hold time | 7 ns |  |



From the rising edge of the clock, it takes $15+23 \mathrm{~ns}$ for the value to propagate to the input of the flip-flop. Then we have to wait for the setup time, for an additional 12 ns . The total delay is 50 ns , giving a max frequency of 20 MHz .

## Question 6: Protein decoder

Once a "start codon" is found in a DNA strand, a ribosome begins matching codons to amino acids to build the protein. Two such amino acids are Asparagine (coded by either AAT or AAC) and Lysine (coded by either AAA or AAG).
(a) [4 pts] Draw the state diagram for an FSM which detects Asparagine and Lysine.

- There should be two outputs, one for each amino acid.
- The output should go high for one cycle when the third nucleotide is received (i.e., not delayed) and a matching sequence is detected.
- Note that a repeated sequence (e.g., AAAAAAA) should only produce a Lysine once every third nucleotide.
$\square$
(b) [1 pt $]$ How many flip-flops do you need to implement this FSM?

There are three states, so we need at least 2 bits, and therefore two flip-flops.

## Question 7: VHDL

(a) $[6 \mathrm{pts}]$ Write VHDL code to implement the circuit from question 5 (shown below).


You'll need to fill in the appropriate port declarations as well as the architecture in the code skeleton below.

```
library IEEE;
use IEEE.std_logic_1164.all;
entity parity is
    port (
        clk : in std_logic_vector;
        input : in std_logic_vector;
        output : out std_logic_vector
    );
end parity;
architecture synth of parity is
-- Don't need intermediate signals if we use VHDL 2008
begin
    process (clk) is
```

```
        begin
            if rising_edge(clk) then
                output <= input xor output;
            end if;
    end process;
end architecture;
```

(b) [2 pts] Circle the VHDL constructs below which can be used in synthesizeable logic.

- and
- case
- report
- unsigned
- wait
and, case, and unsigned can all be used in synthesizeable logic.


## Question 8: Bonus

[1 pt] Many 64-bit systems actually only use 48 bits in certain parts of the processor. Approximately what is the largest unsigned number that can be represented with 48 bits? Express your answer in decimal, using scientific notation if it's helpful.

$$
\begin{aligned}
& 2^{48}=2^{10} \times 2^{10} \times 2^{10} \times 2^{10} \times 2^{8} \\
& \approx 10^{3} \times 10^{3} \times 10^{3} \times 10^{3} \times 256 \\
& =256 \cdot 10^{12} \\
& =256 \text { trillion }
\end{aligned}
$$

Such a system could address 256 terabytes of memory, which should be enough for the next few years.

This page should be replaced by the VHDL reference sheet front

This page should be replaced by the VHDL reference sheet back

This page intentionally left blank.

