


CA-ECE-803 Advanced Digital Design Homework 1

Instructor: Dr. Fangning Hu

Submitted By: Neeha Hammad

1. (5%) 163_{10} in hexadecimal?

16	163
16	10 rem 3
	0 rem 10



Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Dec equiv.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15


10 corresponds to A in hex.

Therefore, $163_{10} = A3_{16} = 0xA3$

2. (5%) $0xFF$ in binary?

F F
 15_{10} 15_{10}



2	15
2	7 rem 1
2	3 rem 1
2	1 rem 1
	0 rem 1



$0xFF = 11111111_2$

3. (15%) a) For the following truth table, write the output F as a logic expression of the inputs A, B and C:

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

 $A' B'$
 $A B' C'$

Fill in the true terms of the truth table then OR them.

$$F = A'B' + AB'C'$$

b) Write the corresponding VHDL

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity q3b is
    port (
        A : in STD_LOGIC;
        B : in STD_LOGIC;
        C : in STD_LOGIC;
        F : out STD_LOGIC
    );

end q3b;

architecture 3qb of q3b is
    signal intermediate1 : std_logic;
    signal intermediate2 : std_logic;
begin
    intermediate1 <= not A and not B;
    intermediate2 <= A and not B and not C;
    F <= intermediate1 or intermediate2;
end 3qb;
```

4. (10%) What is the difference between C programming and VHDL?

C is modelled after a **sequential** process where operations are performed in sequential order, one at a time. Since an operation frequently depends on the result of an earlier operation, the order of execution cannot be altered at will. On the other hand, VHDL is used for implementing hardware circuits thus it allows **both sequential and concurrent** executions. It contains a set of sequential statements to be executed sequentially and the whole process is a concurrent statement

5. (10%) Write the logic function of the following K-map and simplify it.

A \ BC	00	01	11	10
0	1	1	0	0
1	1	1	0	0

BC \ A	0	1
00	1	1
01	1	1
11	0	0
10	0	0

$$F = A'B'C' + A'B'C + AB'C' + AB'C$$

$$= A'B'(C' + C) + AB'(C' + C)$$

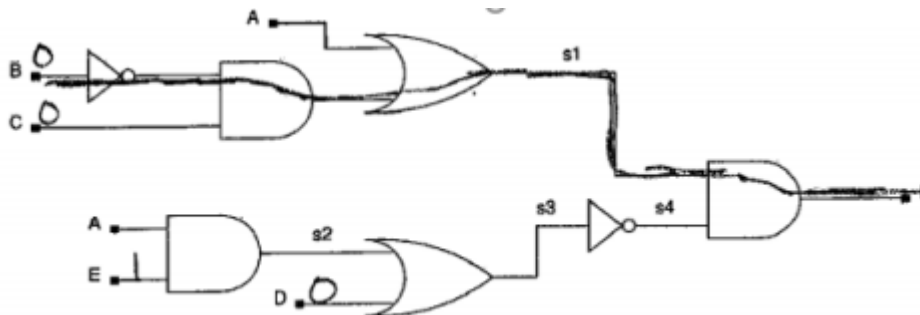
$$= A'B' + AB'$$

$$F = B' (A' + A)$$

1 (Compliment rule)

$$F = B'$$

6. (15%) Consider the following logic circuit:



1) write down the boolean expression of the circuit

$$s1 = A + \overline{BC}$$

$$s2 = AE$$

$$s3 = s2 + D = AE + D$$

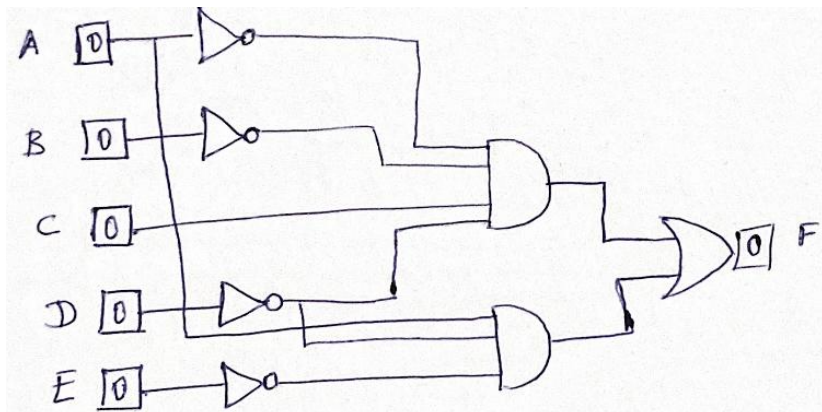
$$s4 = \overline{AE + D}$$

$$F = s1 + s4 = A + \overline{BC} + \overline{AE + D}$$

A	B	C	D	E	$F = (A + \overline{BC}) * (\overline{AE} + D)$
0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	1	0	0
0	0	0	1	1	0
0	0	1	0	0	1
0	0	1	0	1	1
0	0	1	1	0	0
0	0	1	1	1	0
0	1	0	0	0	0
0	1	0	0	1	0
0	1	0	1	0	0
0	1	0	1	1	0
0	1	1	0	0	0
0	1	1	0	1	0
0	1	1	1	0	0
0	1	1	1	1	0
1	0	0	0	0	1
1	0	0	0	1	0
1	0	0	1	0	0
1	0	0	1	1	0
1	0	1	0	0	1
1	0	1	0	1	0
1	0	1	1	0	0
1	0	1	1	1	0
1	1	0	0	0	1
1	1	0	0	1	0
1	1	0	1	0	0
1	1	0	1	1	0
1	1	1	0	0	1
1	1	1	0	1	0
1	1	1	1	0	0
1	1	1	1	1	0

2) modify the circuit to be only 2 levels of logic, the first level is AND, the second level is OR

$$F = \overline{A}BC\overline{D} + A\overline{D}E$$

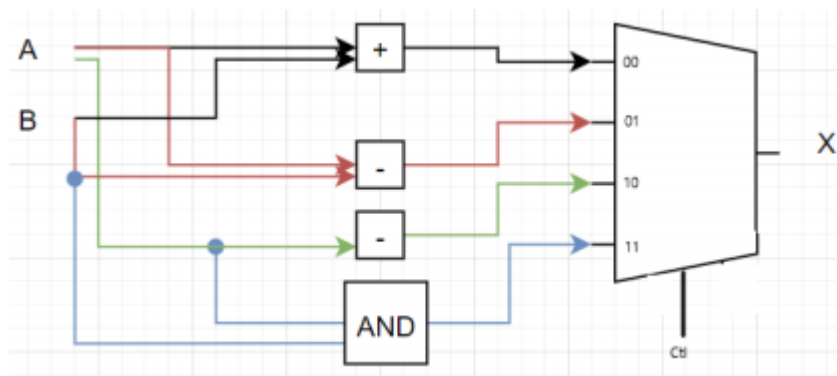


7. (10%) Consider the VHDL code fragment
- ```

signal ctl: std_logic_vector (1 downto 0);
with ctl select
x <= a+b when "00"
 a-b when "01"
 -a when "10"
 (a and b);

```

(a) Draw the conceptual diagram of what this code represents.



(b) Rewrite the code using a conditional assignment, and draw the corresponding conceptual diagram.

```

signal ctl: std_logic_vector (1 downto 0);

```

```

architecture COND of BRANCH is
begin

```

```

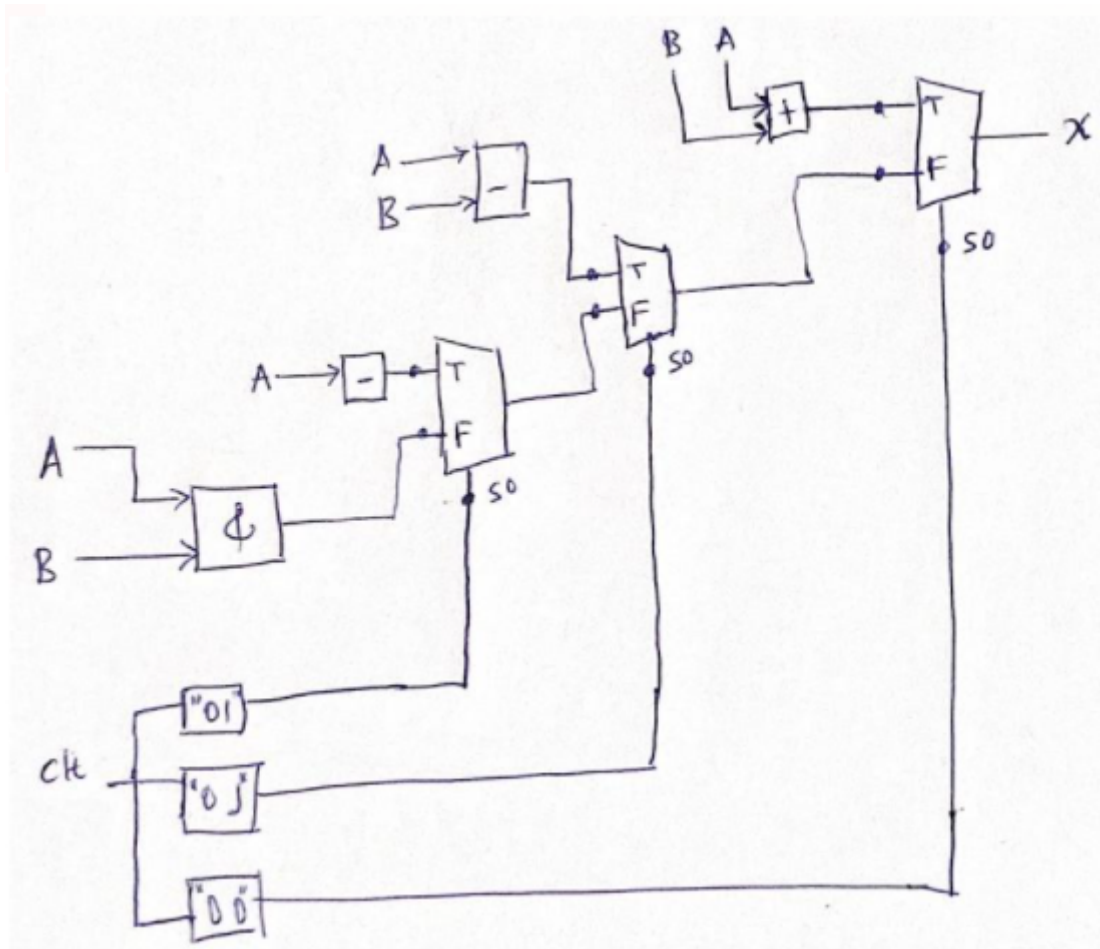
x <= a+b when ctl = "00" else
 a-b when ctl = "01" else
 -a when ctl = "10" else
 (a and b);

```

```

End COND

```



8. (5%) Consider the code

```
process(a, b)
```

```
begin
```

```
 if (a = b) then
```

```
 eq <= '1';
```

```
 else
```

```
 leq <= '1';
```

```
 end if;
```

```
end process;
```

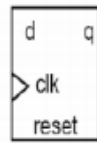
After  $a \leq 6$ ;  $b \leq 6$ ;  $a \leq 10$ ;  $b \leq 6$ ; what are the results for eq and leq?

When a and b are 6, the first condition passes therefore eq becomes 1.

When  $a = 10$  and  $b = 6$ , then the first condition fails and the next passes therefore leq becomes 1.

Both eq & leq are equal to 1

9. (10%) Use the VHDL to describe a register



| reset | clk | q* |
|-------|-----|----|
| 1     | -   | 0  |
| 0     | 0   | q  |
| 0     | 1   | q  |
| 0     | f   | d  |

```

library ieee;
use ieee.std_logic_1164.all;
entity dffr is
 port(
 clk: in std_logic;
 reset: in std_logic;
 d: in std_logic;
 q: out std_logic
);
end dffr;

architecture arch of dffr is
begin
 process(clk,reset)
 begin
 if (reset='1') then
 q <= '0';
 elsif (clk'event and clk='1') then
 q <= d;
 end if;
 end process;
end arch;

```

10. (15%) Similar to what you did in the lab for the 7-segment decoder, draw the block diagram of a circuit that takes two 4-bit numbers A and B (input using switches 0-3 and 4-7, respectively) and outputs the following values on the digits in hexadecimal:

| Digit           | Displayed value           |
|-----------------|---------------------------|
| 3 (left digit)  | A+B (note: discard carry) |
| 2               | A-B (note: discard carry) |
| 1               | -A (lower 4 bits)         |
| 0 (right digit) | A and B (lower 4 bits)    |

You may assume that you already have logic blocks that implement the operations +, -. You can also assume you have a block that does the 4-bit binary to 7-segment decoder function.

