CS 151
Final

Name : ___________________  ___________________
      (Last Name)      (First Name)

Student ID : __________________

Signature : __________________

Instructions:

1. Please verify that your paper contains 19 pages including this cover.
2. Write down your Student-Id on the top of each page of this final.
3. This exam is closed book. No notes or other materials are permitted.
4. Total credits of this final are 90 + 20 EXTRA CREDIT.
5. To receive credit you must show your work clearly.
6. No re-grades will be entertained if you use a pencil.
7. Calculators are NOT allowed.

<table>
<thead>
<tr>
<th>PART I</th>
<th>PART II</th>
<th>PART III</th>
<th>PART III</th>
<th>PART III</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>Q1</td>
<td>Q2</td>
<td>Q3</td>
<td>Q3 (Extra Credit)</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>30</td>
<td>40</td>
<td>20</td>
</tr>
</tbody>
</table>

Total Credit
PART I - Multiple choice questions

Note:
- DO NOT GUESS! A correct answer will receive 2 points and an incorrect answer will receive -1 point.
- No partial credits will be awarded in Part I.

Q1: [Standard forms]

Select the canonical sum-of-products representation of the following function:
\[ f(x,y,z) = xy' + y(x+z) \]

(a) \( f(x,y,z) = xy' + xy + yz \)
(b) \( f(x,y,z) = x + yz \)
(c) \( f(x,y,z) = xy'z + xy'z' + xyz + x'yz + xyz' \)
(d) \( f(x,y,z) = (x + y + z).(x + y + z').(x + y' + z) \)
(e) \( f(x,y,z) = x'y'z' + x'y'z + x'yz' \)

Q2: [Prime Implicants]

Select the definition of an essential prime implicant:
(a) A sub-cube that is not contained within any other sub-cube
(b) A sub-cube that includes a 1-minterm that is not included in any other sub-cube
(c) The set of prime implicants which have common sub-cubes
(d) All of the above
(e) Both (a) and (b)

Q3: [Karnaugh Map]

Select the derivation with minimum number of literals for the function represented in the following K-map:

```
 X_1X_0  Y_1Y_0
 00 | 01 | 11 | 10
-----|----|----|----
 01 | 1 |
 11 | 1 | 1 | 1
 10 | 1 | 1 |
```

(a) \( X_1Y_1' + X_0Y_1'Y_0' + X_1X_0Y_0' \)
(b) \( X_1Y_1 + X_0Y_1Y_0 + X_1X_0Y_0 \)
(c) \( X_1'Y_1 + X_0'Y_1Y_0 + X_1'X_0'Y_0 \)
(d) \( X_1'X_0'Y_0' + X_1X_0Y_1'Y_0' + X_1X_0Y_1Y_0' + X_1X_0'Y_1Y_0' + X_1X_0'Y_1Y_0 + X_1X_0Y_1Y_0' \)
(e) All of the above

Q4: [Setup and Hold]

For the positive-edge triggered D flip-flop shown below, the clock input is logic low (0) from \( t = 0 \text{ns} \) to \( t = 10 \text{ns} \), makes a transition from logic low to logic high (1) at \( t = 10 \text{ns} \), and stays high until \( t = 20 \text{ns} \).

However, for the input at D to be propagated to Q correctly at this clock transition, the flip-flop designer specifies that the input has to be available before \( t = 9 \text{ns} \) and has to remain unchanged until \( t = 12 \text{ns} \).

The setup and hold times for this D flip-flop are:

\[
\begin{align*}
\text{(a)} & \quad T_{\text{setup}} = 9 \text{ns}, \quad T_{\text{hold}} = 12 \text{ns} \\
\check{(b)} & \quad T_{\text{setup}} = 1 \text{ns}, \quad T_{\text{hold}} = 2 \text{ns} \\
\text{(c)} & \quad T_{\text{setup}} = 2 \text{ns}, \quad T_{\text{hold}} = 1 \text{ns} \\
\text{(d)} & \quad T_{\text{setup}} = 9 \text{ns}, \quad T_{\text{hold}} = 8 \text{ns} \\
\text{(e)} & \quad T_{\text{setup}} = 8 \text{ns}, \quad T_{\text{hold}} = 9 \text{ns}
\end{align*}
\]

Q5: [latches and flip flops]

The circuit below is the NAND-gates implementation of:

\[
\begin{align*}
\text{(a)} & \quad \text{S-R latch.} \\
\check{(b)} & \quad \text{D latch.} \\
\text{(c)} & \quad \text{D flip-flop.} \\
\text{(d)} & \quad \text{Gated S-R latch.} \\
\text{(e)} & \quad \text{Gated D latch}
\end{align*}
\]
PART II – Short Design Questions

Q1: [Hierarchical Multiplexer Design] [8 points]

Design a 16-to-1 multiplexer using ONLY 4-to-1 multiplexers.

This is the block for a 16-to-1 MUX:
**PART III – Design Questions**

Q1: [FSM Design] [30 points]
Design a synchronous 2-bit counter which has an input M. When M = 0 the counter should count in Binary numbers and when M = 1 it should count in Gray sequence.

NOTE: During operation of the counter, if Mode bit M changes, the counter should start from the next number in the other sequence. For instance, if the current value on counter is 01 during Binary counting, and Mode Bit M changes from 0 to 1 then the counter should switch to the Gray sequence and the next value on the counter should be 11.

(HINT: 2-bit Gray sequence is 00, 01, 11, 10)

For this counter:

a. Capture the FSM using symbolic states. (15 points)
b. Draw the architecture template for the FSM, clearly show the inputs, outputs, next state, etc (3 points)

![ FSM Architecture Diagram ]

- 3 bits
- Current state
- Next state

c. Encode the states (use a simple Binary encoding) (2 points)

\[ S_0 = 000, \quad S_1 = 001, \quad S_2 = 010, \quad S_3 = 011 \]
\[ S_4 = 100, \quad S_5 = 101 \]
**d. Draw the state table using the state encoding from part c. (5 points)**

<table>
<thead>
<tr>
<th>Current State</th>
<th>Next State M=0</th>
<th>Next State M=1</th>
<th>Output X1</th>
<th>Output X0</th>
</tr>
</thead>
<tbody>
<tr>
<td>P₂ P₁ P₀</td>
<td>n₂ n₀</td>
<td>n₂ n₀</td>
<td>X₁</td>
<td>X₀</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
e. Derive the equations for the FSM’s combinational logic (no need to draw the gates, just write the equations) (5 points)

\[ X_1 = P'_2 P_1 + P_2 P'_1 = P'_2 P_1 P_0 + P'_2 P_1 P'_0 + P_2 P'_1 P_0 + P_2 P'_1 P'_0 \]

\[ X_0 = P'_2 P'_1 P_0 + P'_2 P_1 P_0 + P_2 P'_1 P'_0 \]

\[ n_2 = P'_2 P'_1 P'_0 M + P'_2 P_1 P'_0 M + P_2 P'_1 P'_0 M \]

\[ n_1 = P'_2 P'_1 P'_0 M' + P'_2 P_1 P'_0 M' + P_2 P'_1 P'_0 M' \]

\[ n_0 = P'_2 P'_1 P'_0 M' + P'_2 P_1 P'_0 M' + P_2 P'_1 P'_0 M' + P'_2 P'_1 P'_0 M + P'_2 P_1 P'_0 M + P_2 P'_1 P'_0 M \]
Q2: RTL design [40 points]

We want to design the logic for dispensing cash from an ATM machine in this question. There are two phases for doing a transaction.

**Phase 1: Authentication**

When you insert your card into this ATM machine it asks you for your password. It has a database of card numbers and the corresponding passwords. The card numbers and passwords are stored in a memory with 256 words. Every word is 16 bits and the most significant 8 bits represent the card number and the least significant 8 bits represent the password. When the card is inserted if the card number is not found in the data base the machine “swallows” the card.

Otherwise it asks for the 8 bit password. If the password is not correct it asks you for the password two more times (i.e., a total of three times). If an incorrect password it entered three times, the machine swallows the card and does not return it to the user.

**Phase 2: Dispensing Cash**

If the card and password are authenticated (i.e., the card number is found in the database and the password is correct) the machine lets you enter the amount of money that you need. You are allowed maximum of $200 for each transaction. (Assume you input a value up to $200 and don’t need to check for this condition.) The machine dispenses $1, $5 and $20 bills. Based on the amount entered it calculates the number of $20, $5 and $1 bills that it should dispense. Furthermore the machine dispenses the least number of bills possible based on the amount requested. (For example a request for $26 returns one $20, one $5 and one $1 bill.)

In this question for simplicity assume Phase 1 is successful (i.e., the user’s card and password have been authenticated) and the ATM starts Phase 2 (dispensing cash) when an ENTER signal is hit. **You will do the RTL design for Phase 2.**
Student ID: _______________

Assume you can use only the following types of components in your design:

Adder, Subtractor, MUX, Register, comparator

a. In the FSM and Data path template below, connect the signals to the appropriate boxes to indicate the high-level FSM and Datapath organization. (You will design the logic for each later) (5 points)

**NOTE:** REQ_AMT is the amount of money that is requested.
b. Develop pseudo code for the Phase 2 of the ATM (dispensing cash) using the inputs and outputs defined in Part a.

(5 points)

Input: REQ_AMT, ENTER
Output: DONE, ONES, FIVES, TWENTIES

begin
Reg_Amt, Reg, OneBills = 0, FiveBills = 0, TwentyBills = 0;
Reg_Amt = REQ_AMT;

while (! ENTER) ;

while (Reg_Amt > 20) {
    Reg_Amt = Reg_Amt - 20 ;
    TwentyBills ++;
}

while (Reg_Amt > 5) {
    Reg_Amt = Reg_Amt - 5 ;
    FiveBills ++;
}

while (Reg_Amt > 1) {
    Reg_Amt = Reg_Amt - 1 ;
    OneBills ++;
}

TWENTIES = TwentyBills;
FIVES = FiveBills;
ONES = OneBills;
DONE = 1;
End
c. Design the high-level state machine. (10 points)
d. Design the data path. (10 points)
e. Connect the data path to the controller (i.e., show the control signals between the FSM and data path.) *(5 points)*
f. Derive the controller's FSM (5 points)
Q3. EXTRA CREDIT [20 points]

Design the logic for the authentication circuit (Phase 1) in previous question.

Assume you can use only the following types of components in your design:

Adder/Subtractor, MUX, Register, comparator

a. In the FSM and Datapath template below, connect the signals to the appropriate boxes to indicate the high-level FSM and Datapath organization. (You will design the logic for each later) (2 points)

![Diagram of ATM Machine with inputs and outputs](image_url)
b. Design the high-level state machine. (8 points)
c. Design the data path. (8 points)
d. Interconnect the two designs in this question (Phase 1: authentication) with the previous question (Phase 2: dispensing cash) to complete the design of the complete ATM machine. Show the primary inputs and outputs to the system and the interconnection signals between the two systems. (2 points)

Primary inputs: Card Number, Password, ENTER 1, ENTER 2, REQ-AMT

Primary outputs: DONE, ONES, FIVES, TWENTIES