# Introduction to Computing: Homework 4

**Fall 2025**

**Dr. Ramesh Yerraballi**

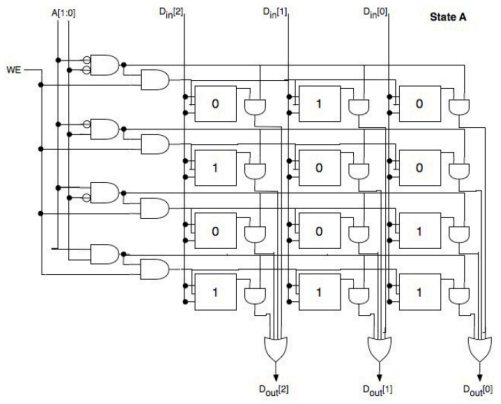
**The University of Texas at Austin**

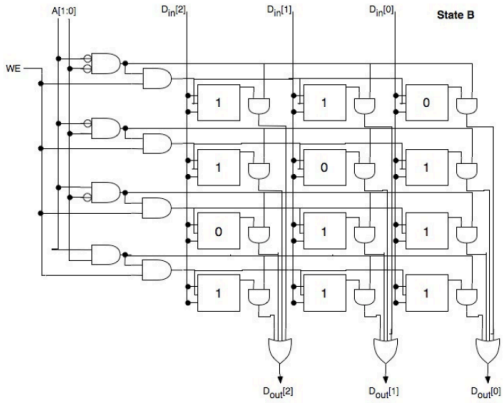
Due: Tuesday, 9/30, 11:59pm

Instructions:
You may discuss the problem set solutions with your fellow classmates but the write up must be your own. Please use the TAs and the Instructor for help before you seek out a friend or classmate. Show your work. You may handwrite or type-write your answers.

1. (10 pts) Rework the NAND R-S Latch by replacing the NAND gates with NOR gates. What are the outputs for the four possible input scenarios?

2. (10 pts) After five clock cycles memory goes from state A to state B (shown below). The table below shows values of the 2-bit address(A[1:0]), 1-bit Write-Enable and 3-bit $D_{in}$ signals. For each cycle, fill in the missing bits (represented by dashes)




| | A[1:0] | WE | $D_{in}$[2:0] |
|---|---|---|---|
| Cycle 1 | 1 ___ | 1 | 0 ___ ___ |
| Cycle 2 | 1 1 | ___ | 1 1 1 |
| Cycle 3 | ___ 0 | 1 | 0 0 0 |
| Cycle 4 | 0 0 | 1 | ___ ___ ___ |
| Cycle 5 | ___ ___ | ___ | ___ ___ ___ |

3. (15 pts) A task in yaos (yet another operating system) can be seen as being in one of three states: {**ready, running, blocked**}. The event that causes a change in a task's state is, whether or not the Scheduler picks it. A process starts in the ready state and stays in that state until it is picked, at which time it moves to the running state. In the running state if the scheduler picks it, it moves to a blocked state otherwise it goes back to the ready state. It remains in the blocked state until it is picked, at which point it returns to the ready state. The actions performed (outputs) in each state are as follows: In ready state neither the cpu nor devices are used by the task. In running state, the task uses the cpu but not the devices. In the blocked state it uses devices but not the cpu.
    a. Give the State Transition Graph for the FSM that captures a task's behavior in yaos.
    b. Give the Truth Table for the FSM.

c. Give the Output Table for the FSM

4. (10 pts) Shown below are the State Transition Table and Output Table for an FSM:

| CS | | Input | | NS | |
|---|---|---|---|---|---|
| S1 | S0 | I1 | I0 | N1 | N0 |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 |

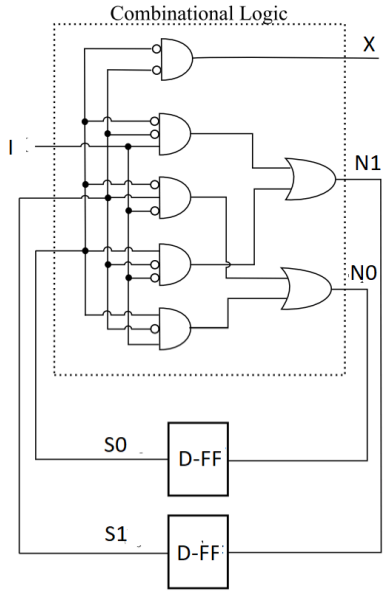
| CS | | Output |
|---|---|---|
| S1 | S0 | Y |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

a. Give the State Transition Graph for the FSM.

b. What size decoder is needed to implement a PLA-based circuit for the FSM ?

c. Write a simplified boolean expression for the Output (Y)

5. (20 Points) Shown below is the logic circuit implementation of a Finite State Machine.




a. Give the State Transition Table and the Output Table that this circuit is designed from?

b. Give the State Transition Graph for the FSM.

6. (20 points) Given the following truth table where the State transition table and the Output table have been merged.

| Current State | Input | Next State | Output |
|---|---|---|---|
| 0 0 | 0 0 | 0 0 | 1 1 |
| 0 0 | 0 1 | 0 1 | 1 1 |
| 0 0 | 1 0 | 1 0 | 1 1 |
| 0 0 | 1 1 | 0 0 | 1 1 |
| 0 1 | 0 0 | 0 0 | 0 1 |
| 0 1 | 0 1 | 0 1 | 0 1 |
| 0 1 | 1 0 | 1 0 | 0 1 |
| 0 1 | 1 1 | 0 1 | 0 1 |
| 1 0 | 0 0 | 0 0 | 1 0 |
| 1 0 | 0 1 | 0 1 | 1 0 |
| 1 0 | 1 0 | 1 0 | 1 0 |
| 1 0 | 1 1 | 0 1 | 1 0 |

a. Show what the Output Table is.
b. Construct the State Transition Graph for the FSM

7. (15 points) Give the FSM State Diagram/Graph for a detector that detects a pattern in successive tosses of a coin. If an even number of tails follow an even number of heads then detector light turns ON (1). In all other cases the detector stays OFF (0). For example:

```
Input:    THHTTTTHTHHTHHHHTTH
Output: 00000101000000000010
```

Notes: 1. The first `0` in the output is the output produced in the initial state before any input has been received/processed; 2. *Zero is not even*.