**Project 1: Car Parking Controller**

A Parking plaza needs to automate its car parking functionality. The parking plaza has four floors i.e. Basement, Ground Floor, First Floor and Second floor each having a capacity 10 cars. When a car comes in, the controller should tell the user about status of each floor i.e. “*Space Available*” or “*Full*”. User can choose a floor to park his car. If a floor is full, the controller will lock the entrance door of the floor and car cannot enter that floor. You have to implement the Car Parking Controller. Also your system should display the cars parked on each floor. You can assume that the cars will come in and get out of the floor in an order.

***Input Signals:***

CarIn

Floor to park the car

CarOut

***Output Signals:***

Status of Floors

Door Locks

Parked Cars

**Project 2: Circular Linked List**

Design a linked list of 3 bit numbers with capacity for 3x4 bits. It should be able to insert input with 2-bit input for position and 3 bits for actual data. Every position should know whether there is data available or not. This should be set on insert. It should be able to rotate data right & left. It should also be able to swap data of 2 nodes. In that case there should be two 2 bits for source & destination.

Your circuit has 2 inputs A, B to read the command

|  |  |
| --- | --- |
| **Command** | **A,B** |
| Insert | 00 |
| Rotate right | 01 |
| Rotate left | 10 |
| Swap | 11 |

A Seven segment display is attached to each node.

|  |  |  |
| --- | --- | --- |
| ***Input*** |  |  |
| Clock Pulse | CP |  |
| Insert Position | P0,P1 | |
| Insert Data | I0…I2 | |
| Source Position | S0,S1 | |
| Destination Position | D0,D1 | |
| Command | A,B | |
| ***Output*** |  |  |
| 4 Seven segment display showing the contents of each node | | |
| Data Already present | IF | (insert failure) |

**Note:**

If node is empty then Seven segment display should display E

**Project 3: 4-bit Processor**

Design a 4-bit processor which consists of 4 data registers each of 4 bits and an instruction register (IR) of 7 bits. The first 3 bits of the instruction tells which operation is to be performed, the next 2 bits signifies the first register and the last two bits signifies the second register.

|  |  |  |
| --- | --- | --- |
| I6-I4 | I3-I2 | I1 – I0 |
| Operation Code | 4-bit register operand 1(R1) | 4-bit register operand 2 (R2) |

The following operations are performed by the processor.

|  |  |  |
| --- | --- | --- |
| Operation Code | Operation Performed | Description |
| 000 | R1 = A | Load the contents of input A in to the register operand 1. |
| 001 | R1 = R2 | Move the contents of register operand 2 in to register operand 1. |
| 010 | R1 = R1 + R2 | Add the contents of register operand 1 and register operand 2 and load in register operand 1. |
| 011 | R1 = R1 - R2 | Subtract the contents of register operand 2 from register operand 1 and load in register operand 1. |
| 100 | R1 = R1 \* R2 | Multiply the contents of register operand 1 and register operand 2 and load in register operand 1 and 2. (As the result is of 8 bits) |
| 101 | R1 = R1/2i | Divide the register contents of register operand 1 with 2i (i is an input) and load the result in register operand 1. |
| 110 | R1 = R1.R2 | Logical And the contents of register operand 1 and register operand 2 and load in register operand 1. |
| 111 | R1 = R1 + R2 | Logical OR the contents of register operand 1 and register operand 2 and load in register operand 1. |

**Inputs:** Clock Pulse (CP), 7-bits Instruction, A, i

**Output:** Contents of each register

**Project 4: Snakes and Ladders**

It’s a two player game. You have 0-35 cells, with each cell having a unique number. Each player roles a dice and output can only be between 1 to 6. The player moves number of cells ahead according to face value of dice. The first one to reach 35 will win. You need to fix the positions of snakes and ladders on the cells. The game also has to display the turn of each player.

***Input***

Clock Pulse CP

Start Dice Roll 1 to start and 0 to stop

***Output***

Position of each player on board

Winner Player WP

Note:

There should be at least 3 snakes and 2 ladders.