floydCycle_hardware

Proteus Simulation and Implementation of Floyd's cycle finding algorithm in a blockchain

Jishnu Das
Hobby project
IIT-BHU
29 June 2023

Outline

1. Theory
2. Cyclic Right Shifting
3. Problem Statement
4. Implementation Algorithm
5. Back of napkin sketch
6. Block Diagram
7. Sub-blocks
8. Entire Circuit Diagram
9. Links for Video Demonstration

Theory

- **What** is Blockchain **- Decentralized** (access to all) database of all transactions/ data.

- **How** is data stored
  - **Linked list** fashion. Generally the data is encrypted to ensure that no one can access it without authorization.
  - The list contains 2 elements - **data** and next data's **address**

- In our case, data is converted into binary and encrypted in a right cyclic shifting of the next node address by 3 bits

Cyclic Right Shifting

Unhashed Array:
[8,3,20,4,10,9,78,85,23,5,12,11,15,17,31,25,65,30,22,19,44,66,35,9]

4 = 0000100 -> hash -> 10000000 = 128
9 = 00001001 -> hash -> 00100001 = 33
12 = 00001100 -> hash -> 10000001 = 129 and similarly for the rest....

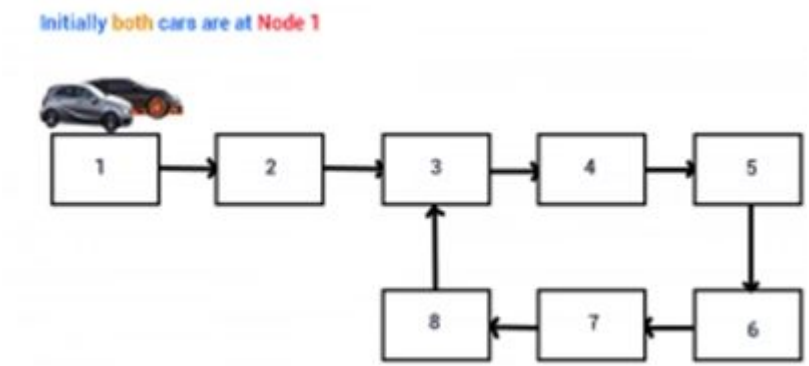
Corresponding Hashed array in the ROM:
[8,3,20,128,10,33,78,85,23,5,129,11,15,34,31,25,65,30,194,19,44,66,35,33]

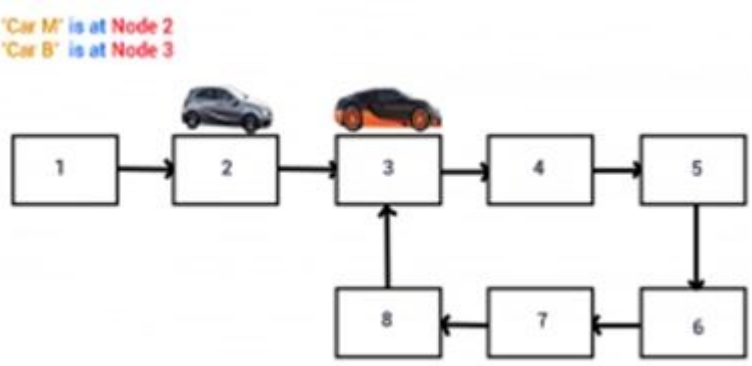
Problem Statement

- **Data** will be in **unhashed 8 bits**. The **address** to the next data is **hashed 5 bits**
- We given this hashed array in the ROM.
- Probably someone has tampered with the data and there is a **cycle** in the linked list, which would collapse the entire linked list.
- Find if there is **any such cycle formed** and if present, find the decrypted **address of the head.**

Implementation Algorithm

- Floyd Cycle Detection Algorithm
  - Initialize 2 pointer - Fast and Slow pointer
  - Move the fast pointer by two steps while slow pointer by 1 step



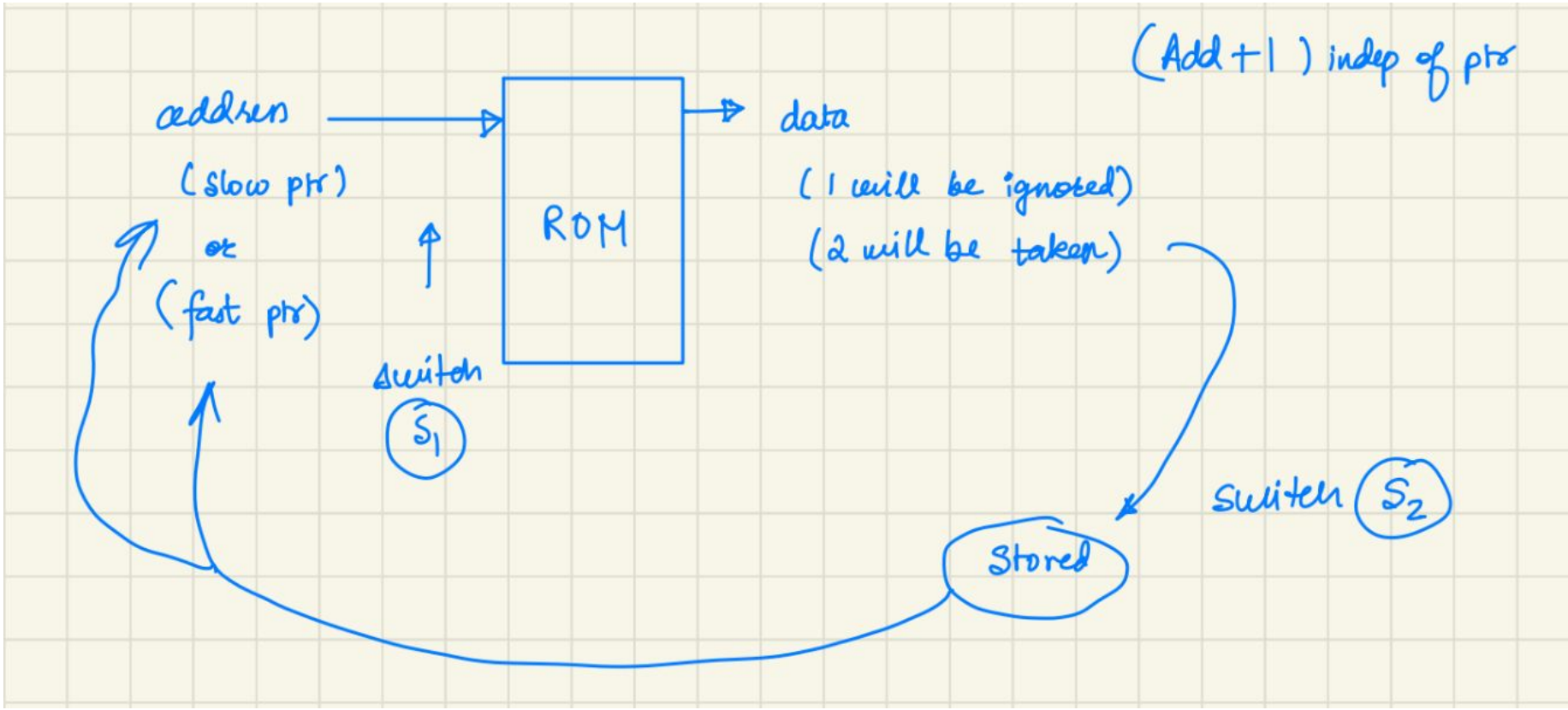


Implementation Algorithm

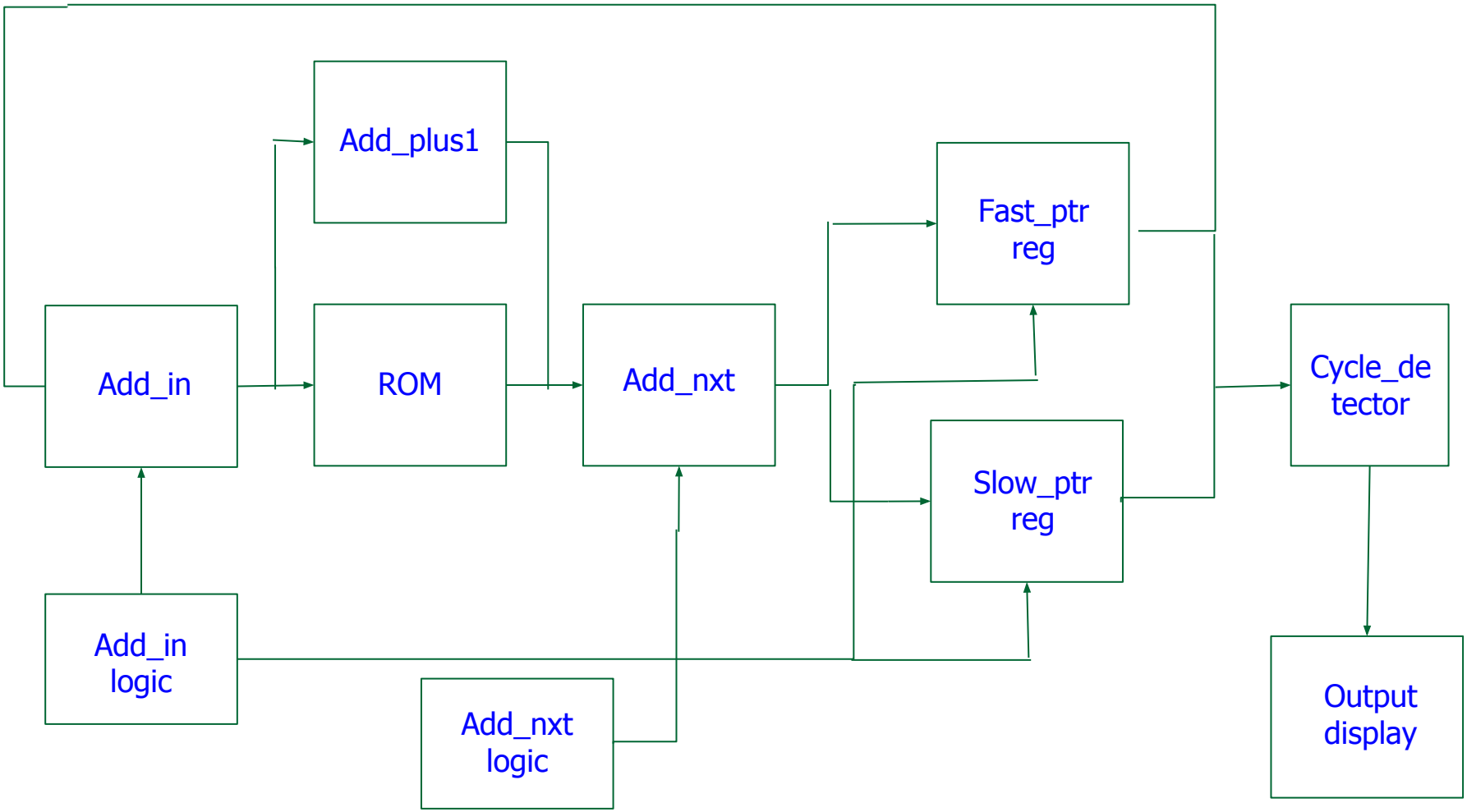
- Algorithm
  - **Initialize** them (fast ptr and slow ptr) to head of the linked list
  - In the implementation the **fast pointer** goes by **one step in every clock cycle** and the **slow pointer** goes **one step in every two clock cycles**.

  - Either fast pointer meets the end. The end pointer's address is 255 or 256. In this case **no loop is detected.**
  - If **fast** pointer's **address** is **less** than **slow** pointer's **address**, then cycle is detected.
  - The **address of the fast pointer** will determine the **address of head**.

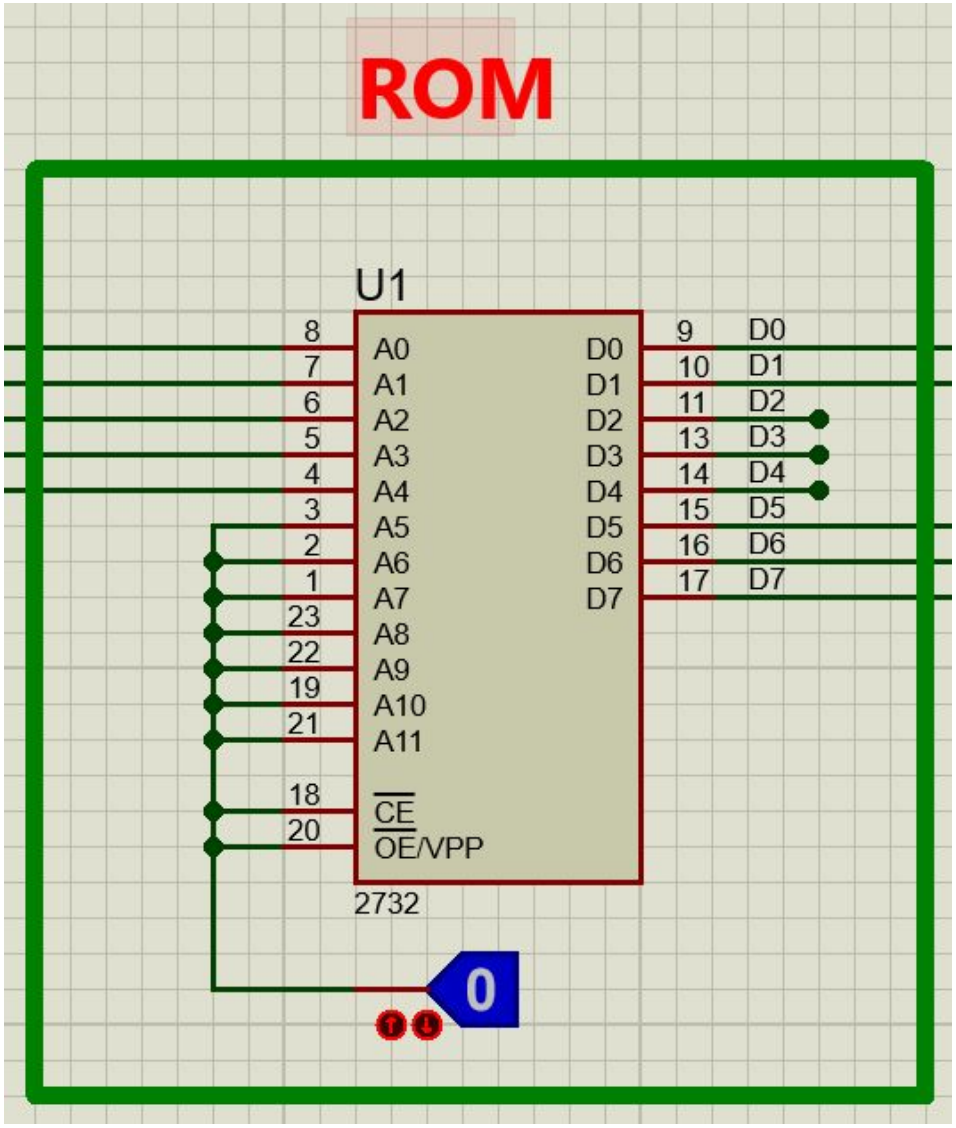
Back of napkin block diagram



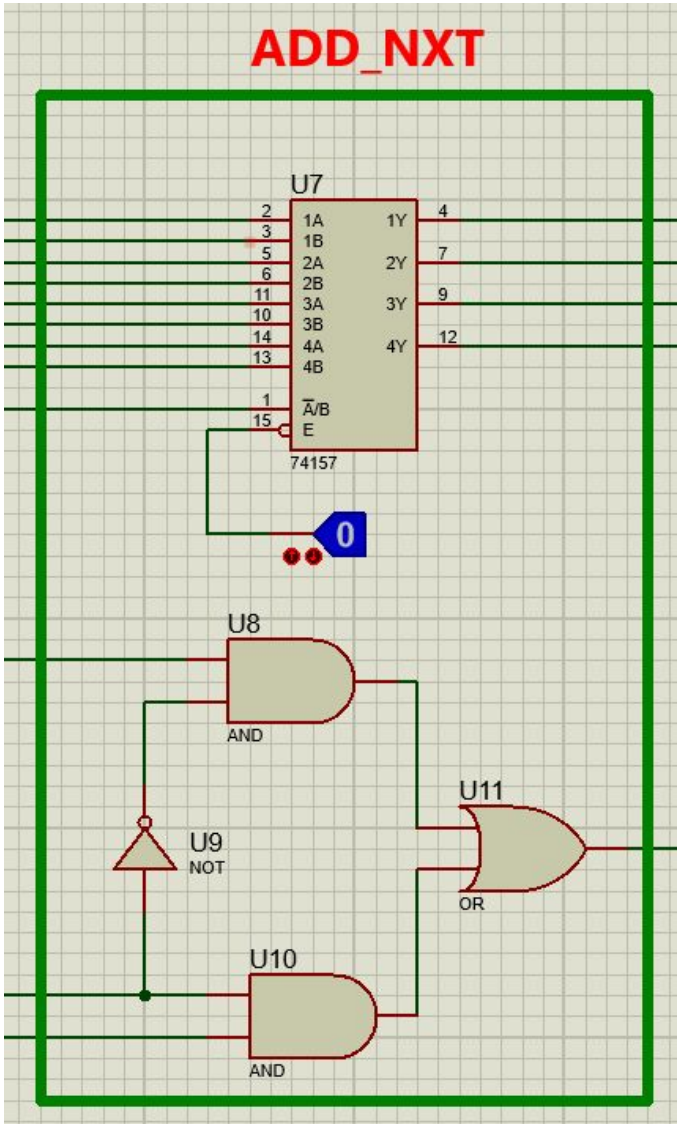
Block diagram



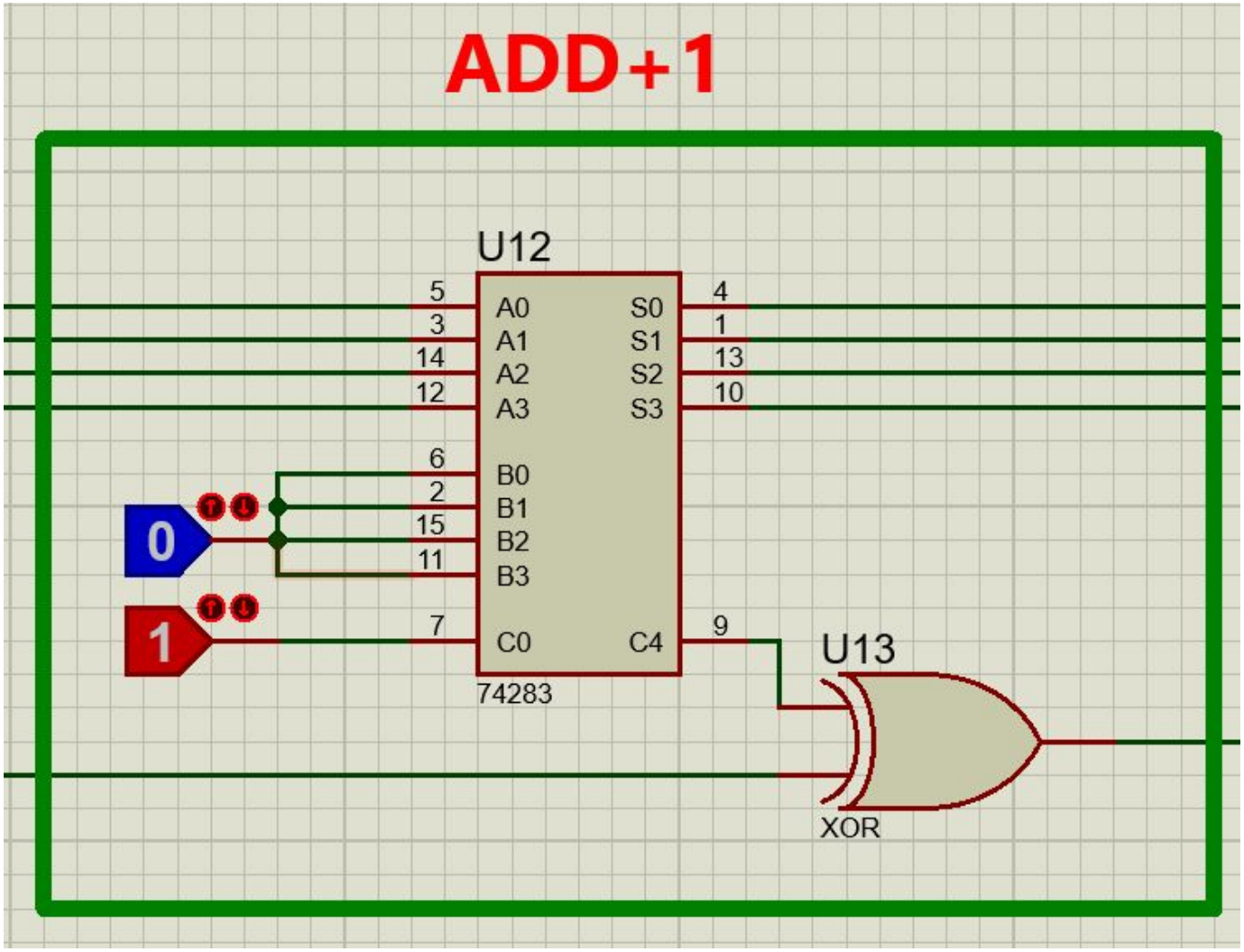
ROM block


ROM
U1
8
7
6
5
4
3
2
1
23
22
19
21
18
20
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
CE
OE/VPP
D0
D1
D2
D3
D4
D5
D6
D7
9
10
11
13
14
15
16
17
D0
D1
D2
D3
D4
D5
D6
D7
2732
0

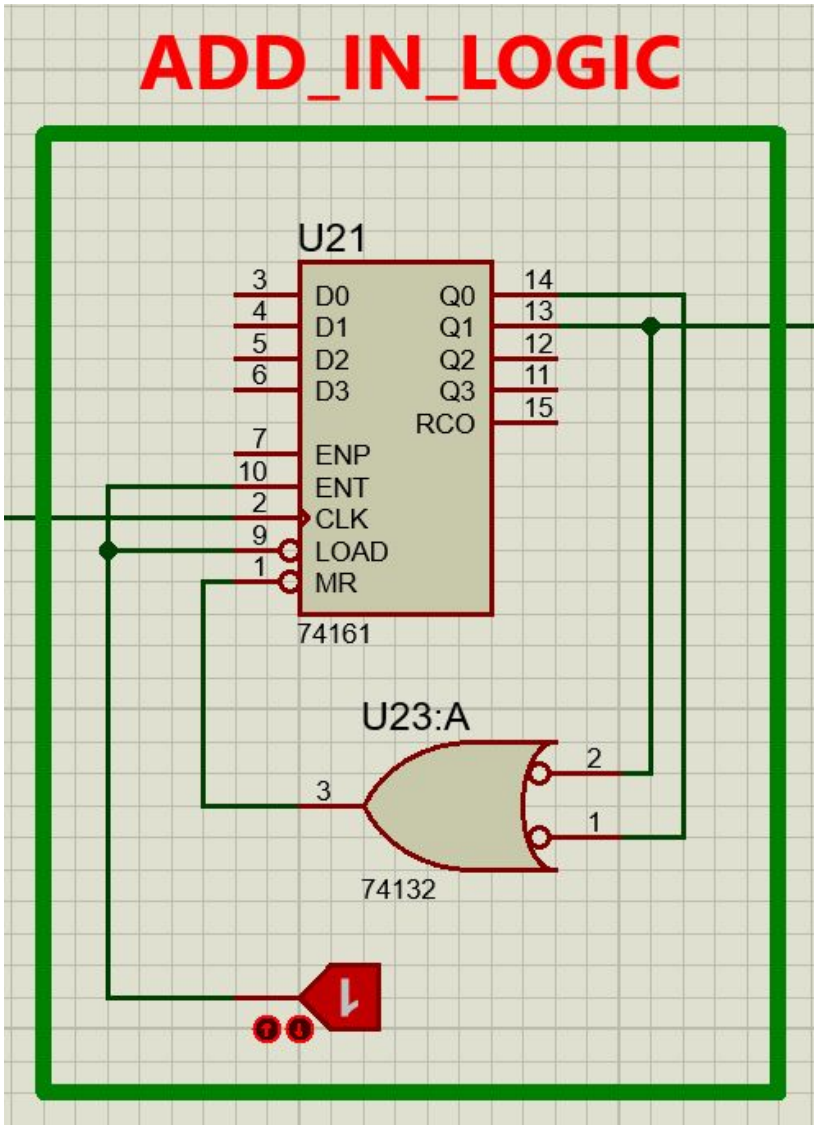
Add_nxt block



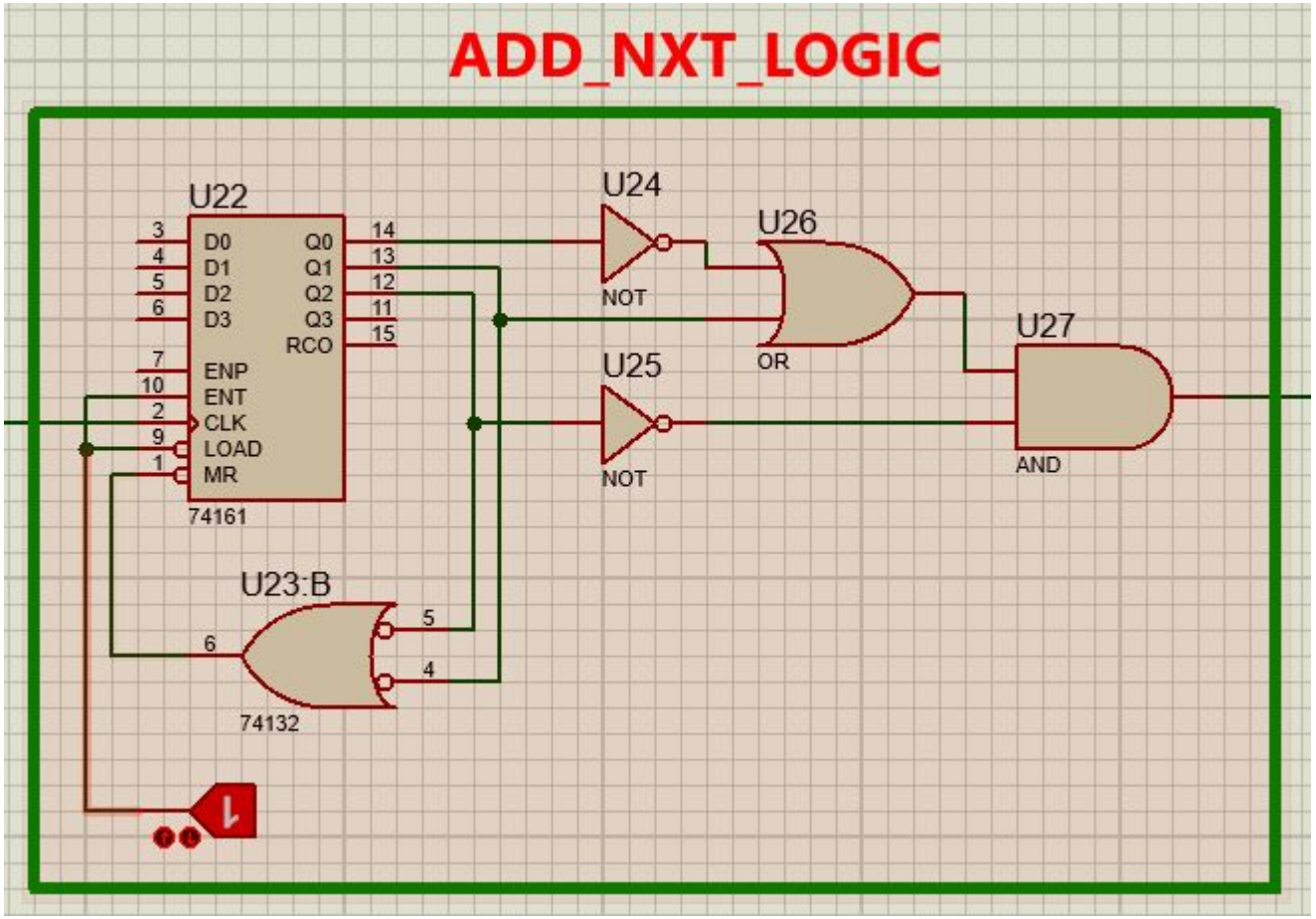
Add_plus1 block



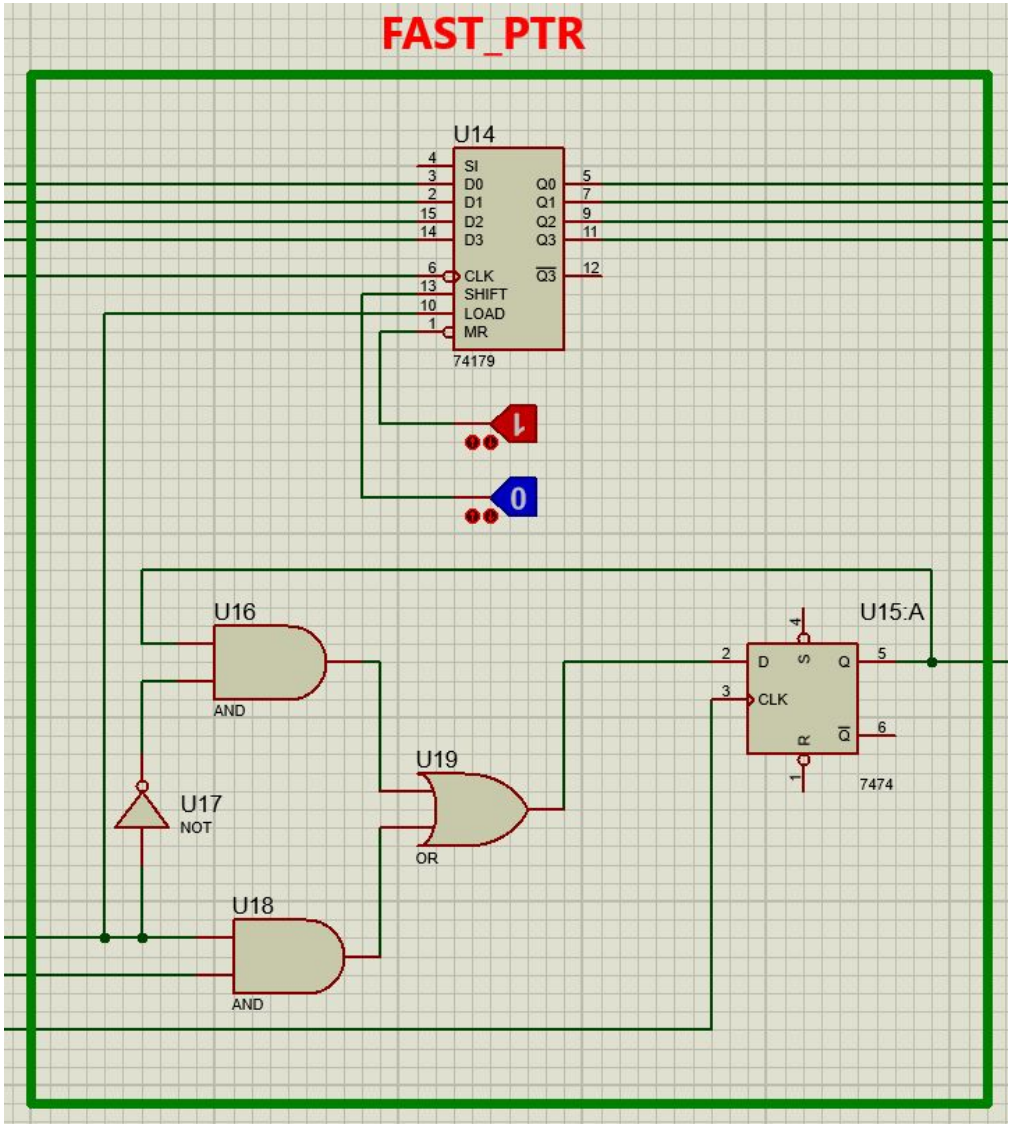
Add_in logic


ADD_IN_LOGIC
U21
3
4
5
6
7
10
2
9
1
D0
D1
D2
D3
ENP
ENT
CLK
LOAD
MR
Q0
Q1
Q2
Q3
RCO
14
13
12
11
15
74161
U23:A
3
2
1
74132

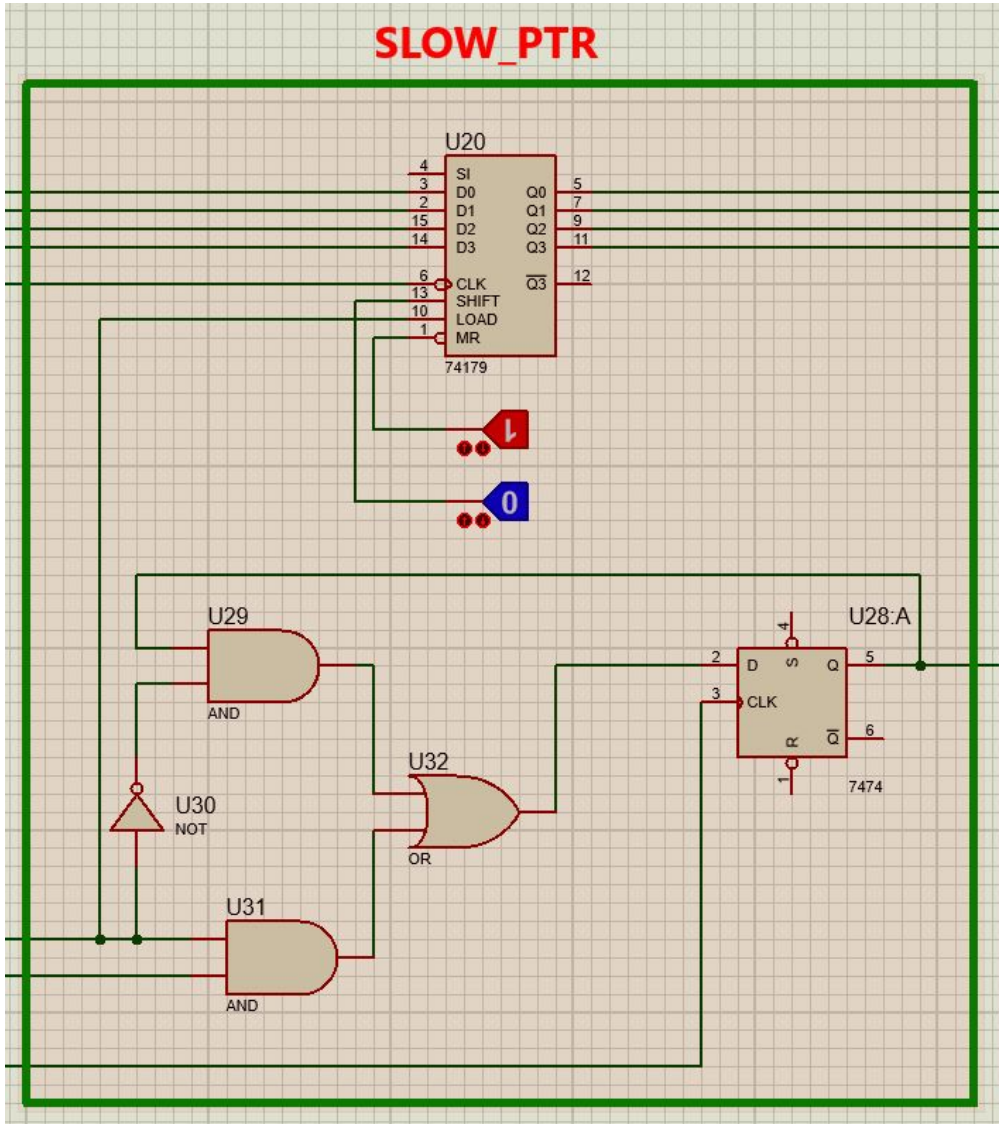
Add_nxt logic



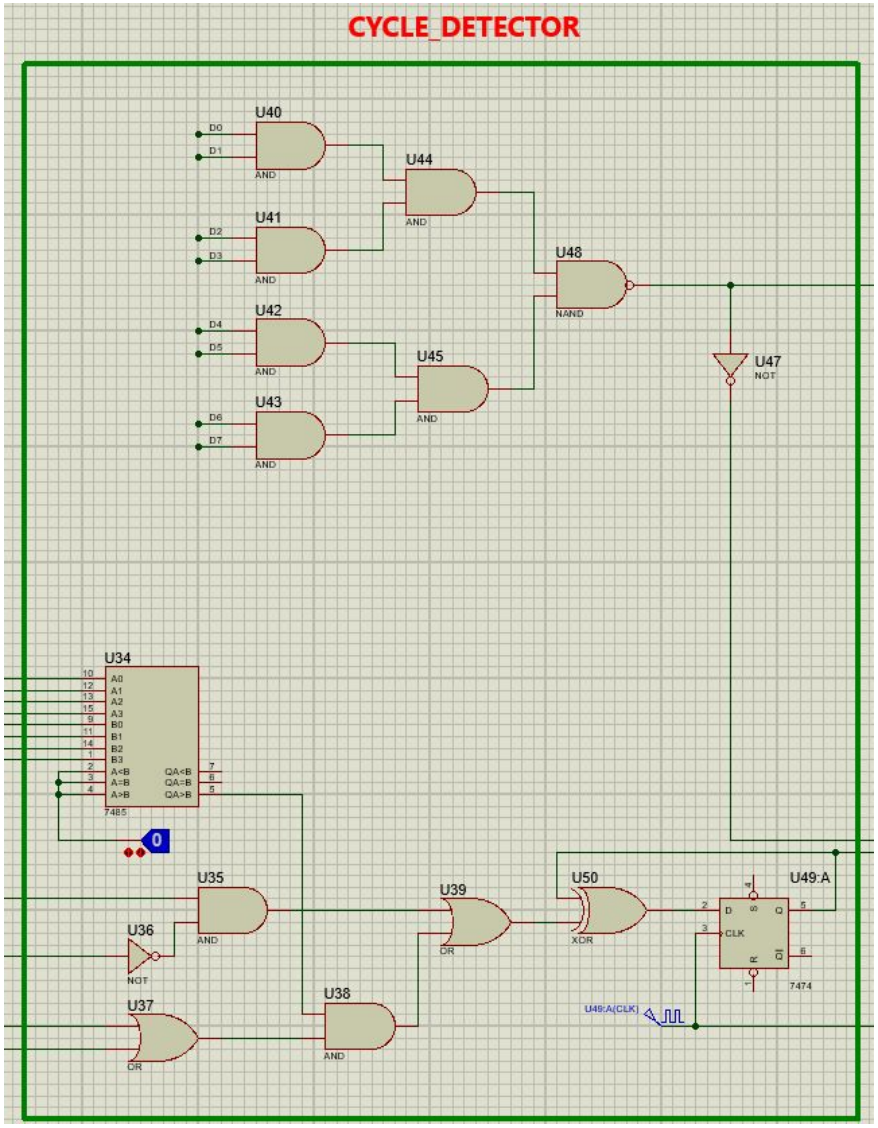
Fast_ptr reg


FAST_PTR
U14
SI
D0
D1
D2
D3
CLK
SHIFT
LOAD
MR
Q0
Q1
Q2
Q3
Q3
74179
1
0
U16
AND
U17
NOT
U18
AND
U19
OR
U15:A
D
CLK
S
R
Q
Q
7474

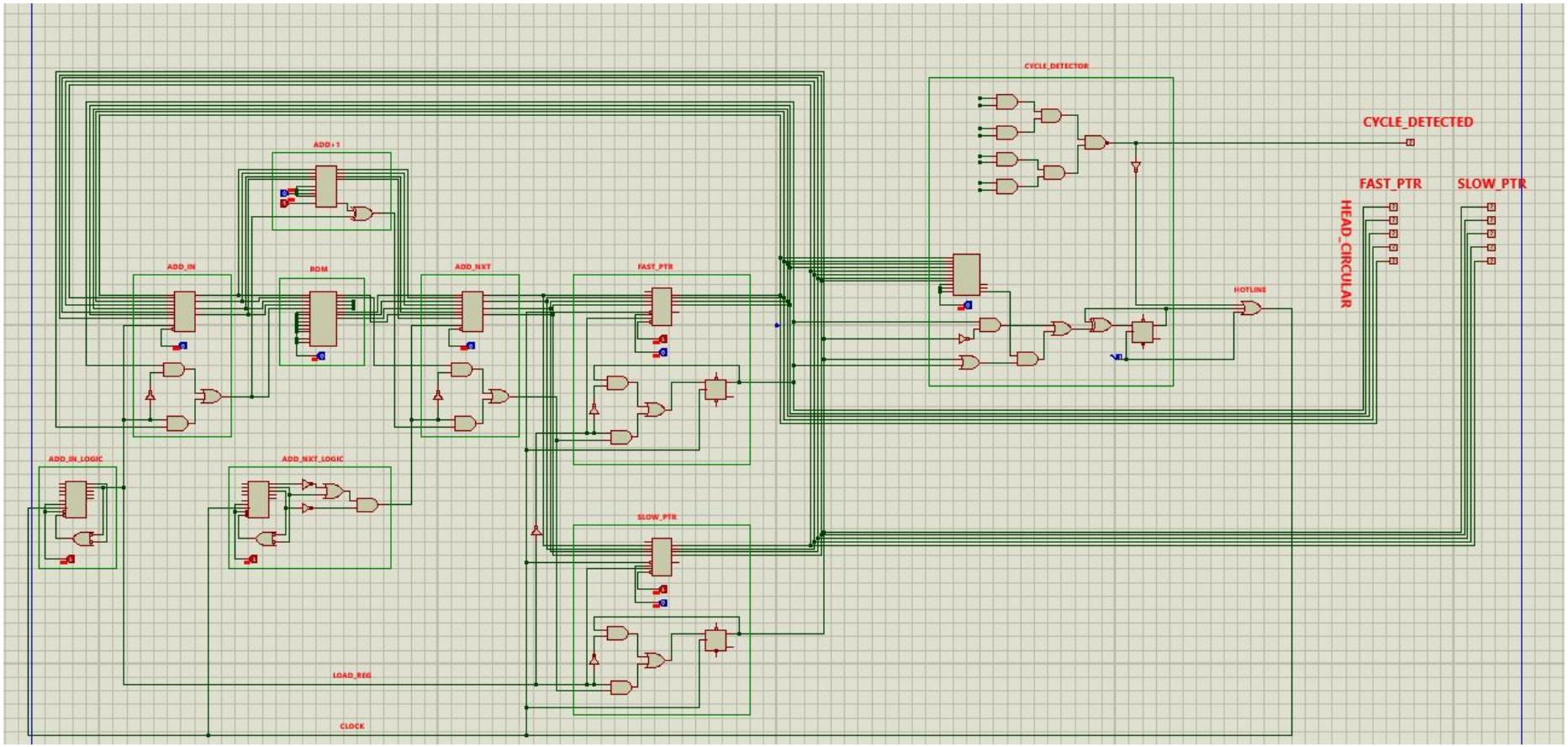
Slow_ptr reg


SLOW_PTR
U20
SI
D0
D1
D2
D3
CLK
SHIFT
LOAD
MR
Q0
Q1
Q2
Q3
Q3
74179
1
0
U29
AND
U30
NOT
U31
AND
U32
OR
U28:A
D
CLK
S
R
Q
Q
7474

Cycle_detector logic



Entire Circuit Diagram



Links for Video Demonstration