# Design of Digital Circuits
# Sequential Circuits

...

| |
|---|
| Michiliia Gureva |
| Sandeep Kumar |
| Stephen Nnamani |
| Vulcu Horia - Rares |
| Yew Heng Ngoh |

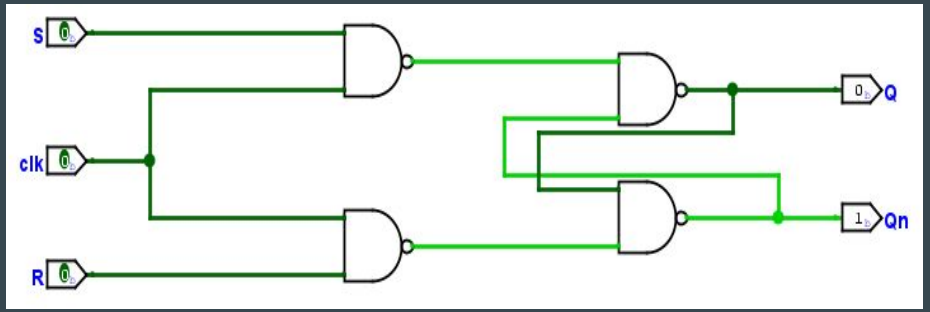
# Overview

1. D-Flip-Flop
2. JK-Flip-Flop
3. Sequence Detector
4. Shift Register
5. Serial Adder

# 1. D-Flip-Flop

Synchronous SR- Latch to D-flip-flop:

- The SR-Latch is in Active High, which means irrespective of the clock cycle if Q was once High it will remain will remain high until the reset R input goes high.

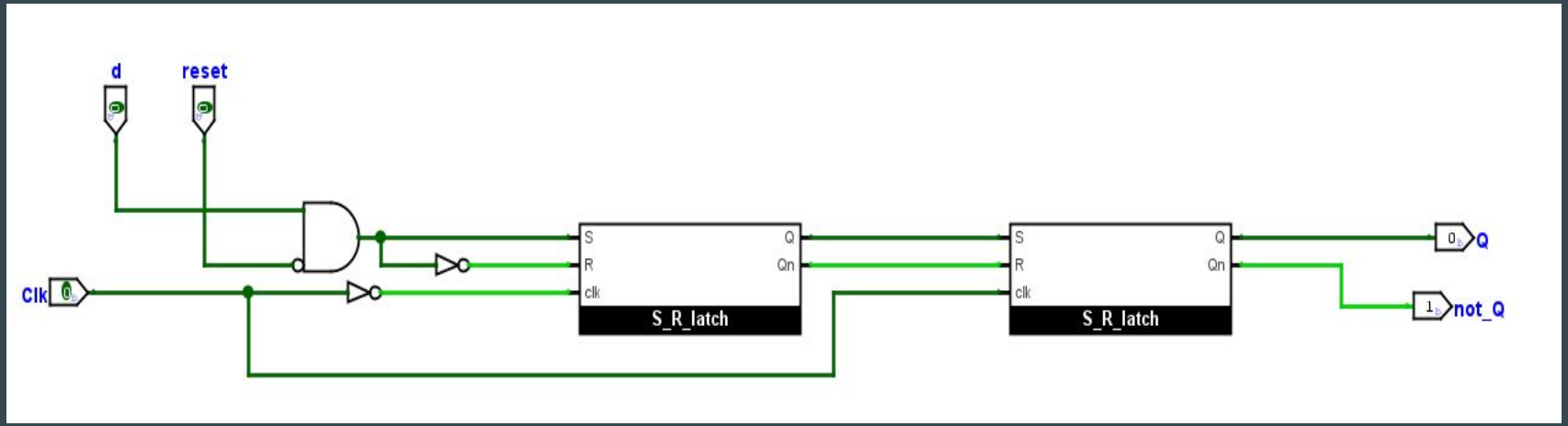



8 of 8 rows shown

| S | Clk | R | Q | Qn | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | | | |
| 0 | 0 | 1 | | | |
| 0 | 1 | 0 | | | |
| 0 | 1 | 1 | 0 | 1 | Reset |
| 1 | 0 | 0 | | | |
| 1 | 0 | 1 | | | |
| 1 | 1 | 0 | 1 | 0 | Set |
| 1 | 1 | 1 | 1 | 1 | Invalid |

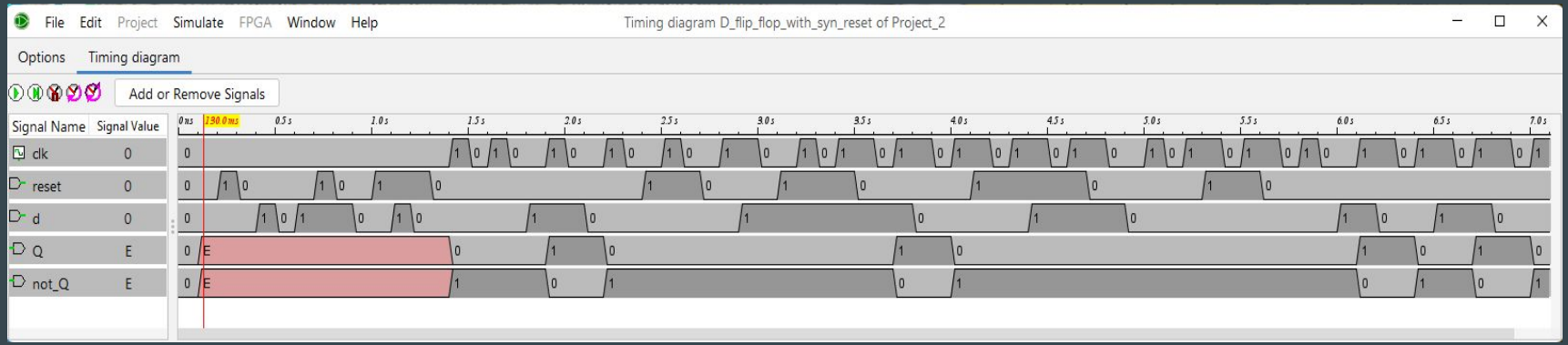
# Positive Edge Triggered D-flip-flop with Synchronous reset:

- A D flip-flop with synchronous reset, the reset signal is synchronized with the clock signal.
- The flip-flop only resets when both the clock signal and the reset are high at the same time.

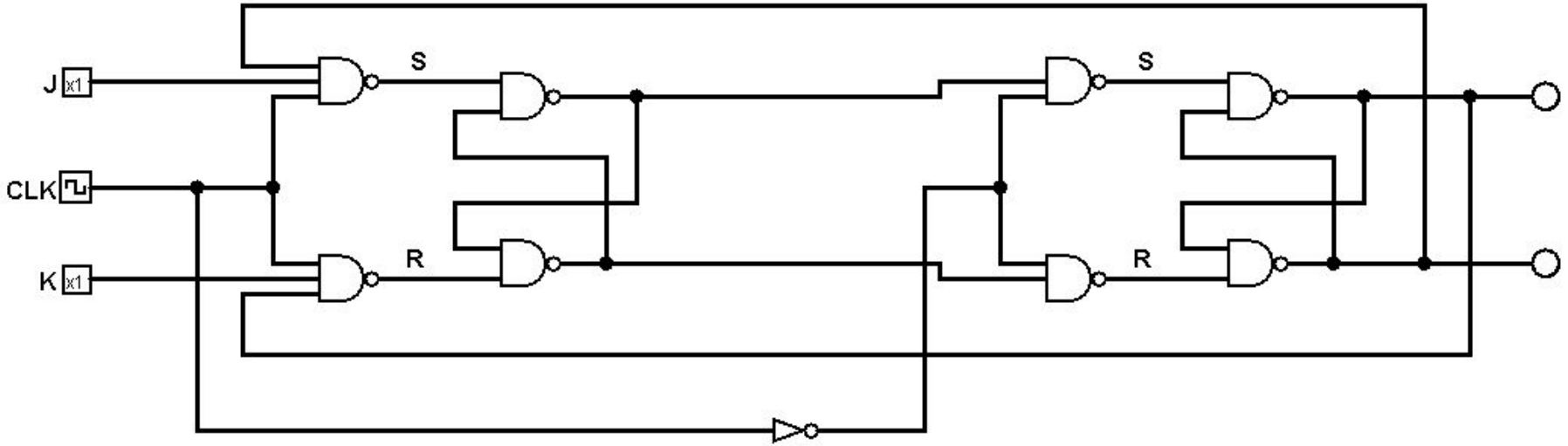


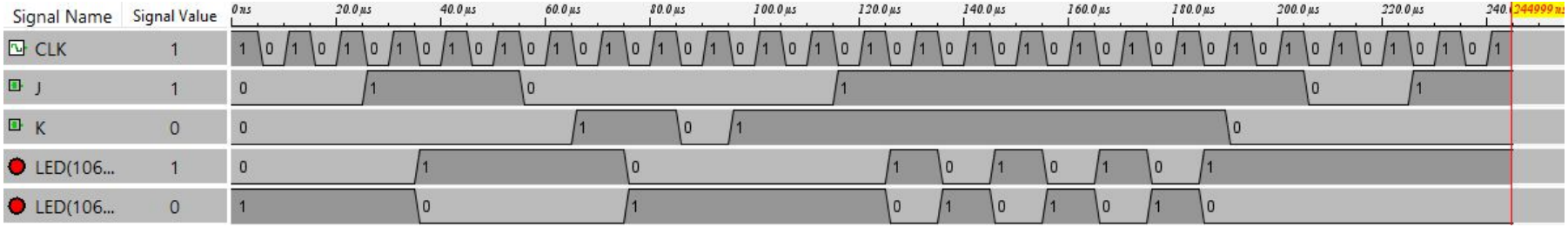
# Chronogram for D-Flip-Flop:

The initial Error is due to the recursive Q and not_Q which were connected to the internal Signal-loop.I overcame this by just putting toggling the clock my low to high.



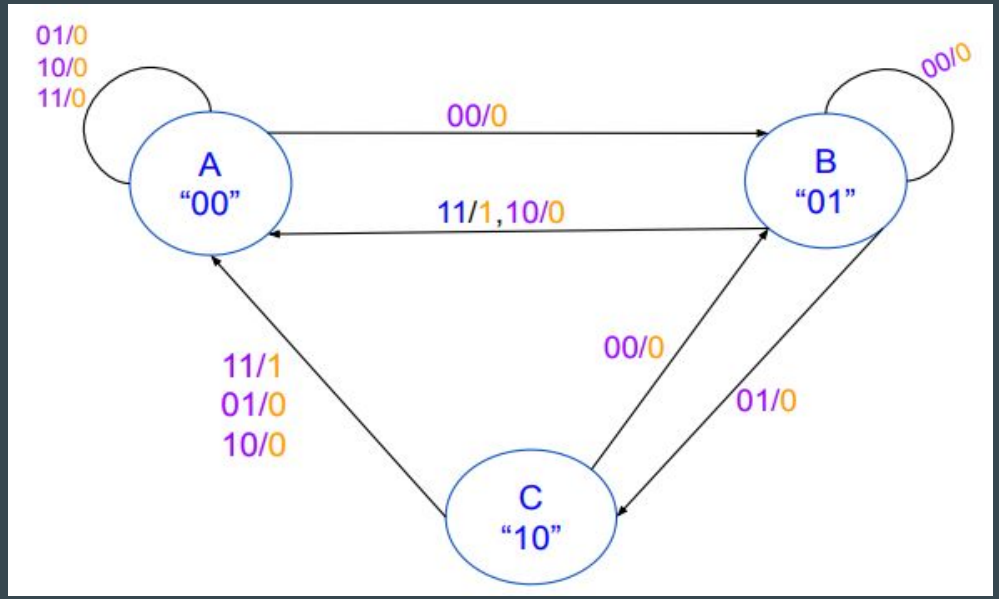
# 2. JK-Flip-Flop



# 3. Sequence Detector

1.Finite state Machine state Diagram:

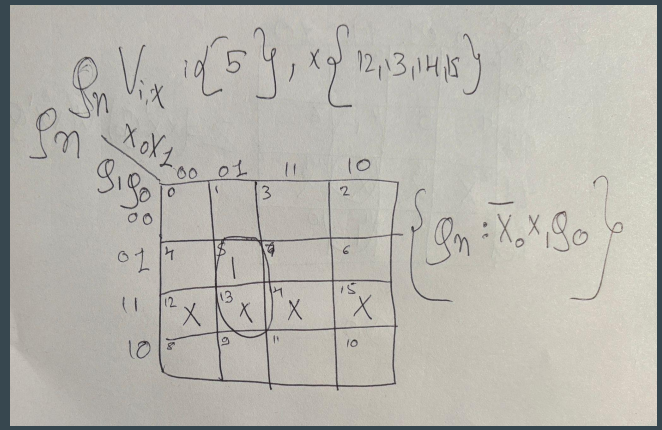


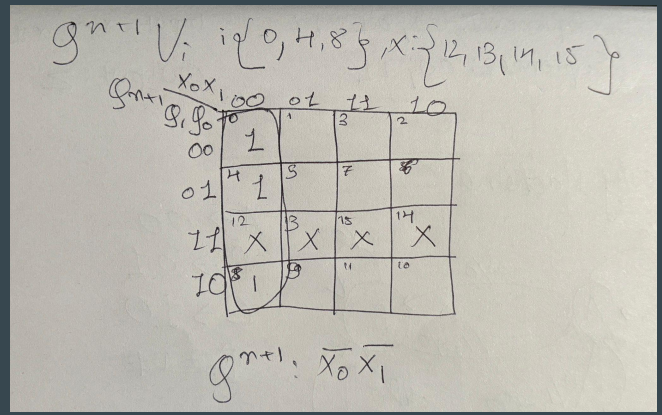

Mealy Machine
Sequence 1: 00;01;11
Sequence 2: 00;11

2.State encoding:
A = 00
B = 01
C = 10

# 3.State transition table & optimized input equations:

| Index | Input | | Current state | | | Next State | | | Z |
|---|---|---|---|---|---|---|---|---|---|
| i | Xo | X1 | State | Q1 | Q0 | State | Qn | Qn+1 | Output |
| 0 | 0 | 0 | A | 0 | 0 | B | 0 | 1 | 0 |
| 1 | 0 | 1 | A | 0 | 0 | A | 0 | 0 | 0 |
| 2 | 1 | 0 | A | 0 | 0 | A | 0 | 0 | 0 |
| 3 | 1 | 1 | A | 0 | 0 | A | 0 | 0 | 0 |
| 4 | 0 | 0 | B | 0 | 1 | B | 0 | 1 | 0 |
| 5 | 0 | 1 | B | 0 | 1 | C | 1 | 0 | 0 |
| 6 | 1 | 0 | B | 0 | 1 | A | 0 | 0 | 0 |
| 7 | 1 | 1 | B | 0 | 1 | A | 0 | 0 | 1 |
| 8 | 0 | 0 | C | 1 | 0 | B | 0 | 1 | 0 |
| 9 | 0 | 1 | C | 1 | 0 | A | 0 | 0 | 0 |
| 10 | 1 | 0 | C | 1 | 0 | A | 0 | 0 | 0 |
| 11 | 1 | 1 | C | 1 | 0 | A | 0 | 0 | 1 |
| 12 | 0 | 0 | D | 1 | 1 | X | X | X | X |
| 13 | 0 | 1 | D | 1 | 1 | X | X | X | X |
| 14 | 1 | 0 | D | 1 | 1 | X | X | X | X |
| 15 | 1 | 1 | D | 1 | 1 | X | X | X | X |



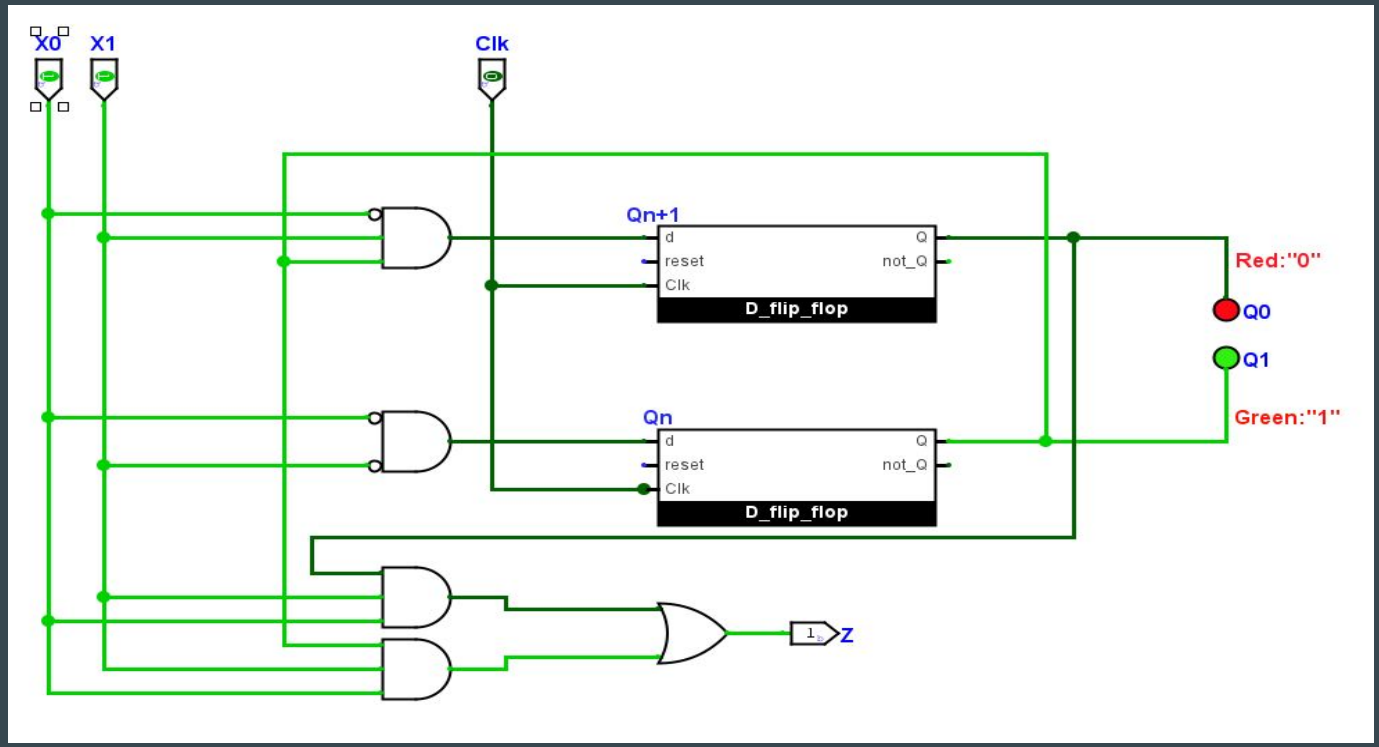
$Z \forall i: i \in \{7, 11\}, X: \{12, 13, 14, 15\}$

$Z$

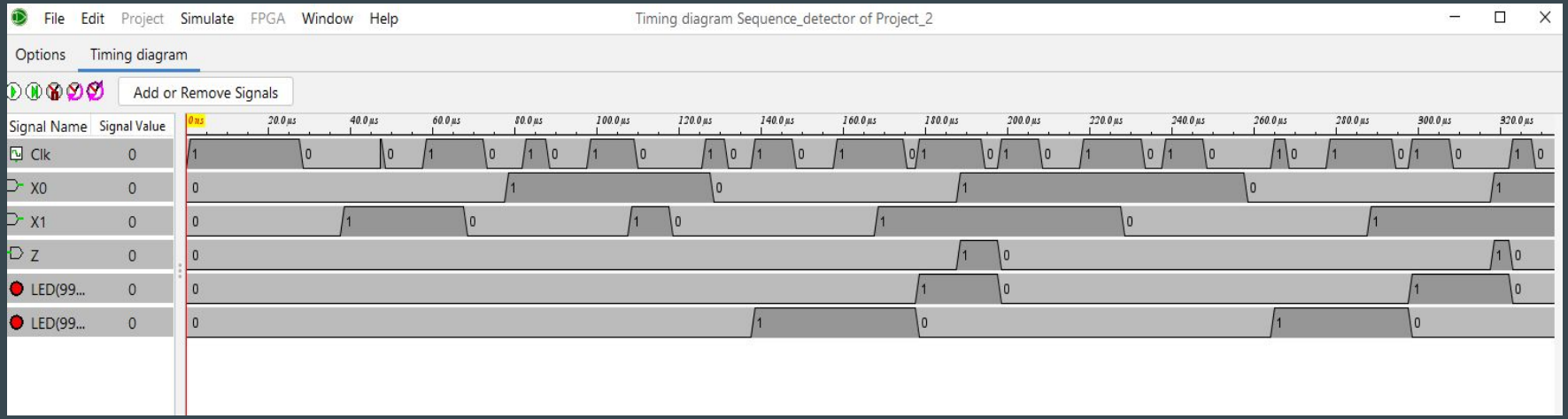
| $q_1q_0$ \ $x^0x^1$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 0 | 1 | 3 | 2 |
| 01 | 4 | 5 | 7 (1) | 6 |
| 11 | 12 X | 13 X | 15 (X) | 14 X |
| 10 | 8 | 9 | 11 (1) | 10 |

$Z = (x^0x^1q_0) \vee (x^0x^1q_1)$

# 4. circuit in Logisim:

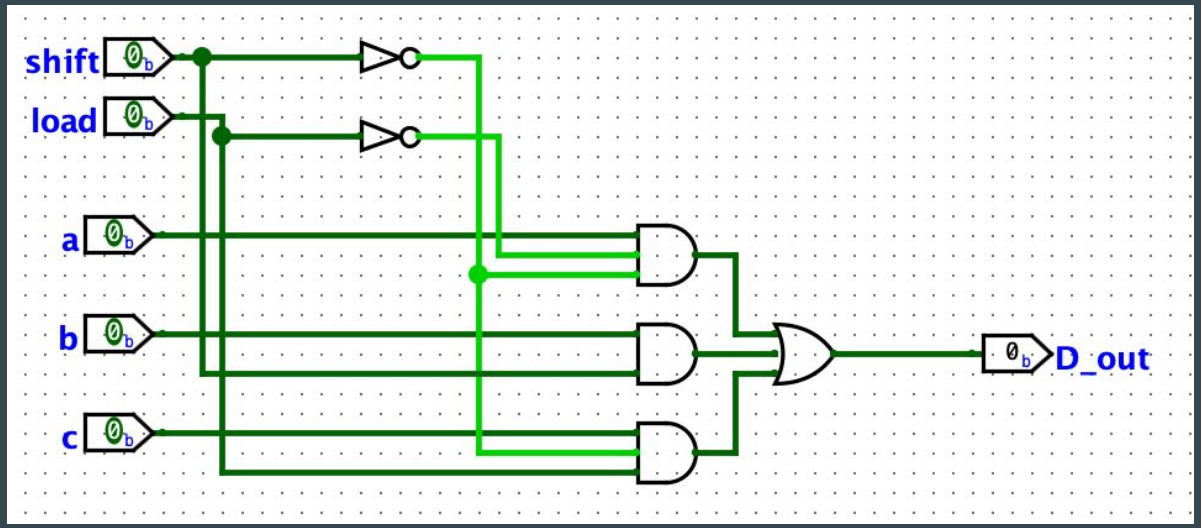


# 5.Chronogram



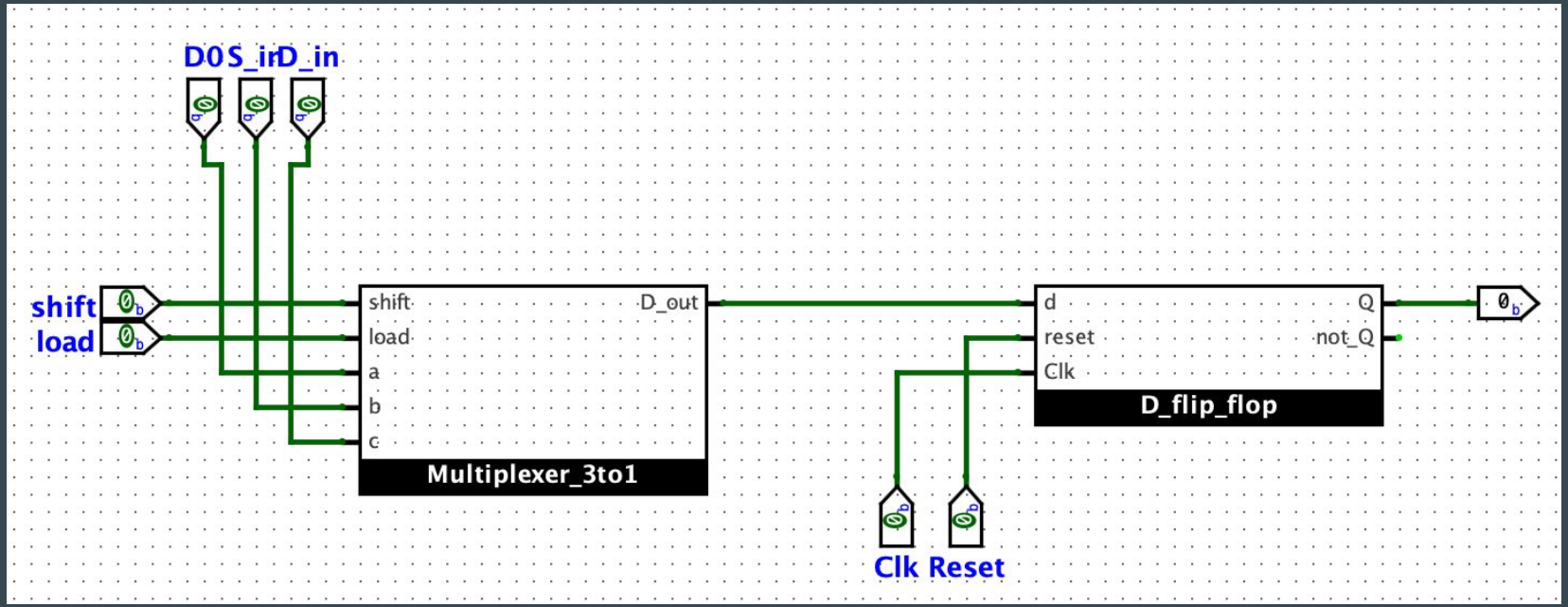
# 4. Shift Register

3 to 1 Multiplexer



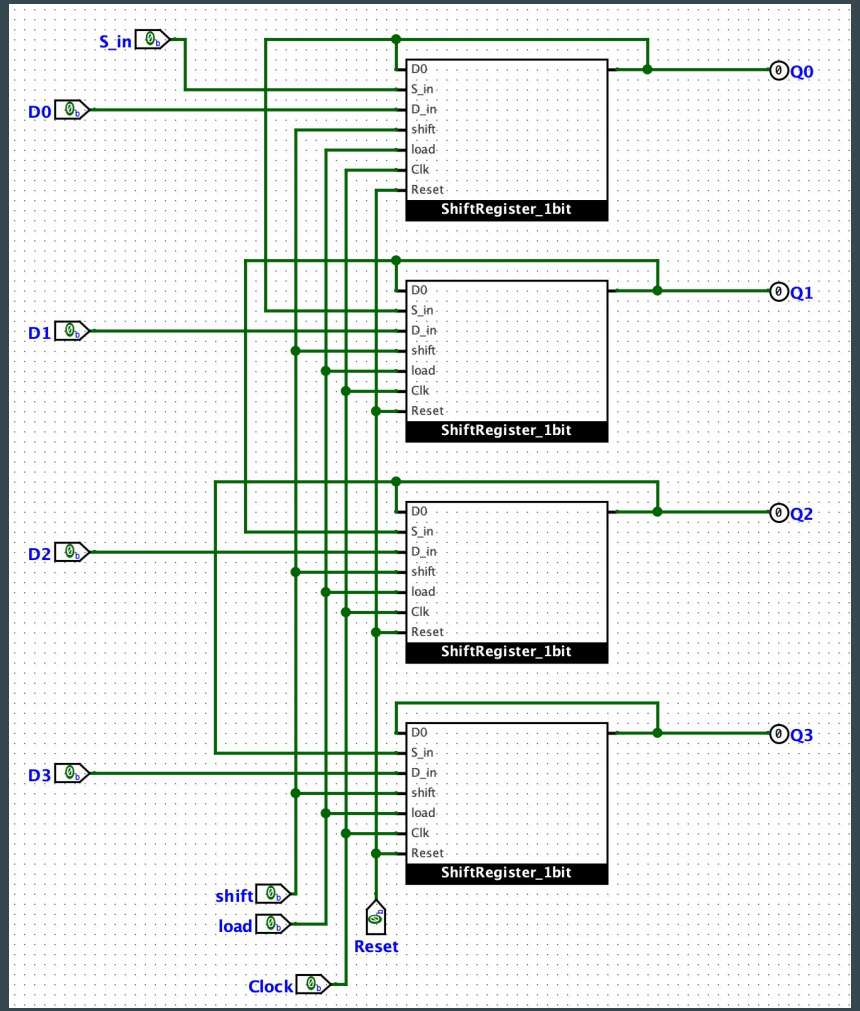
# 4. Shift Register

1-bit Shift Register



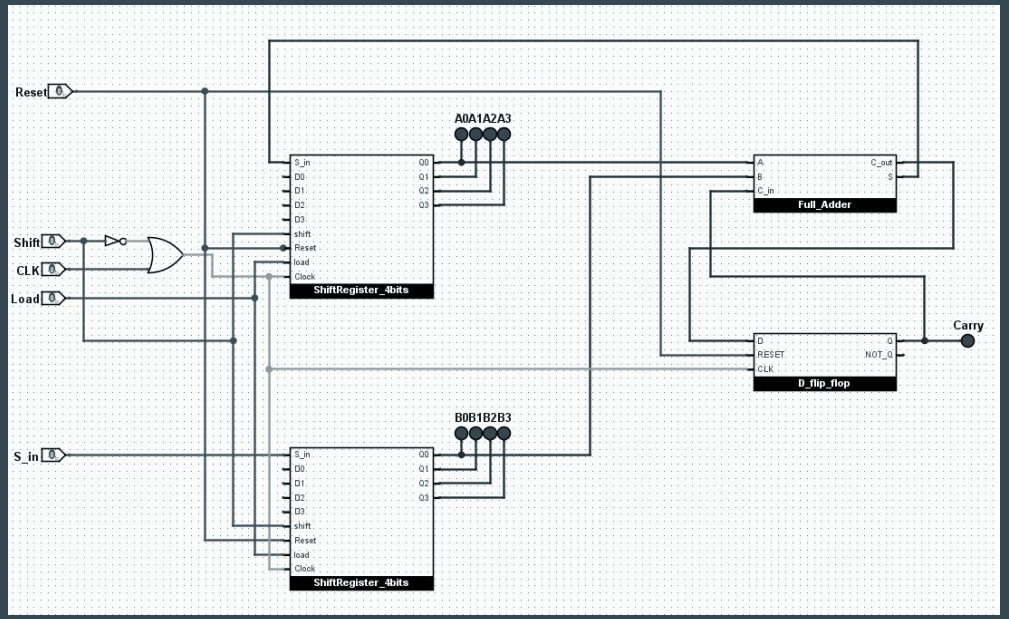
# 4. Shift Register

4-bit Shift register



# 5. Serial Adder

Pen & paper test




| | Tick 0 | Tick 1 | Tick 2 | Tick 3 | Tick 4 | Tick 5 | Tick 6 | Tick 7 | Tick 8 | Tick 9 | Tick 10 | Tick 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Register A | 0000 | 0000 | 0000 | 0000 | 0000 | 1000 | 1100 | 0110 | 0011 | 0001 | 0000 | 0000 |
| Register B | 0000 | 1000 | 1100 | 0110 | 0011 | 1001 | 0100 | 1010 | 0101 | 0010 | 0001 | 0000 |
| Carry | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| Sum | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Sin | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | - |

# 5. Serial Adder