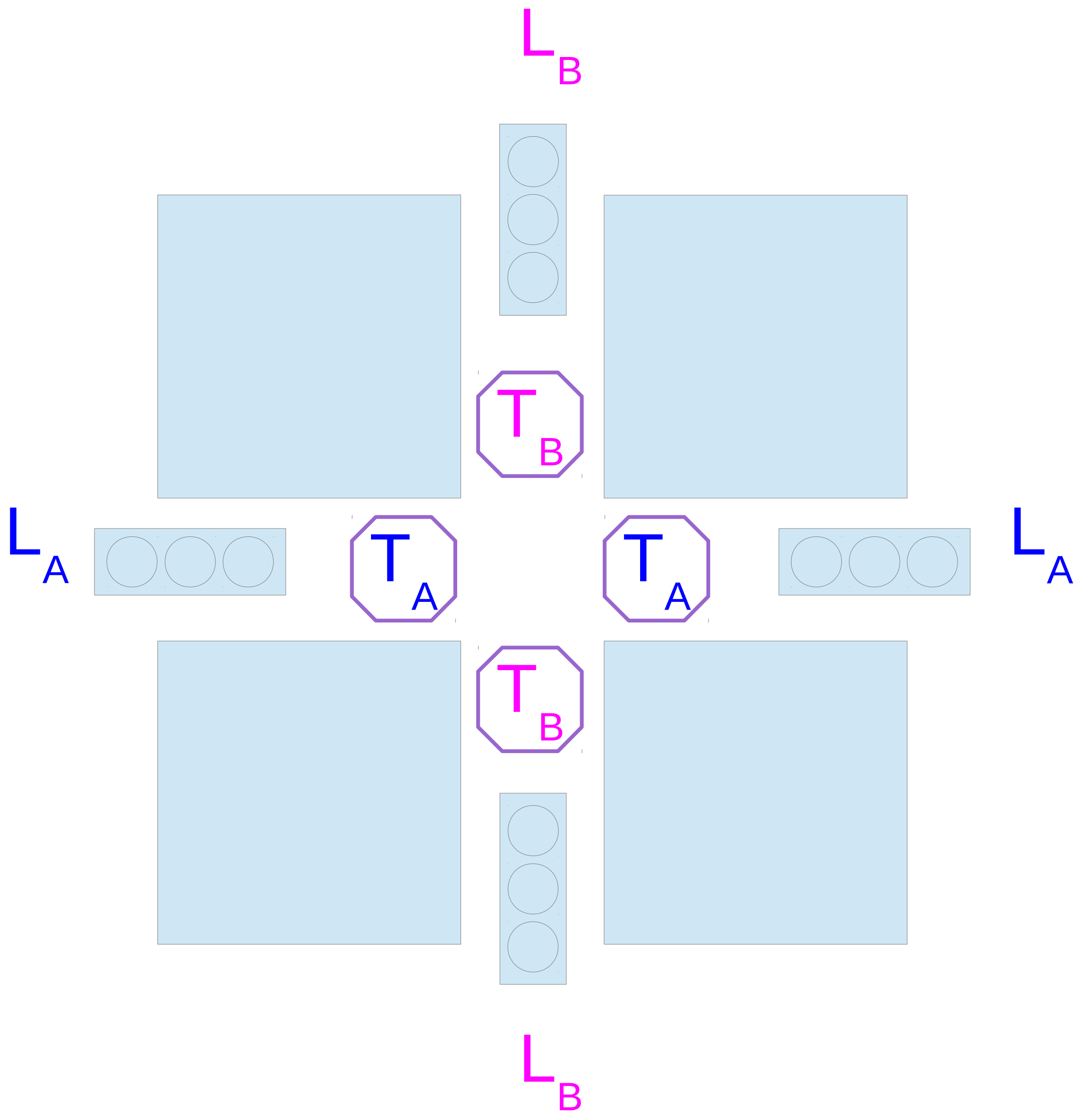
FSM Examples

Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice and Octave.

Latches and FF's

FSM Inputs and Outputs



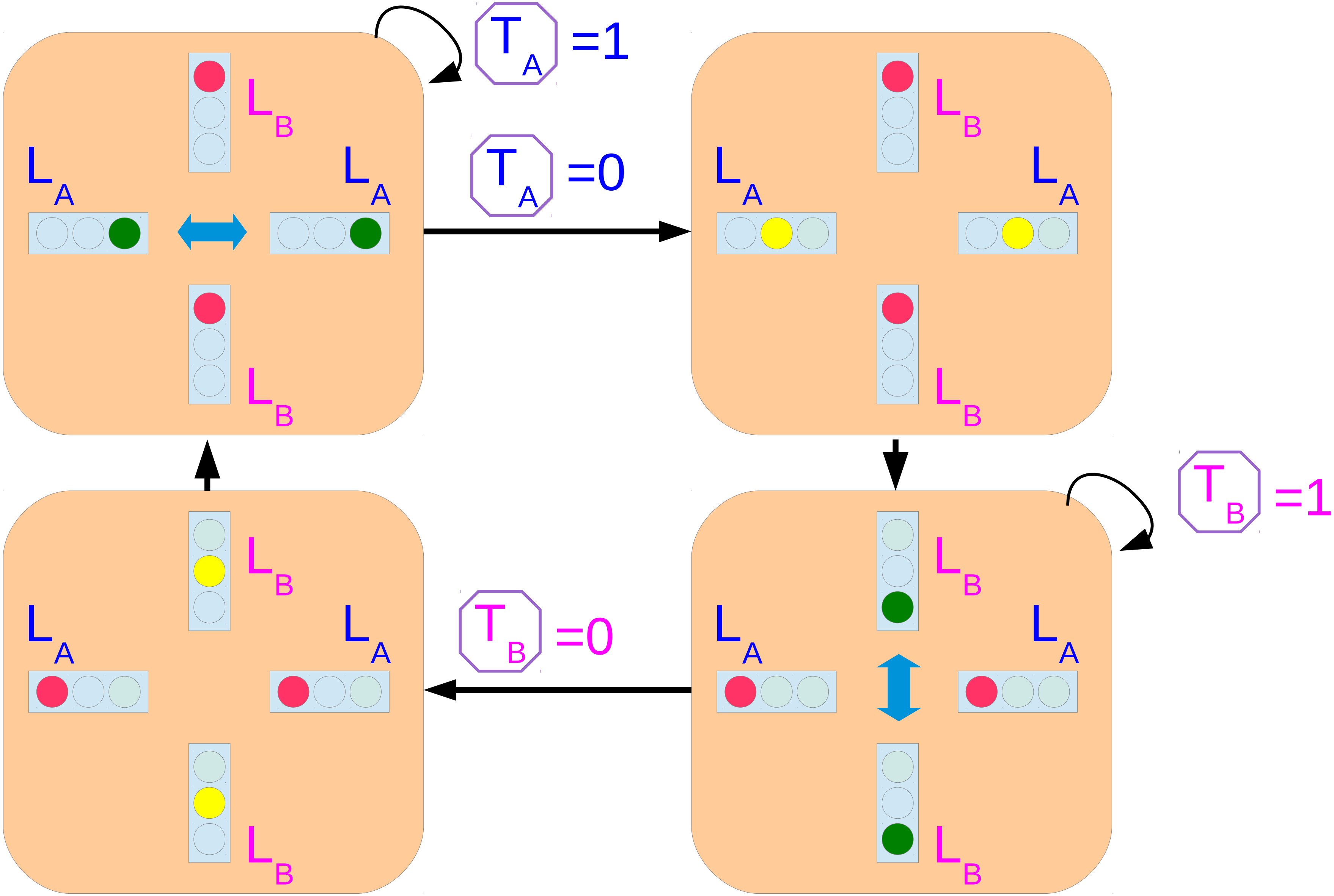
Traffic Lights - Outputs

L_A L_B

Sensor - Inputs

T_A T_B

States



Moore FSM State Transition Table

S_1	S_0	T_A	T_B	S'_1	S'_0
0	0	0	X	0	1
0	0	1	X	0	0
0	1	X	X	1	0
1	0	X	0	1	1
1	0	X	1	1	0
1	1	X	X	0	0

	S_1	S_0	T_A	T_B	S'_1
	0	0	0	X	0
	0	0	1	X	0
$\overline{S_1}\,S_0$	0	1	X	X	1
$S_1\,\overline{S_0}\,\overline{T_B}$	1	0	X	0	1
$S_1\,\overline{S_0}\,T_B$	1	0	X	1	1
	1	1	X	X	0

$$S'_1 = \overline{S_1}\,S_0 + S_1\,\overline{S_0}$$

$$= S_1 \oplus S_0$$

	S_1	S_0	T_A	T_B	S'_0
$\overline{S_1}\,\overline{S_0}\,\overline{T_A}$	0	0	0	X	1
	0	0	1	X	0
	0	1	X	X	0
$S_1\,\overline{S_0}\,\overline{T_B}$	1	0	X	0	1
	1	0	X	1	0
	1	1	X	X	0

$$S'_0 = \overline{S_1}\,\overline{S_0}\,\overline{T_A} + S_1\,\overline{S_0}\,\overline{T_B}$$

States

S_1	S_2	L_{A1}	L_{A0}	L_{B1}	L_{B0}		
0	0	0	0	1	0	🟢	🔴
0	1	0	1	1	0	🟡	🔴
1	0	1	0	0	0	🔴	🟢
1	1	1	0	0	1	🔴	🟡

🟢	00
🟡	01
🔴	10

S_1	S_2	L_{A1}
0	0	0
0	1	0
1	0	1
1	1	1

$$L_{A1} = S_1$$

S_1	S_2	L_{A0}
0	0	0
0	1	1
1	0	0
1	1	0

$$L_{A0} = \overline{S}_1 S_0$$

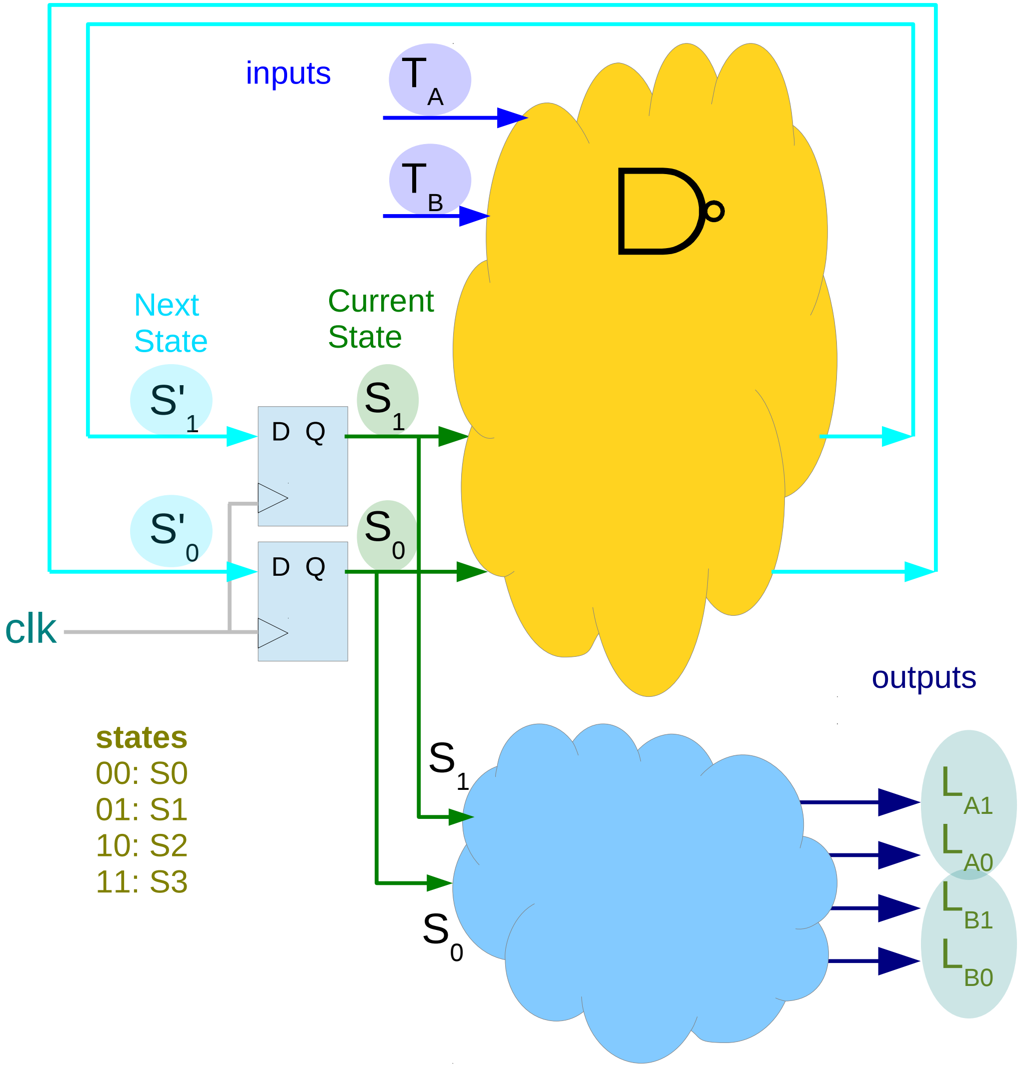
S_1	S_2	L_{B1}
0	0	1
0	1	1
1	0	0
1	1	0

$$L_{B1} = \overline{S}_1$$

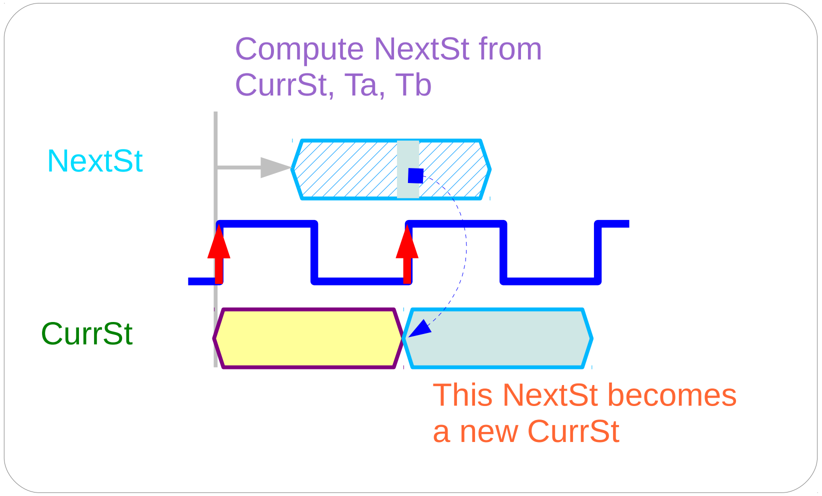
S_1	S_2	L_{B0}
0	0	0
0	1	0
1	0	0
1	1	1

$$L_{A0} = S_1 S_0$$

Moore FSM (1)

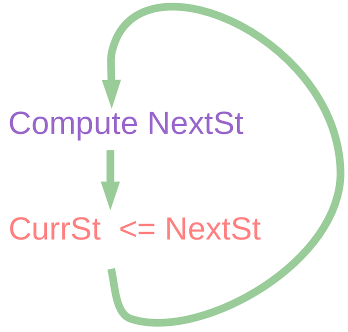


states
00: S0
01: S1
10: S2
11: S3



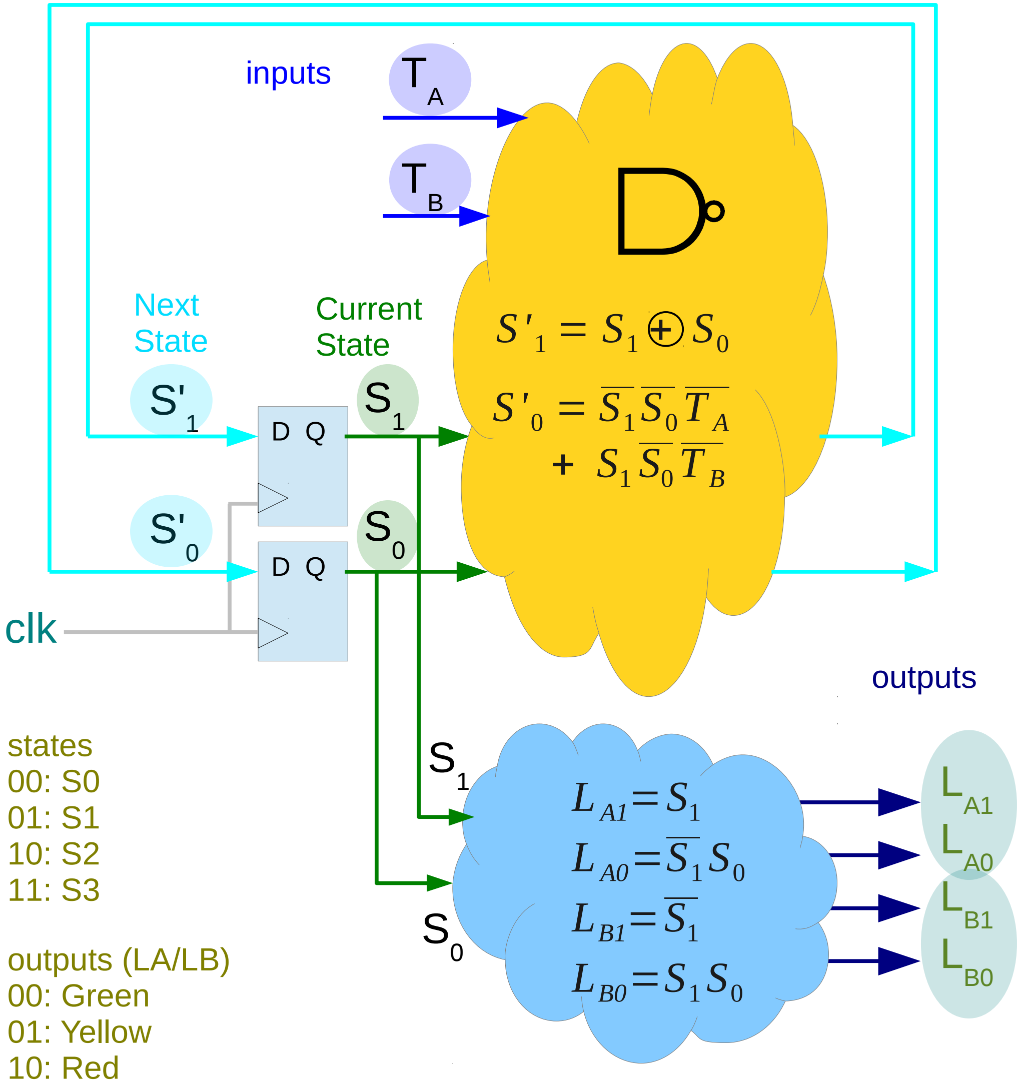
Compute NextSt from CurrSt, Ta, Tb

This NextSt becomes a new CurrSt



outputs (LA/LB)
00: Green
01: Yellow
10: Red
11: X

Moore FSM



states
00: S0
01: S1
10: S2
11: S3

outputs (LA/LB)
00: Green
01: Yellow
10: Red
11: X

Inputs T_A T_B

Current State S_1 S_0

Next States

$$S'_1 = S_1 \oplus S_0$$

$$S'_0 = \overline{S_1}\,\overline{S_0}\,\overline{T_A} + S_1\,\overline{S_0}\,\overline{T_B}$$

Current State S_1 S_0

Outputs

$L_{A1} = S_1$ $\quad L_{B1} = \overline{S_1}$

$L_{A0} = \overline{S_1} S_0$ $\quad L_{B0} = S_1 S_0$

Verilog Gate Level Design - testbench

```
`timescale 1ns/100ps

module traffic_controller_testbench;

  parameter cycle = 40;

  reg clock;

  always
    begin
      #(cycle/2) clock=~clock;
    end


  traffic_controller DUT (.clock(clock),
                         .reset(reset),
                         .TA(TA),
                         .TB(TB),
                         .LA(LA),
                         .LB(LB) );


  reg       reset;
  reg       TA, TB;
  wire [1:0] LA, LB;
```

```
  initial
    begin
      clock = 1;
      reset = 1;
      TA = 1;
      TB = 0;

          #1;
        #(cycle)       reset = 0;
        #(cycle)       TB = 1;
        #(cycle)       TA = 0;
        #(cycle*3)   TA = 1;
                       TB = 0;
        #(cycle*3)   TA = 0;
    end

  initial
    begin
      $dumpfile("traffic.vcd");
      $dumpvars(0, DUT);
      #(cycle * 10);
      $finish;
    end

endmodule
```

Verilog Gate Level Design – traffic_gate.v

```
module traffic_controller(clock, reset, TA, TB, LA, LB);
 input clock, reset;
 input TA, TB;
 output [1:0] LA, LB;

 reg    [1:0] S;
 wire   [1:0] NextS;

 always @(posedge clock)
   begin: SEQ
     if (reset)
       #8 S = 2'b00;
     else
       #8 S = NextS;
   end
```

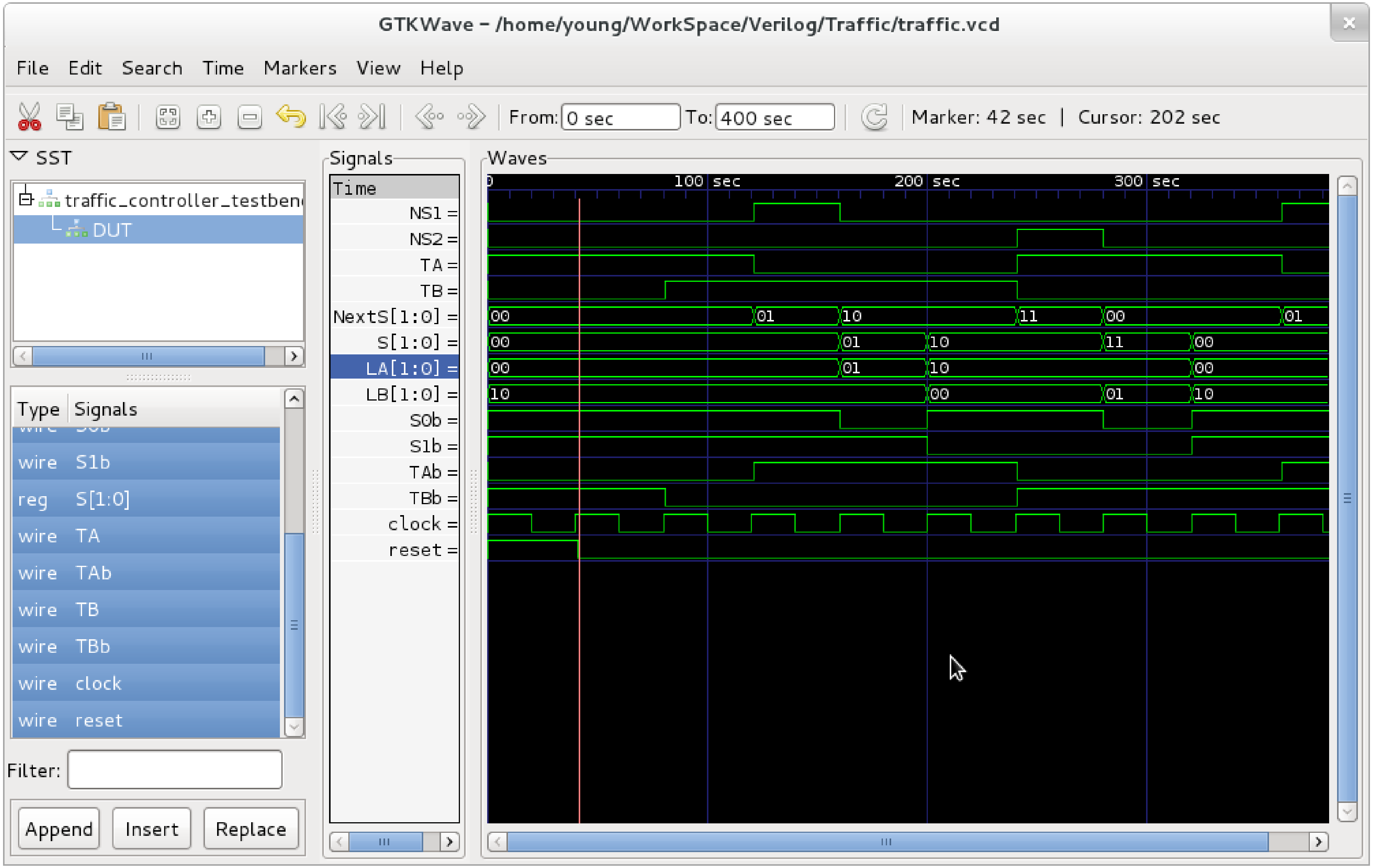
```
  not #8 (S1b, S[1]);
  not #8 (S0b, S[0]);
  not #8 (TAb, TA);
  not #8 (TBb, TB);

  xor #8 (NextS[1], S[1], S[0]);
  or   #8 (NextS[0], NS1, NS2);
  and #8 (NS1, S1b,  S0b, TAb);
  and #8 (NS2, S[1], S0b, TBb);


  buf #8 (LA[1], S[1]);
  and #8 (LA[0], S1b, S[0]);
  not #8 (LB[1], S[1]);
  and #8 (LB[0], S[1], S[0]);

endmodule
```

VCD Output with zero delay



GTKWave – /home/young/WorkSpace/Verilog/Traffic/traffic.vcd

From: 0 sec To: 400 sec Marker: 42 sec | Cursor: 202 sec

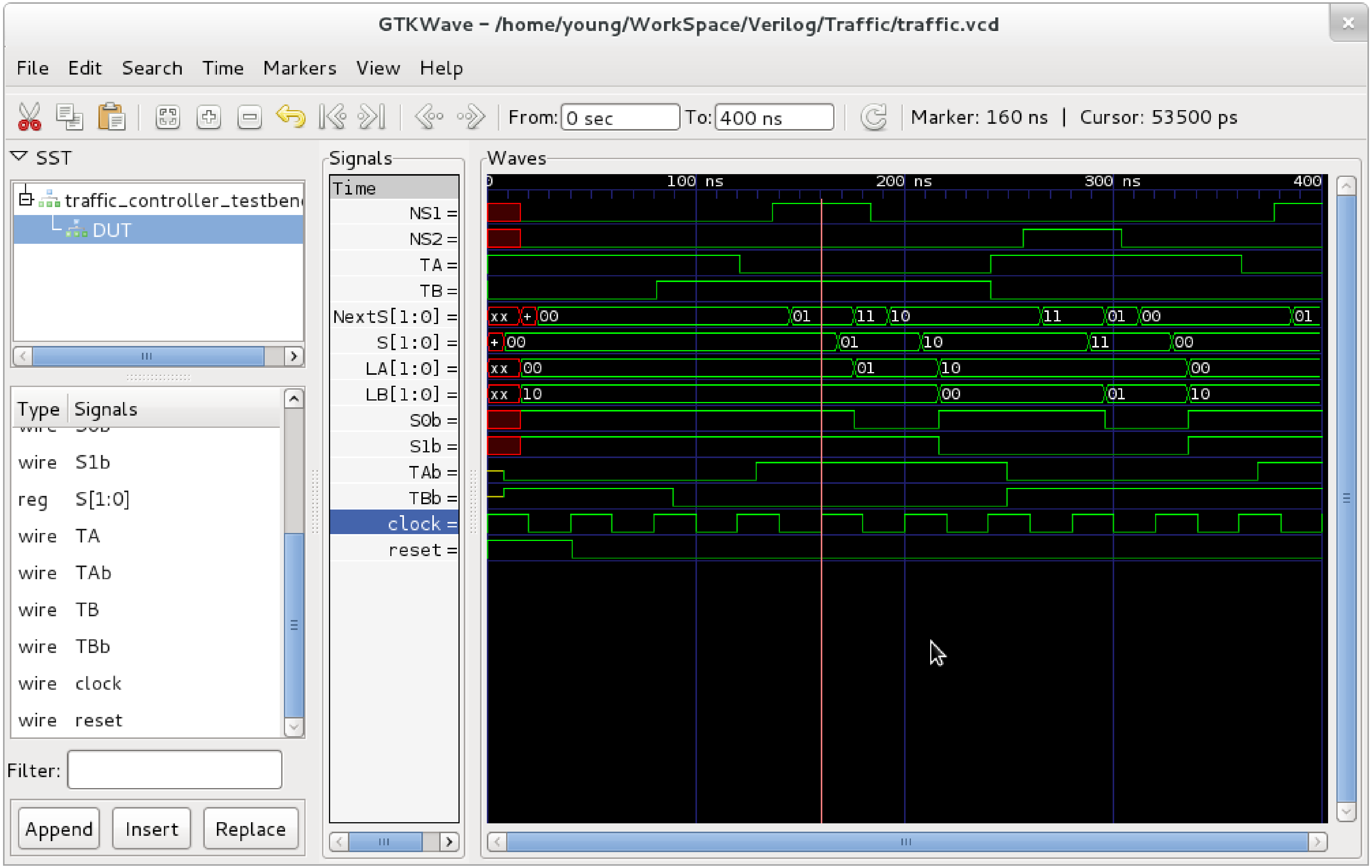
SST

traffic_controller_testben
DUT

Type	Signals
wire	S0b
wire	S1b
reg	S[1:0]
wire	TA
wire	TAb
wire	TB
wire	TBb
wire	clock
wire	reset

Filter:

VCD Output with gate delays



GTKWave - /home/young/WorkSpace/Verilog/Traffic/traffic.vcd

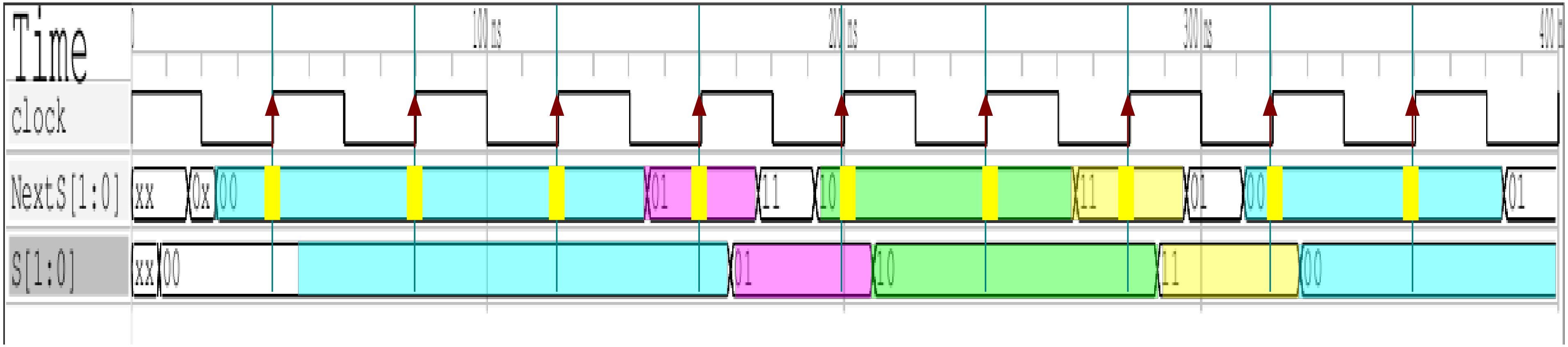
From: 0 sec To: 400 ns

Marker: 160 ns | Cursor: 53500 ps

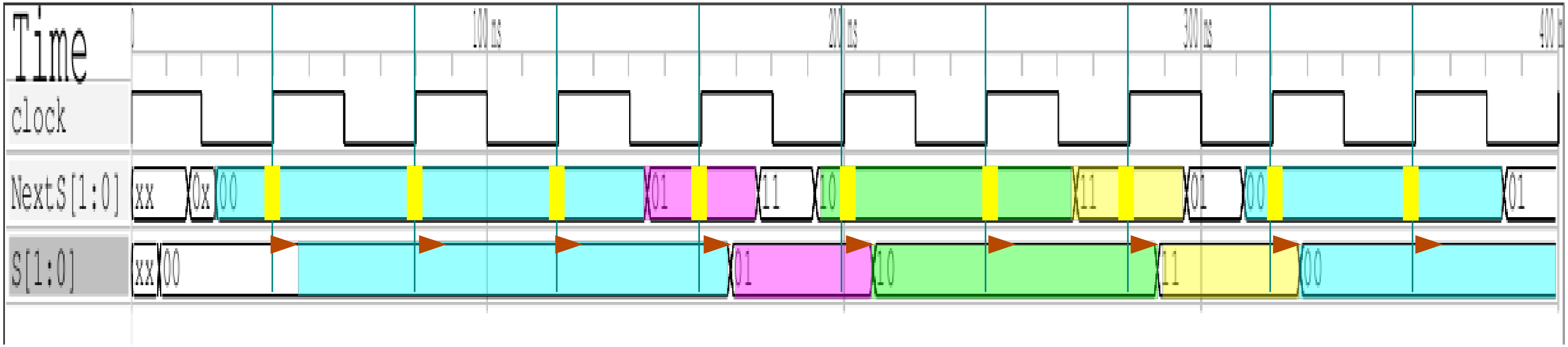
Type	Signals
wire	S0b
wire	S1b
reg	S[1:0]
wire	TA
wire	TAb
wire	TB
wire	TBb
wire	clock
wire	reset

Filter:

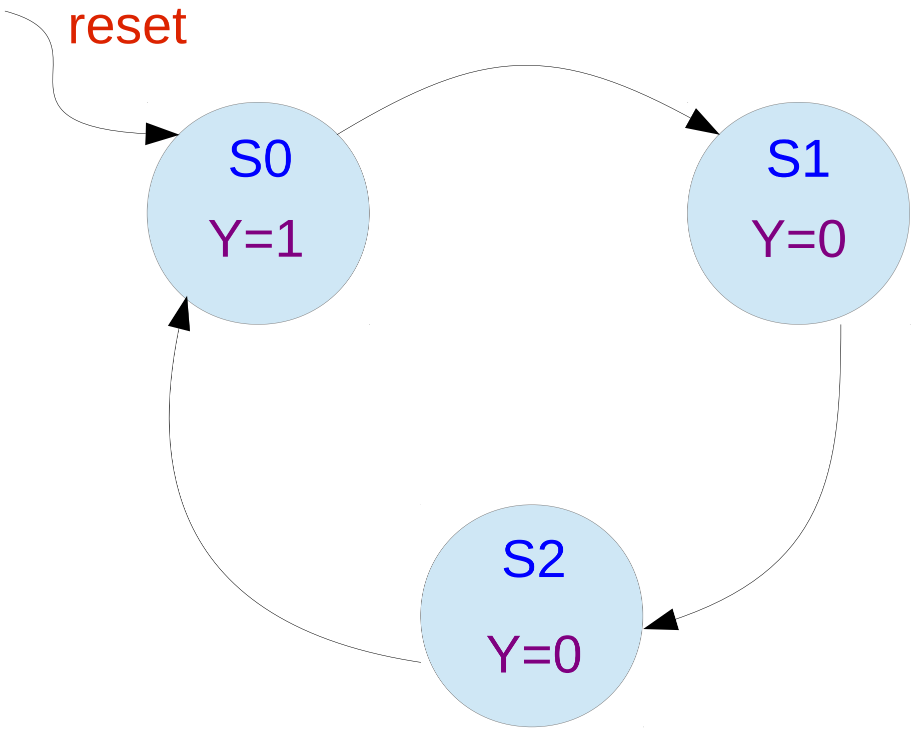
VCD Output with gate delays



Output Delay



Divide By N Counter FSM



Input: none

Output: Y=1 every 3 cycles

State Transition Table

Curr St	Next St
S0	S1
S1	S2
S2	S0

Output Table

Curr St	Output
S0	1
S1	0
S2	0

Encoding States

State Transition Table

Curr St	Next St
S0	S1
S1	S2
S2	S0

Output Table

Curr St	Output
S0	1
S1	0
S2	0

S_1	S_0	S'_1	S'_0
0	0	0	1
0	1	1	0
1	0	0	0

S_1	S_0	Y
0	0	1
0	1	0
1	0	0

$$S'_1 = \overline{S_1} S_0$$

$$S'_0 = \overline{S_1}\,\overline{S_0}$$

$$Y = \overline{S_1}\,\overline{S_0}$$

State Transition Table

Curr St	Next St
S0	S1
S1	S2
S2	S0

Output Table

Curr St	Output
S0	1
S1	0
S2	0

S_2	S_1	S_0	S'_2	S'_1	S'_0
0	0	1	0	1	0
0	1	0	1	0	0
1	0	0	0	0	1

S_2	S_1	S_0	Y
0	0	1	1
0	1	0	0
1	0	0	0

$$S'_2 = \overline{S_2} S_1 \overline{S_0} \quad \Rightarrow S_1$$

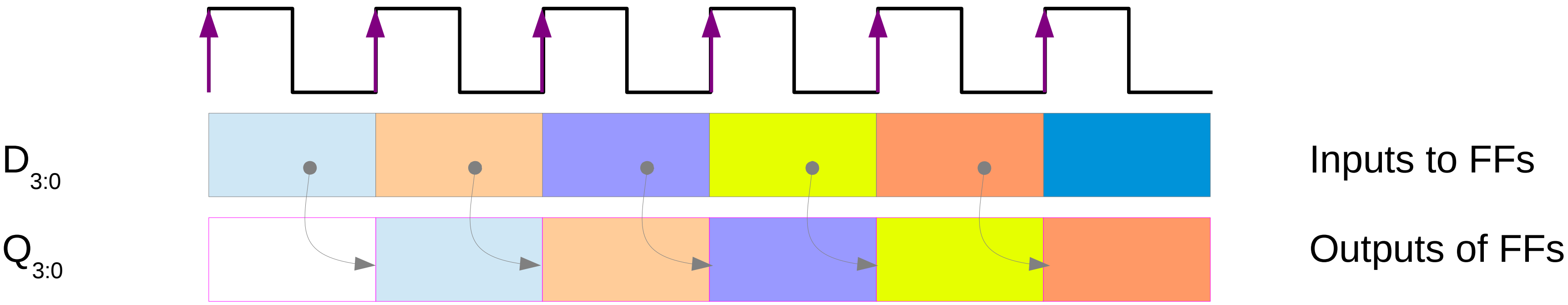
$$S'_1 = \overline{S_2}\,\overline{S_1} S_0 \quad \Rightarrow S_0$$

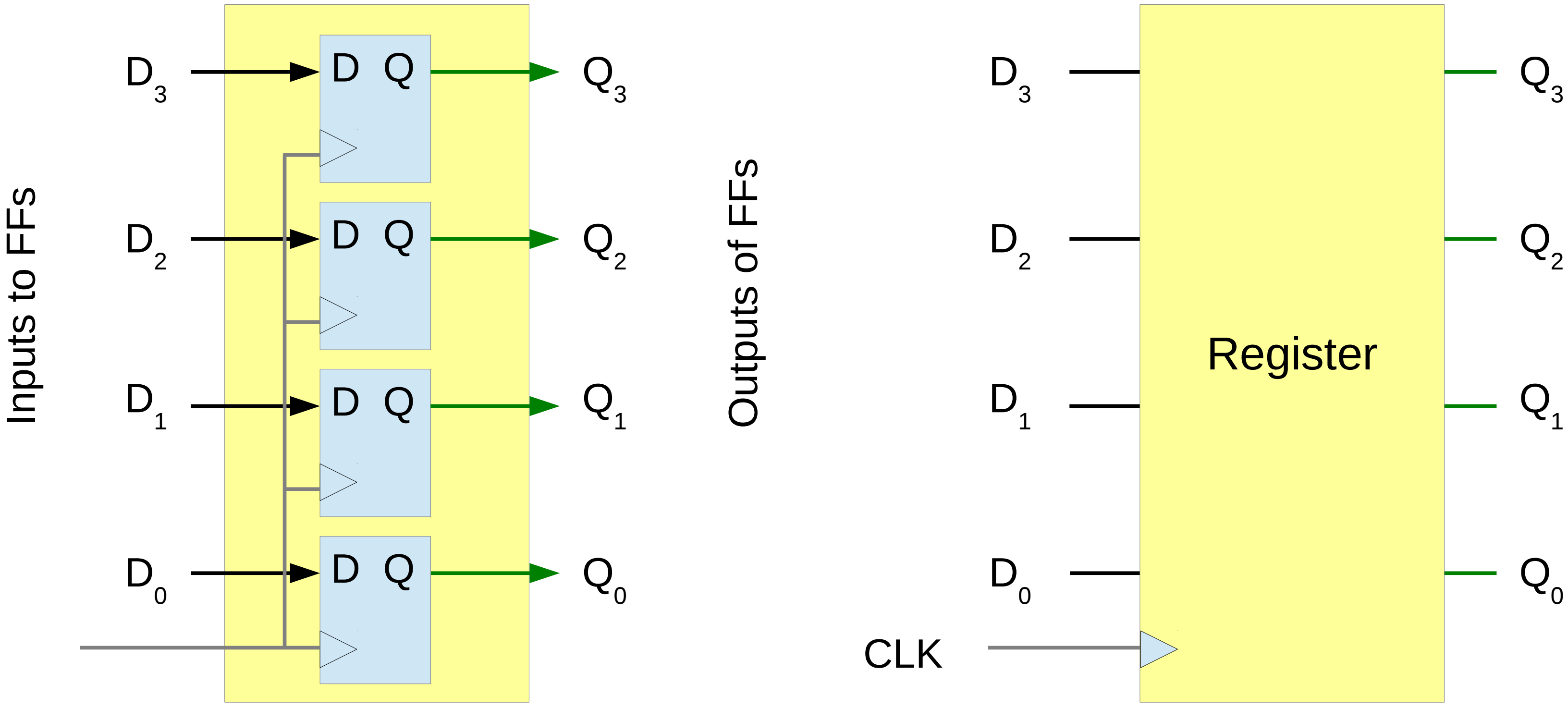
$$S'_0 = S_2 \overline{S_1}\,\overline{S_0} \quad \Rightarrow S_2$$

$$Y = \overline{S_2}\,\overline{S_1} S_0 \quad \Rightarrow S_0$$

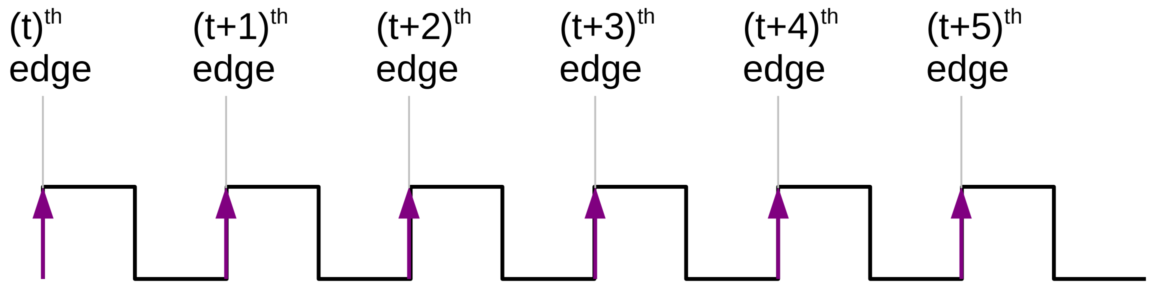
Resolution Time

FF Timing (Ideal)

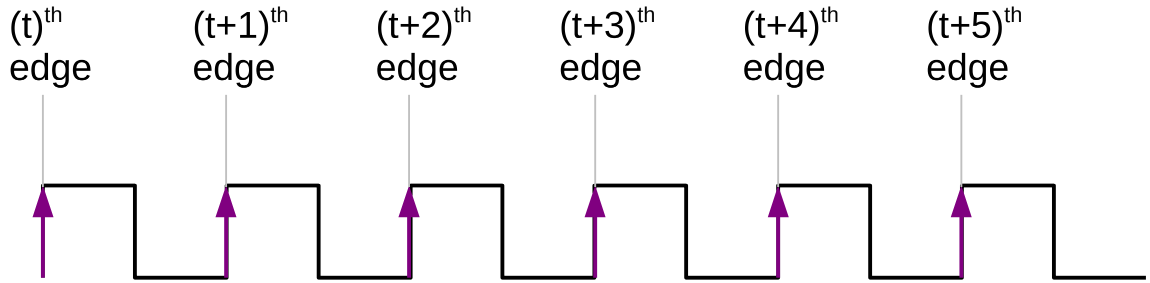




Sequence of States

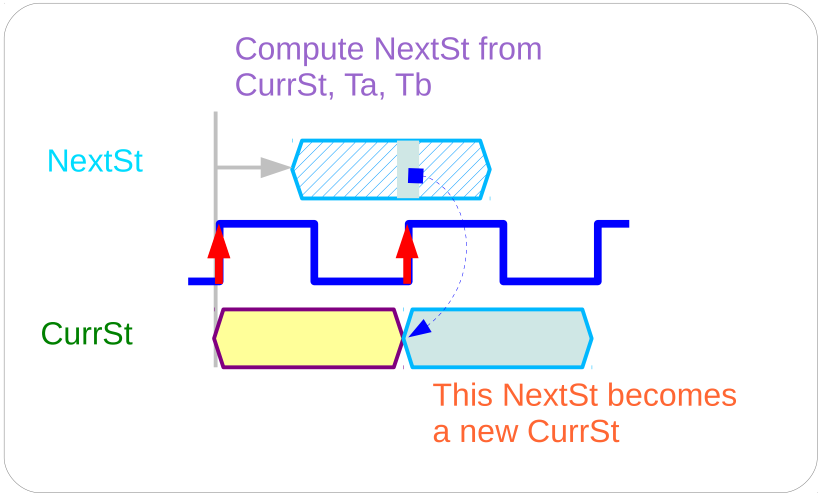


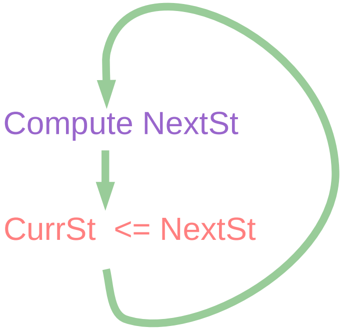
$D_{3:0}$							Inputs to FFs
$Q_{3:0}$	Q(t)	Q(t+1)	Q(t+2)	Q(t+3)	Q(t+4)	Q(t+5)	Outputs of FFs



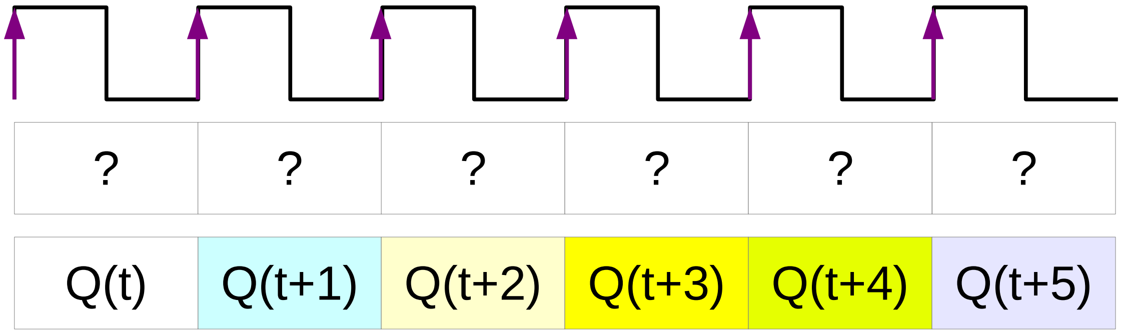
$D_{3:0}$	?	?	?	?	?	?	Find inputs to FFs
$Q_{3:0}$	Q(t)	Q(t+1)	Q(t+2)	Q(t+3)	Q(t+4)	Q(t+5)	which will make outputs in this sequence

When NextSt becomes CurrSt

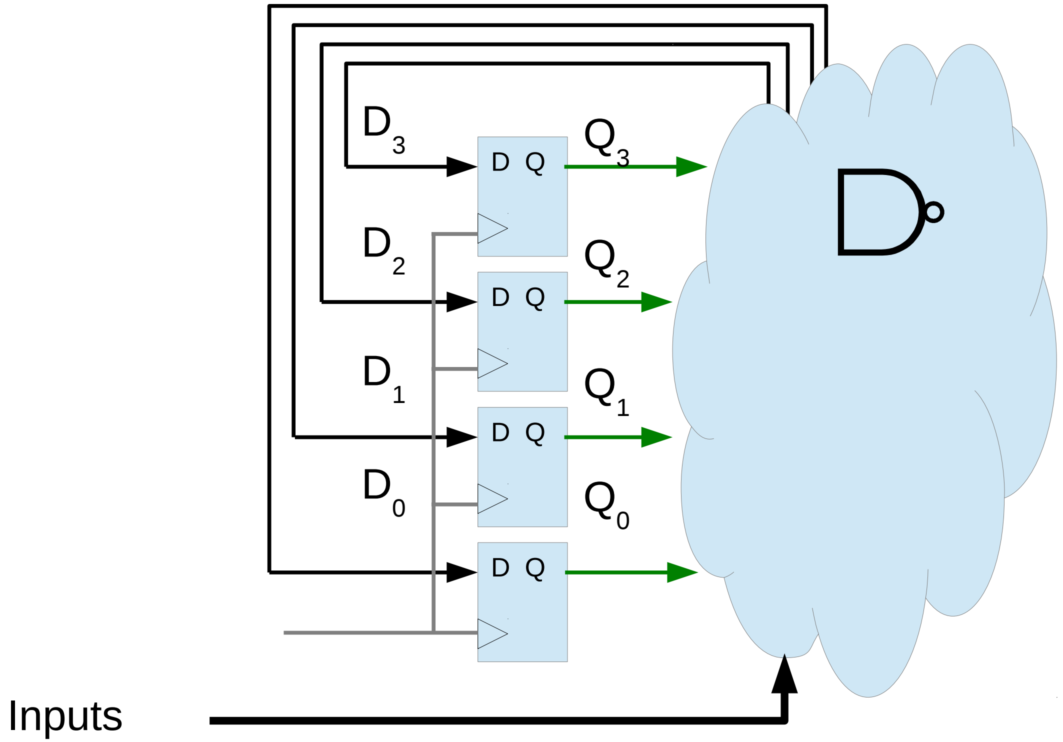




Finding FF Inputs



$D_{3:0}$	?	?	?	?	?	?	Inputs to FFs
$Q_{3:0}$	Q(t)	Q(t+1)	Q(t+2)	Q(t+3)	Q(t+4)	Q(t+5)	Outputs of FFs



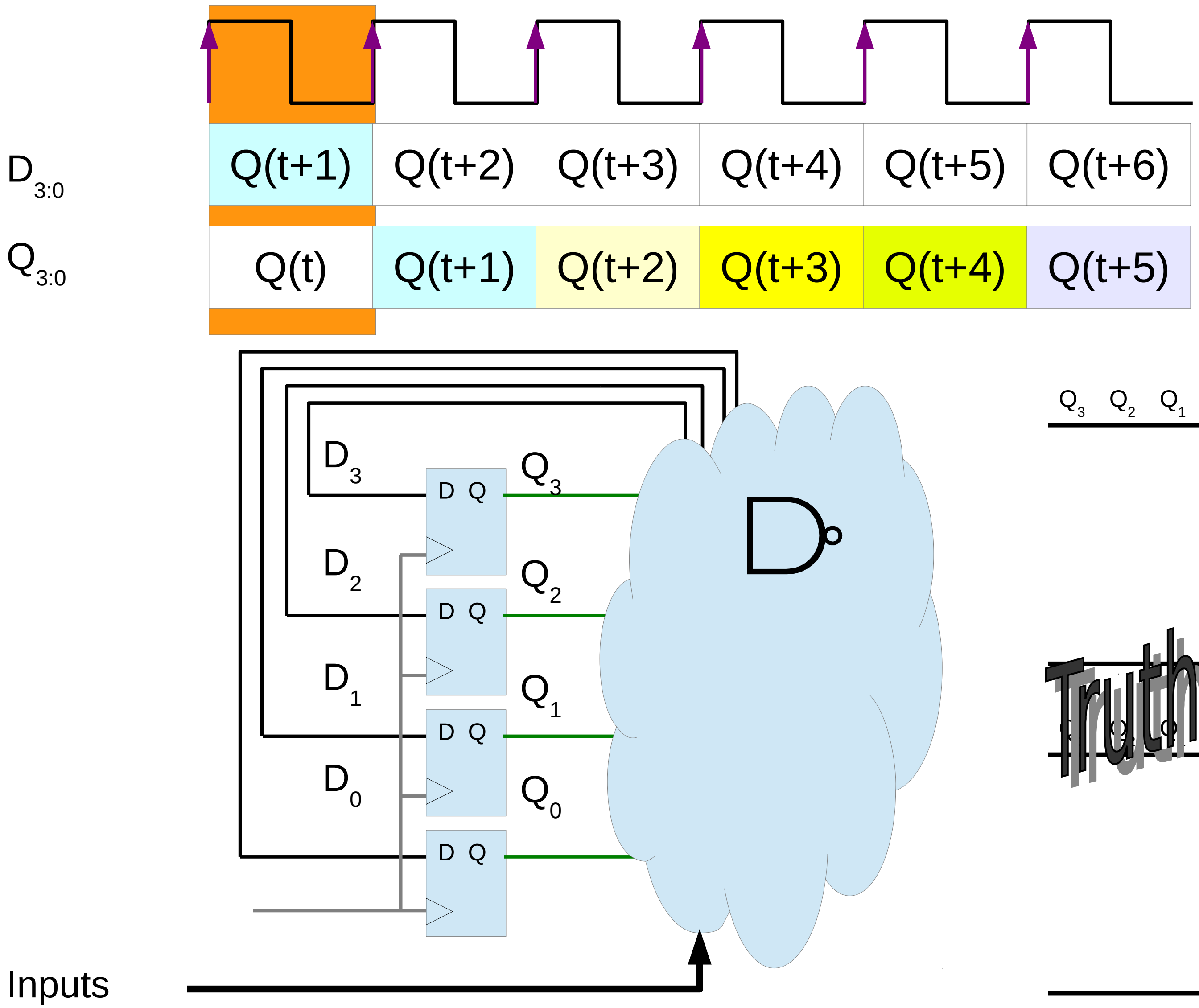
During the t^th clock edge period,

Compute the next state Q(t+1) using the current state Q(t) and other external inputs

Place it to FF inputs

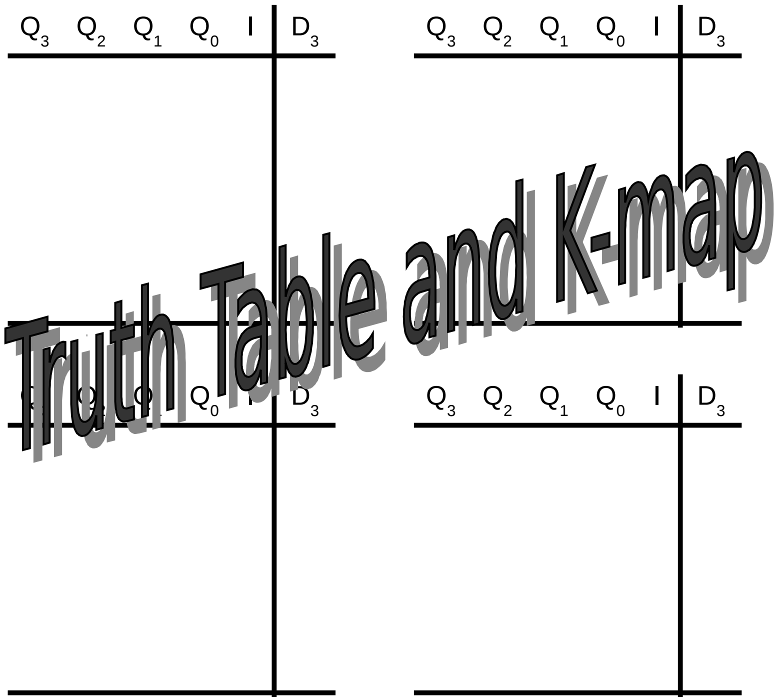
After the next clock edge, (t+1)^th, the computed next state Q(t+1) becomes the current state

Method of Finding FF Inputs



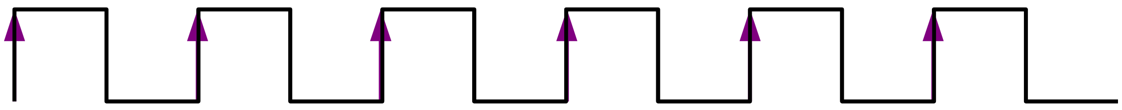
$D_{3:0}$	Q(t+1)	Q(t+2)	Q(t+3)	Q(t+4)	Q(t+5)	Q(t+6)
$Q_{3:0}$	Q(t)	Q(t+1)	Q(t+2)	Q(t+3)	Q(t+4)	Q(t+5)

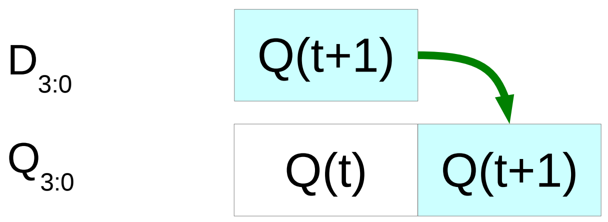
Find the boolean functions D3, D2, D1, D0 in terms of Q3, Q2, Q1, Q0, and external inputs for all possible cases.



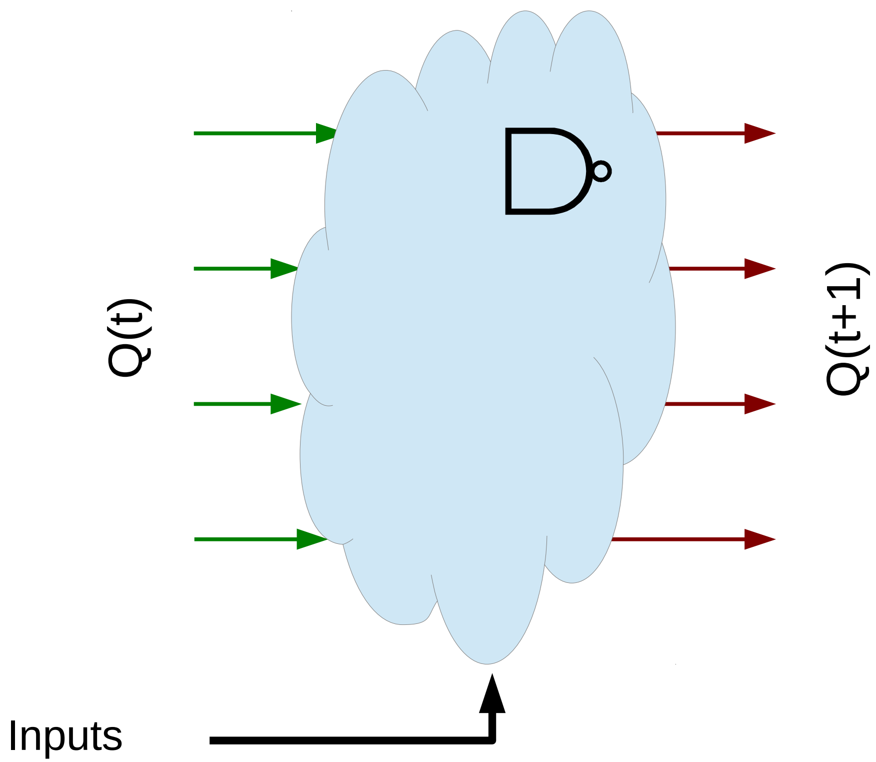
Q_3 Q_2 Q_1 Q_0 I D_3
Q_3 Q_2 Q_1 Q_0 I D_3
Q_3 Q_2 Q_1 Q_0 I D_3
Q_3 Q_2 Q_1 Q_0 I D_3
Truth Table and K-map

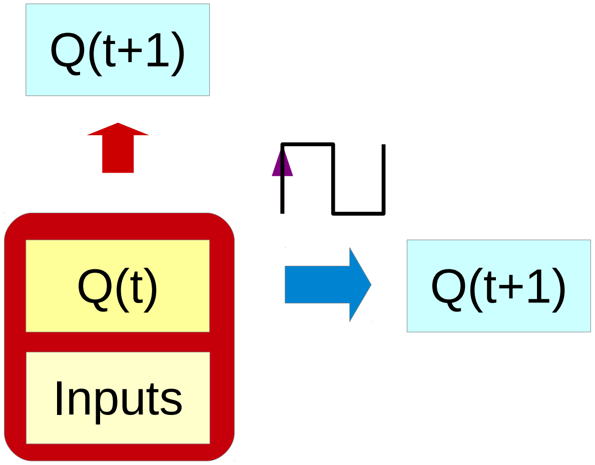
State Transition





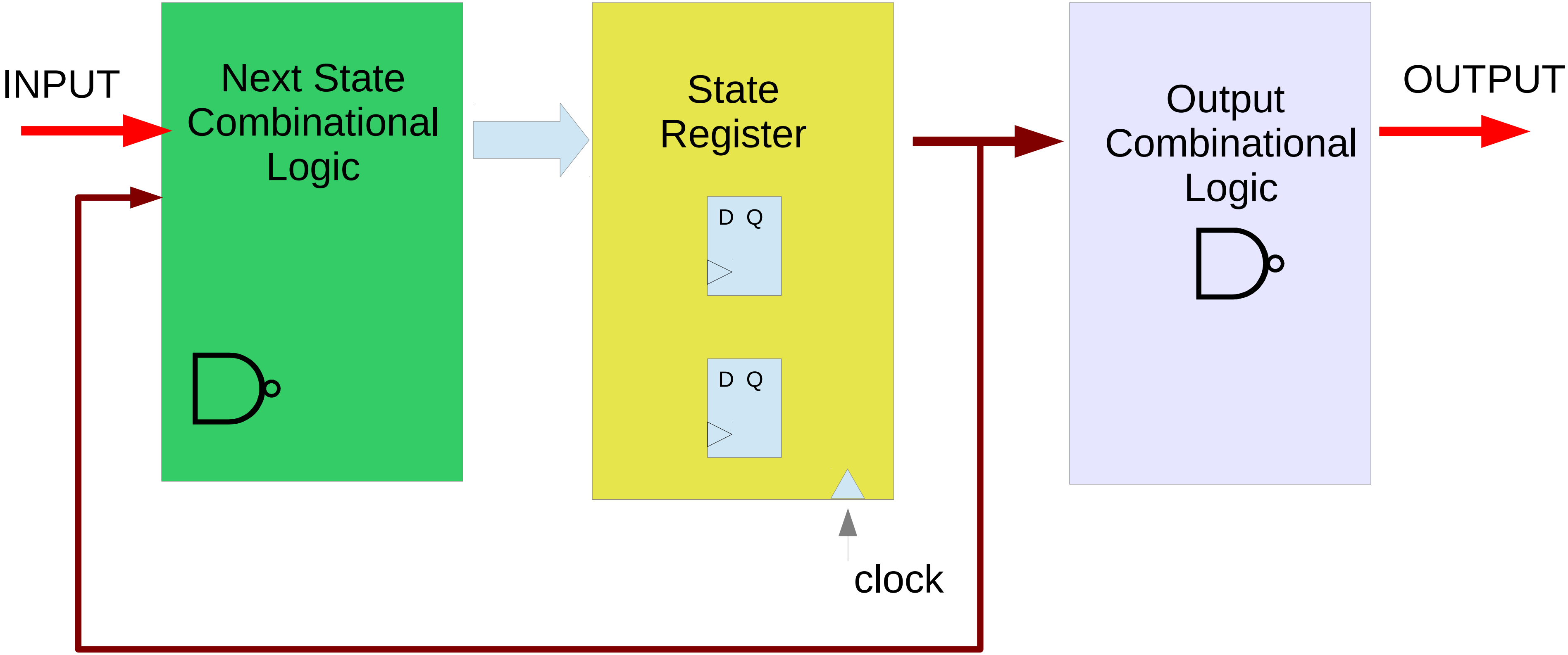
<u>Compute the next state</u>
using the current state
and external inputs
in the current clock cycle



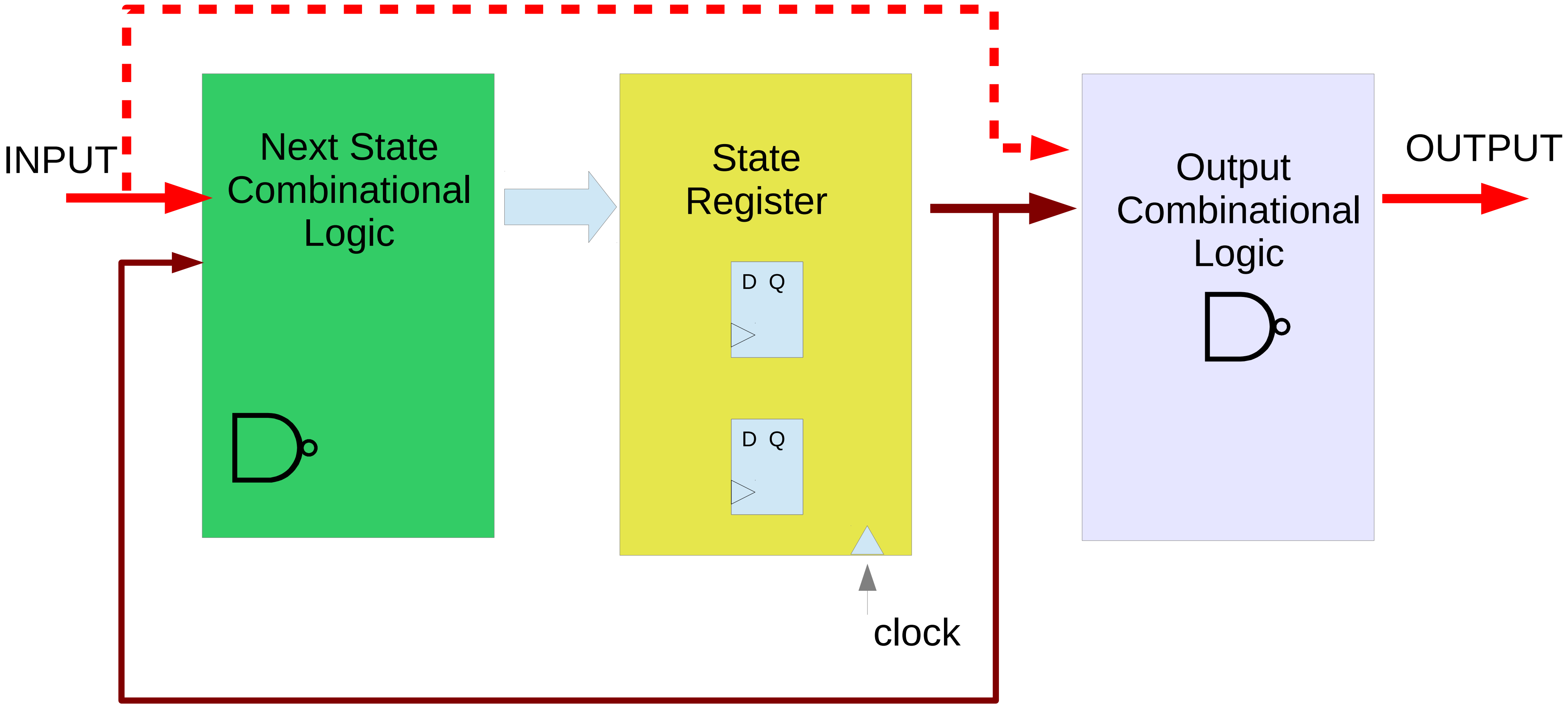


After the next clock edge,
the computed next state (FF Inputs)
becomes the current state (FF Outputs)

Moore FSM


INPUT
Next State Combinational Logic
State Register
D Q
D Q
clock
Output Combinational Logic
OUTPUT

Mealy Machine



References

[1] http://en.wikipedia.org/
[2] M. M. Mano, C. R. Kime, “Logic and Computer Design Fundamentals”, 4th ed.
[3] J. Stephenson, Understanding Metastability in FPGAs. Altera Corporation white paper. July 2009.