

Lecture 4



信息科学与技术学院
School of Information Science and Technology

Sequential Digital Circuits

Junrui Liang
ShanghaiTech University



Introduction to Information Science and Technology (Electronics)

ShanghaiTech University

Outlines



信息科学与技术学院
School of Information Science and Technology

- Sequential logic
 - » definition
- Latches and flip-flop
 - » The evolution of latches and flip-flops
 - » Synchronous and asynchronous circuits
 - » Example: 4 x 3 memory
- Finite state machines (FSM)
 - » Turnstile example
 - » Traffic light example

Introduction to Information Science and Technology (Electronics)

ShanghaiTech University

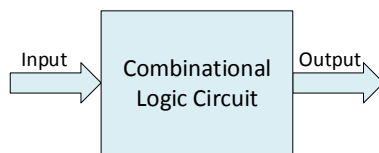


Outlines

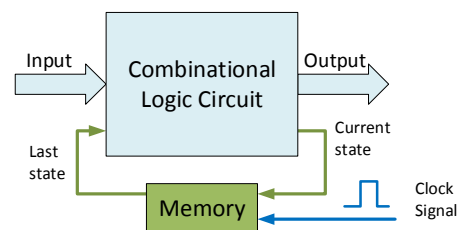
- **Sequential logic**
 - » **definition**
- Latches and flip-flop
 - » The evolution of latches and flip-flops
 - » Synchronous and asynchronous circuits
 - » Example: 4 x 3 memory
- Finite state machines (FSM)
 - » Turnstile example
 - » Traffic light example



Combinational Logic vs. Sequential Logic



- Output depends on
 - » Present input values
- No memory



- Output depends on
 - » Present inputs
 - » **The history of past inputs**
- With memory

(Figures are from internet)



Outlines

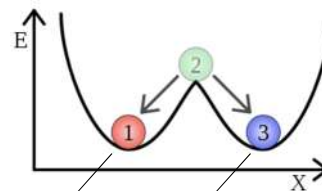
- Sequential logic
 - » definition
- **Latches and flip-flop**
 - » **The evolution of latches and flip-flops**
 - » Synchronous and asynchronous circuits
 - » Example: 4 x 3 memory
- Finite state machines (FSM)
 - » Turnstile example
 - » Traffic light example



Latch



- Bistable element



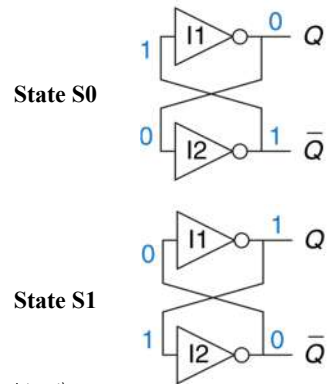
Memorizing
the state

(Figures are from internet)



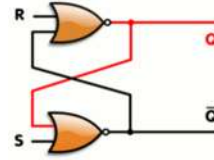
Digital Latch

- Cross-coupled inverter pair
- SR latch (with input)

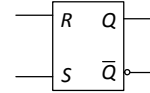


(Figures are from internet)

- » S: set (set output to 1)
- » R: reset (set output to 0)

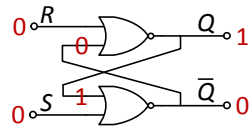


- » Symbols



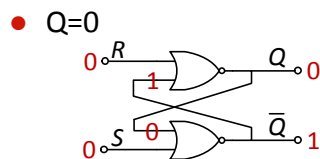
SR Latch ---- S=0, R=0

- Q=1
- True table



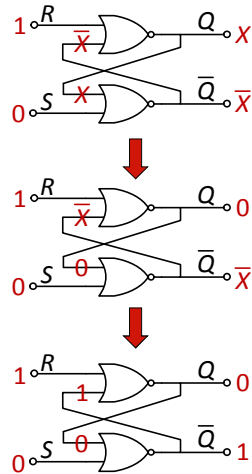
Hold state

S	R	Q	Q̄
0	0	Q_{prev}	\bar{Q}_{prev}





SR Latch ---- S=0, R=1

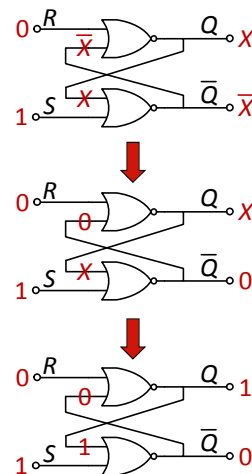


- True table

	S	R	Q	\bar{Q}
Hold state	0	0	Q_{prev}	\bar{Q}_{prev}
Reset	0	1	0	1



SR Latch ---- S=1, R=0

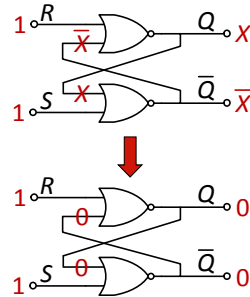


- True table

	S	R	Q	\bar{Q}
Hold state	0	0	Q_{prev}	\bar{Q}_{prev}
Reset	0	1	0	1
Set	1	0	1	0



SR Latch ---- S=1, R=1



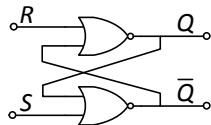
- True table

	S	R	Q	\bar{Q}
Hold state	0	0	Q_{prev}	\bar{Q}_{prev}
Reset	0	1	0	1
Set	1	0	1	0
Not allow!	1	1	0	0



Timing Control

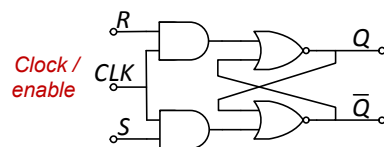
- Transparent latch



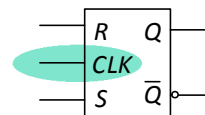
- Operation

C/E	Action
0	No action (keep state)
1	The same as non-clocked SR latch

- Gated SR Latch (level-sensitive)



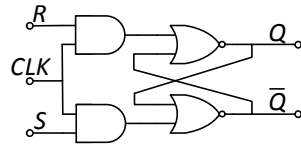
- Symbol





State Determination → D Latch

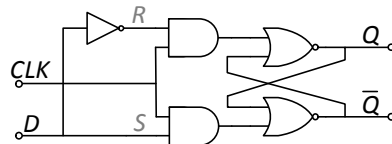
- Avoid the R=1, S=1 confusion



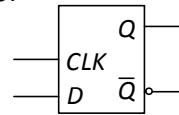
- Operation

CLK	D	\bar{D}	S	R	Q	\bar{Q}
0	X	\bar{X}	0	0	Q_{prev}	\bar{Q}_{prev}
1	0	1	0	1	0	1
1	1	0	1	0	1	0

- Modification → D latch

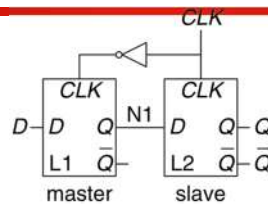


- Symbol



More Accurate Timing Control → Edge-Triggering

- D flip-flop

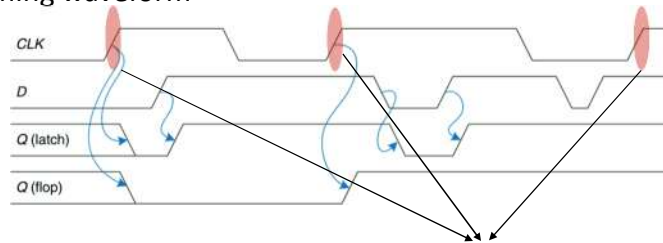


- CLK = 0
 - » Master enabled; D ⇒ N₁
 - » Slave disabled;
- CLK = 1
 - » Master disabled;
 - » Slave enabled; N₁ ⇒ Q
- The moment CLK changes from 0 to 1
 - » Rising edge
 - » Data is transmitted to Q

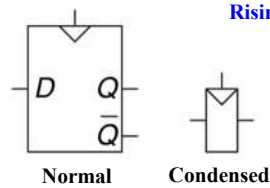


Edge-Triggering

- Timing waveform



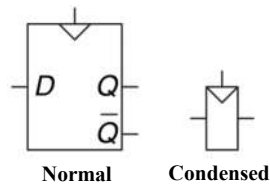
- Symbols



Rising Edges



Summary: D flip-flop



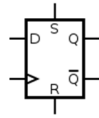
D	Q_{next}
0	0
1	1
X	Q

$$Q(next) = D$$



Other flip-flops

- SR (Set & Reset) flip-flop



S	R	Q(next)
0	0	Q
0	1	0
1	0	1
1	1	NA

$$Q(\text{next}) = S + R'Q$$

$$SR = 0$$

- JK flip-flop (Origin: Ref. 2)



J	K	Q(next)
0	0	Q
0	1	0
1	0	1
1	1	Q'

$$Q(\text{next}) = JQ' + K'Q$$

- T (Trigger) flip-flop



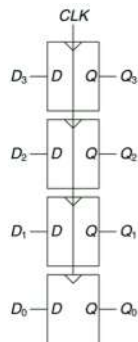
T	Q(next)
0	Q
1	Q'

$$Q(\text{next}) = TQ' + T'Q$$

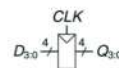


Flip-Flop Derivations

- Register



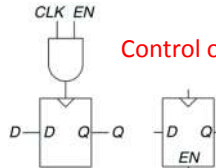
Separating the "states"
 key building block of most sequential
 circuits





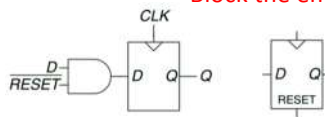
Flip-Flop Derivations ctd.

- Enabled flip-flop



Control of the data loading instant

- Resettable flip-flop



Block the effect of input D (reset=0)



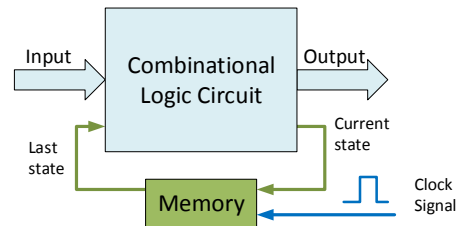
Outlines

- Sequential logic
 - » definition
- **Latches and flip-flop**
 - » The evolution of latches and flip-flops
 - » **Synchronous and asynchronous circuits**
 - » Example: 4 x 3 memory
- Finite state machines (FSM)
 - » Turnstile example
 - » Traffic light example



Synchronous Circuits

- Synchronous circuit

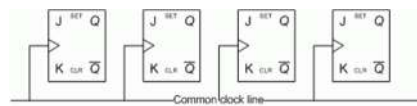


(Figures are from internet)



Example

- Synchronous circuit



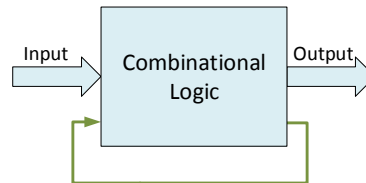
- » Common **clock signal**
- » Output only change at the **edge** of clock pulse
- » clock signal should be long enough so that the **critical path** can settle before next clock edge
- » Easy design

(Figures are from internet)



Asynchronous Circuits

- Asynchronous circuit or self-timed circuit

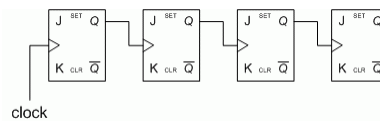


(Figures are from internet)



Example

- Asynchronous circuit or self-timed circuit

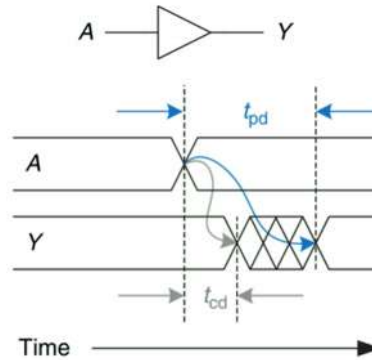


- » Not governed by global clock
- » Resulting state can be sensitive to the relative arrival times of inputs at gates, the **race condition**



Review: Delay in Combinational Circuit

- contamination delay t_{cd}
 - » Y (output) starts to change after the change of A (input)
- propagation delay t_{pd}
 - » Y (output) definitely settles in new value

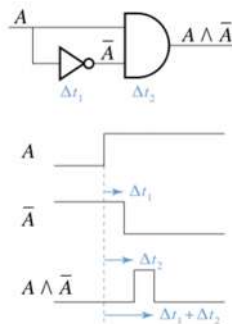


(Figures are from internet)



Example

- In combinational circuit

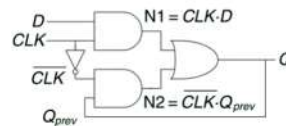




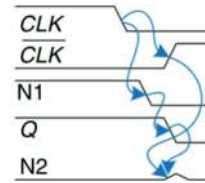
Racing condition

- In asynchronous sequential circuit

$$Q = CLK \cdot D + \overline{CLK} \cdot Q_{prev}$$



- $D=1$
 $CLK=1$
 $Q_{prev}=1 \cdot 1 + 0 \cdot X = 1$
- $CLK=1 \rightarrow 0$
 $Q=0 \cdot 1 + 1 \cdot 1 = 1$



But eventually $Q=0$ because of the race condition

(Figures are from internet)



Rules of Synchronous Sequential Circuit

- Every circuit element is either a **register** or a **combinational** circuit
- At least one circuit element is a **register**
- All registers receive the **same clock signal**
- Every cyclic path contains **at least one register**



Outlines

- Sequential logic
 - » definition
- **Latches and flip-flop**
 - » The evolution of latches and flip-flops
 - » Synchronous and asynchronous circuits
 - » **Example: 4 x 3 memory**
- Finite state machines (FSM)
 - » Turnstile example
 - » Traffic light example

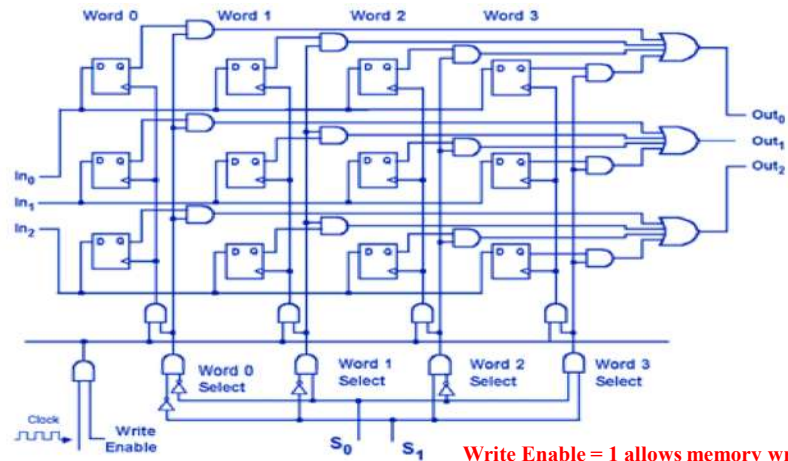


Example: 4 x 3 Memory

- **4 words** and each can store **3 bits** of information
- To represent **4 words** need **2-bits for address**
 - » Address decoder performs the address decoding
- To store information we use **D Flip-Flop** for each bit (total 12, as each location as 3 bits and we have 4 total locations)
- Need select variable to chose if we want to read or write data and that is combined with clock signal (using some combinational logic)
- Also need some other combinational logic...



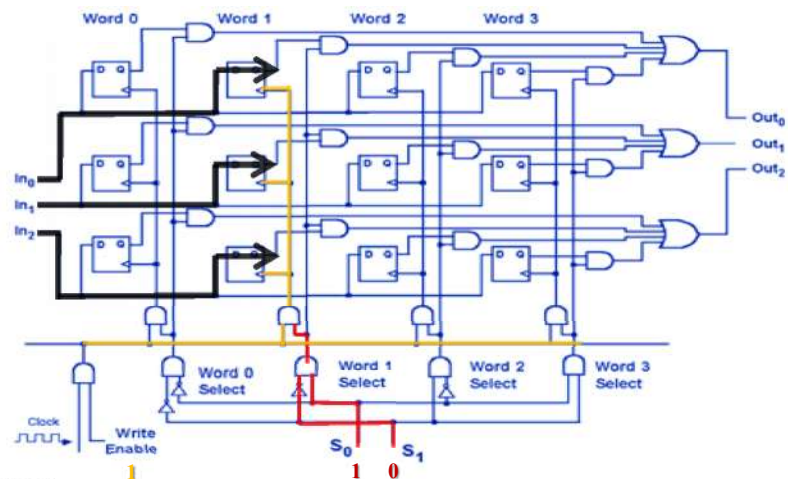
Example: 4 x 3 Memory



(Figures are from internet)



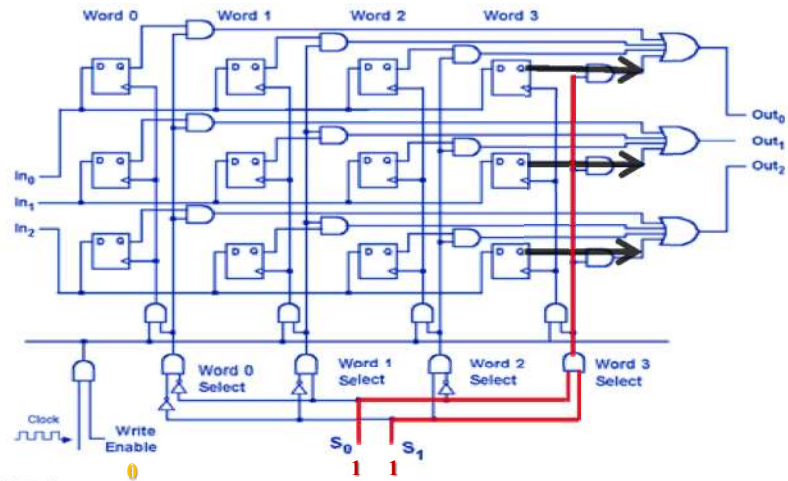
Write Operation to **Word 1**



(Figures are from internet)



Read Operation from **Word 3**



(Figures are from internet)



Outlines

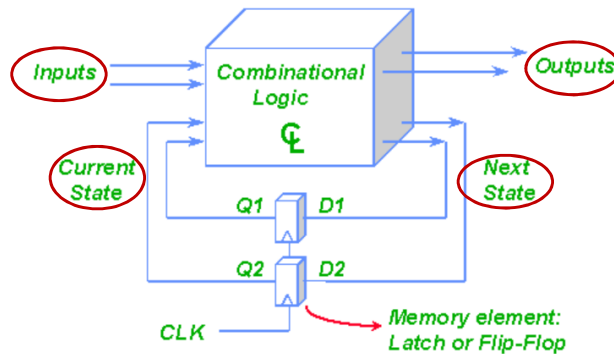
- Sequential logic
 - » definition
- Latches and flip-flop
 - » The evolution of latches and flip-flops
 - » Synchronous and asynchronous circuits
 - » Example: 4 x 3 memory
- **Finite state machines (FSM)**
 - » Turnstile example
 - » Traffic light example

(Figures are from internet)



Finite-State Machine (FSM)

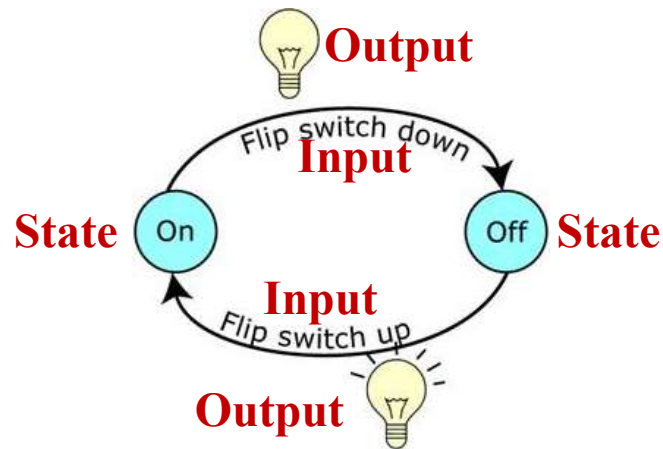
- A **mathematical model of computation** used to design both computer programs and sequential logic circuits



(Figures are from internet)



The Most Seen FSM



(Figures are from internet)



Outlines

- Sequential logic
 - » definition
- Latches and flip-flop
 - » The evolution of latches and flip-flops
 - » Synchronous and asynchronous circuits
 - » Example: 4 x 3 memory
- **Finite state machines (FSM)**
 - » **Turnstile example**
 - » Traffic light example

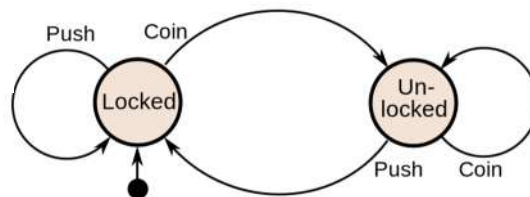


The Turnstile Example



State transition table

Current state S	Inputs		Next state S'
	T _A	T _B	
Locked	Coin		Unlocked
Locked		Push	Locked
Unlocked	Coin		Unlocked
Unlocked		Push	Locked



(Figures are from internet)



State Encoding

Current state S	Inputs		Next state S'
	T _A	T _B	
Locked	Coin		Unlocked
Locked		Push	Locked
Unlocked	Coin		Unlocked
Unlocked		Push	Locked

State encoding

- Locked = 1
- Unlocked = 0

Input encoding

- Coin = 10
- Push = 01

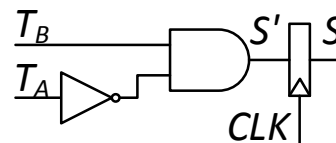
Current state S	Inputs		Next state S'
	T _A	T _B	
1	1	0	0
1	0	1	1
0	1	0	0
0	0	1	1



State Machine Circuit

Current state S	Inputs		Next state S'
	T _A	T _B	
1	1	0	0
1	0	1	1
0	1	0	0
0	0	1	1

$$\begin{aligned}
 S' &= S\bar{T}_AT_B + \bar{S}\bar{T}_AT_B \\
 &= \bar{T}_AT_B
 \end{aligned}$$





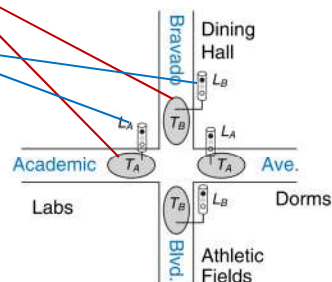
Outlines

- Sequential logic
 - » definition
- Latches and flip-flop
 - » The evolution of latches and flip-flops
 - » Synchronous and asynchronous circuits
 - » Example: 4 x 3 memory
- **Finite state machines (FSM)**
 - » Turnstile example
 - » **Traffic light example**



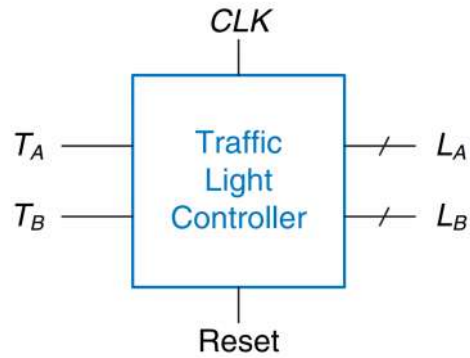
The Traffic Light Controller

- Traffic sensors T_A and T_B
("1" busy, "0" empty)
- Traffic lights L_A and L_B
(each light have red, yellow, and green)
- 5-second clock, at each rising edge, lights may change based on the sensors
- Reset button

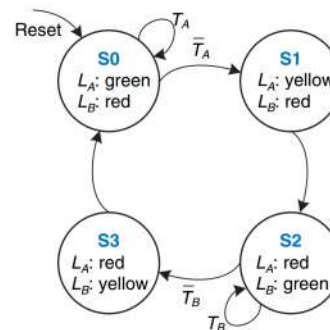
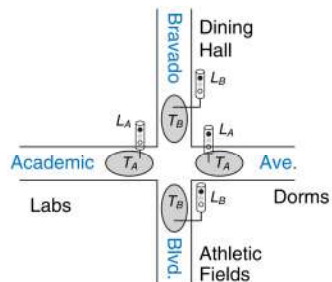





Black Box View



State Transition Diagram



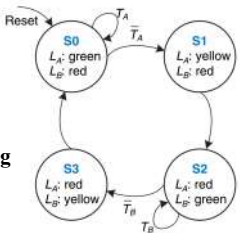


信息科学与技术学院
School of Information Science and Technology

State Table

State encoding

State	Encoding $S_{1:0}$
S0	00
S1	01
S2	10
S3	11



State transition table


Current State S	Input T_A	Input T_B	Next State S(next)
S0	0	X	S1
S0	1	X	S0
S1	X	X	S2
S2	X	0	S3
S2	X	1	S2
S3	X	X	S0

Binary encoded state transition table

Current State S_1 S_0	Inputs T_A T_B	Next State S_1 (next) S_0 (next)
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

Introduction to Information Science and Technology (Electronics)

ShanghaiTech University

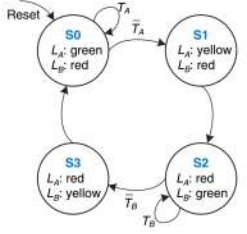


信息科学与技术学院
School of Information Science and Technology

Output Table

Output encoding

Output	Encoding $L_{1:0}$
Green	00
Yellow	01
Red	10



Output table

Current State		Outputs			
S_1	S_0	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

Introduction to Information Science and Technology (Electronics)

ShanghaiTech University



Sum-of-Products Form

State table

Current State		Inputs		Next State	
S_1	S_0	T_A	T_B	S_1 (next)	S_0 (next)
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

$$\begin{aligned}
 S_{1,next} &= \bar{S}_1 S_0 + S_1 \bar{S}_0 T_B + S_1 \bar{S}_0 T_B \\
 &= \bar{S}_1 S_0 + S_1 \bar{S}_0 \\
 &= S_1 \oplus S_0
 \end{aligned}$$

$$S_{0,next} = \bar{S}_1 \bar{S}_0 T_A + S_1 \bar{S}_0 T_B$$

Output table

Current State		Outputs			
S_1	S_0	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

$$L_{A1} = S_1$$

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

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

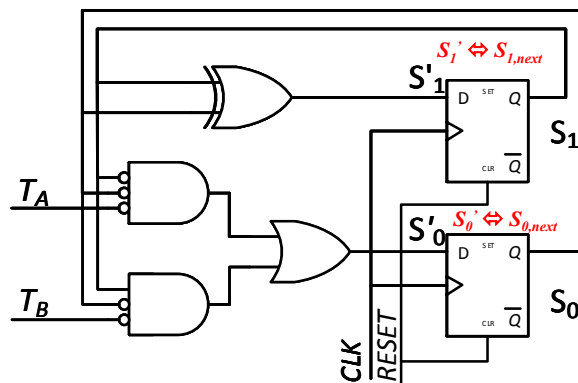
$$L_{B0} = S_1 S_0$$



State Logic

$$S_{1,next} = S_1 \oplus S_0$$

$$S_{0,next} = \bar{S}_1 \bar{S}_0 T_A + S_1 \bar{S}_0 T_B$$

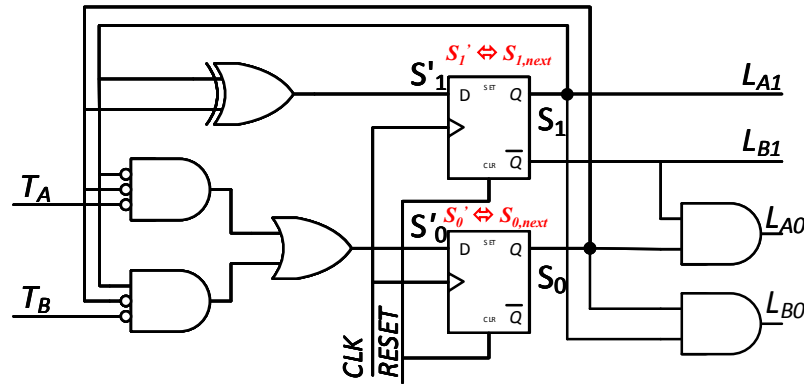




Output Logic

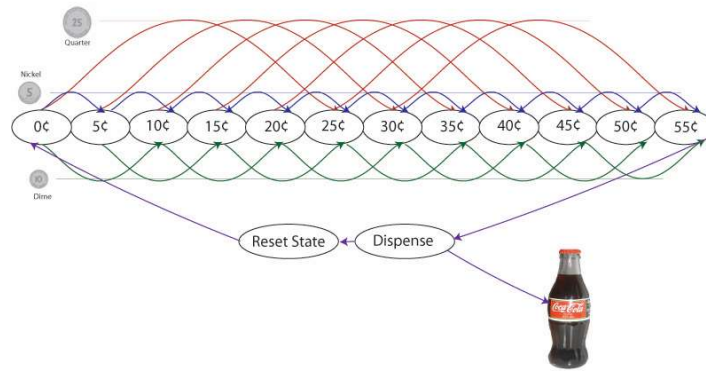
$$L_{A1} = S_1 \quad L_{B1} = \bar{S}_1$$

$$L_{A0} = \bar{S}_1 S_0 \quad L_{B0} = S_1 S_0$$



FSM with More States

Finite State Machine:
Soda Machine State Diagram



(Figures are from internet)



References

- David M. Harris and Sarah L. Harris, [Digital Design and Computer Architecture](#)
- What does J-K in "J-K flip flops" (the electronic circuit) mean? Who named it? <http://www.quora.com/What-does-J-K-in-J-K-flip-flops-the-electronic-circuit-mean-Who-named-it>
- Flip Flop Conversion <http://www.circuitstoday.com/flip-flop-conversion#JKtoD>