Lecture 13: Applications

Professor Peter Cheung

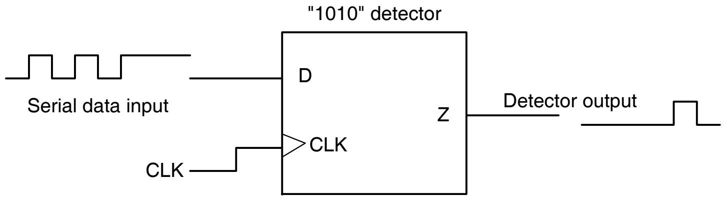
Department of EEE, Imperial College London

Points Addressed in this Lecture

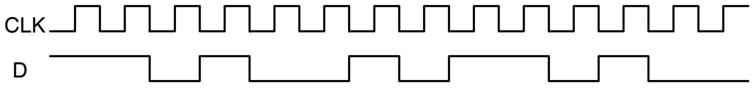
- Design of FSMs
- Delays
- Ring Counter
- Serial Data
- Serial Adder
- Schmitt Trigger

FSM Example - A Sequence Detector

- To detect the occurrence of the binary sequence 1010.

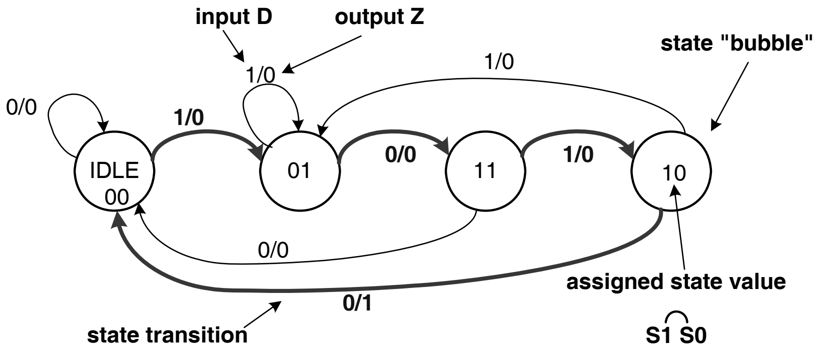


- D input changes on falling edge of CLK, detector changes state on rising edge of CLK.



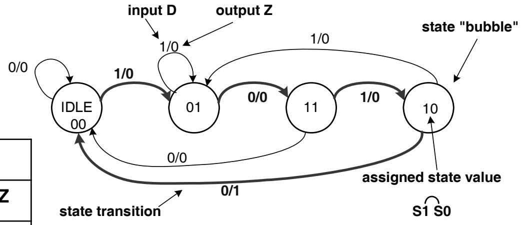
A Sequence Detector (Con't)

- Draw the State Diagram (use Mealy model)



A Sequence Detector (Con’t)

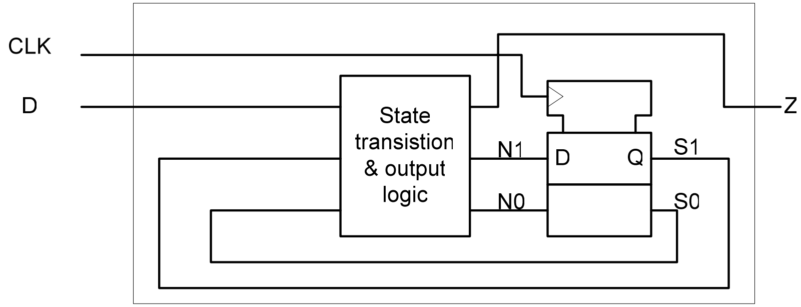
- Draw State Transition Table



	Inputs				Outputs	
S1	S0	D		N1	N0	Z
0	0	0		0	0	0
0	0	1		0	1	0
0	1	0		1	1	0
0	1	1		0	1	0
1	1	0		0	0	0
1	1	1		1	0	0
1	0	0		0	0	1
1	0	1		0	1	0

A Sequence Detector (Con’t)

- Design hardware



A Sequence Detector (Con’t)

- Draw Karnaugh Map

Map for N1

D/ S1:S0	00	01	11	10
0	0	1	0	0
1	0	0	1	0

Map for N0

D/ S1:S0	00	01	11	10
0	0	1	0	0
1	1	1	0	1

Map for Z

D/ S1:S0	00	01	11	10
0	0	0	0	1
1	0	0	0	0

	Inputs				Outputs	
S1	S0	D		N1	N0	Z
0	0	0		0	0	0
0	0	1		0	1	0
0	1	0		1	1	0
0	1	1		0	1	0
1	1	0		0	0	0
1	1	1		1	0	0
1	0	0		0	0	1
1	0	1		0	1	0

A Sequence Detector (Con’t)

- Derive Boolean Equations

Map for N1

D/ S1:S0	00	01	11	10
0	0	1	0	0
1	0	0	1	0

Map for N0

D/ S1:S0	00	01	11	10
0	0	1	0	0
1	1	1	0	1

Map for Z

D/ S1:S0	00	01	11	10
0	0	0	0	1
1	0	0	0	0

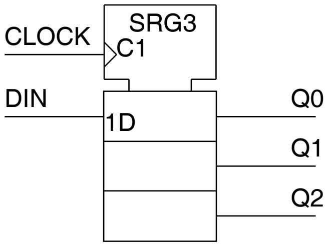
$$N1 = \overline{D} \bullet \overline{S1} \bullet S0 + D \bullet S1 \bullet S0$$

$$N0 = \overline{S1} \bullet S0 + D \bullet \overline{S0}$$

$$Z = \overline{D} \bullet S1 \bullet \overline{S0}$$

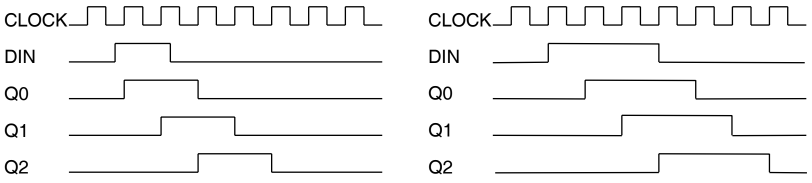
Delays

- A shift register can be used as a delay element
  - Each Q output lags behind the D input by successive numbers of clock periods



- Synchronous Delay
  - The input signal DIN is aligned with the clock
  - The outputs Q0, Q1, Q2 will be delayed by 1,2 and 3 CLOCK periods respectively
- Asynchronous Delay
  - The input signal DIN is NOT aligned with the clock
  - The Q0 output will be delayed by up to 1 clock period
  - Q1 and Q2 will be delayed by further whole clock periods

Timing Diagram of Shift Register



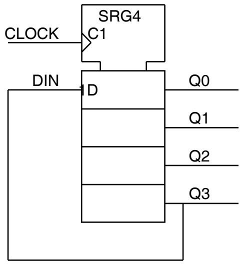
Asynchronous Input

Synchronous Input

Note: These timing diagrams do not show
the CLOCK-to-Q delays of the flip-flops.
In practice, changes in output will follow
slightly later than the rising edge of the CLOCK

Ring Counter

- A ring counter is made from a shift register with the output of the last bit fed to the input of the first bit



- Transition Table
  - The above can be summarised in the following table

Q3	DIN	Q0:3	(Q0:3)+
0	0	n	2n
0	1	n	2n+1
1	0	n	2n-16
1	1	n	2n-15

  - Now reconnect the feedback from Q3 to DIN
  - the operation of the ring counter follows the top and bottom lines of the table only

Serial Adder

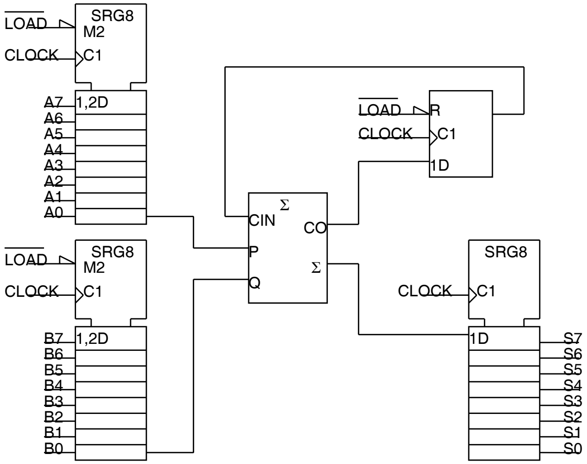
- Bit-serial data
  - Normally when we think of binary numbers represented in digital electronic hardware, we think of parallel data
    - one connection per bit
    - all bits processed together
  - In bit-serial data, the bits making up a binary number are processed one bit at a time
    - takes longer but
    - simplifies the hardware

Eg.

- to add two 8-bit numbers in parallel requires 8 full adders
- to add two 8-bit numbers bit-serially requires only 1 full adder and some registers

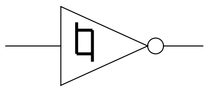
Circuit Diagram for Bit-serial Adder

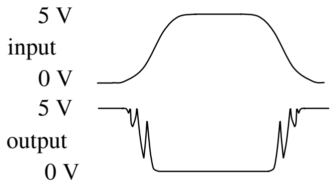
- The 2 numbers to be added (A0:7 and B0:7) are stored in shift registers.
- The bits of A and B are presented to a 1-bit adder one bit at a time starting with the LSB.
- The flip-flop stores the carry from the previous summation and is initialised to 0, ie. CIN=0 for the addition of the LSBs.
- After 8 clock periods the result is stored in the output shift register S0:7.



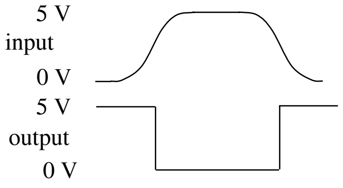
Schmitt Trigger

- Slowly changing input transitions may cause oscillations on the output of digital devices
  - EG: inverter



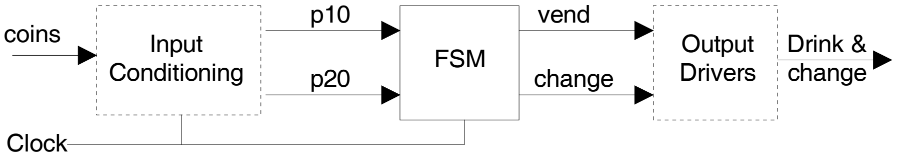


- Schmitt Trigger circuits produce “clean” outputs
  - this is achieve by having different thresholds for low-to-high and high-to-low transitions.
  - known as hysteresis



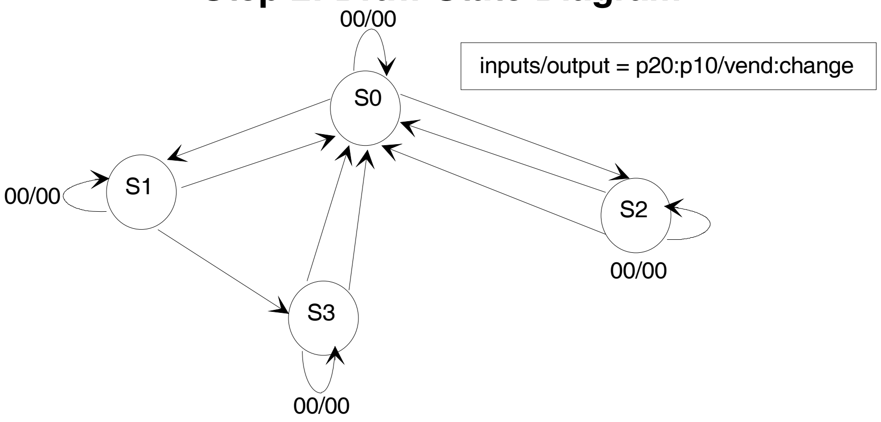
FSM Example – Vending Machine

- Step 1: Define the problem
  - Accepts 10p and 20p coins
  - Delivers a can of drink costing 30p
  - Provide change where appropriate



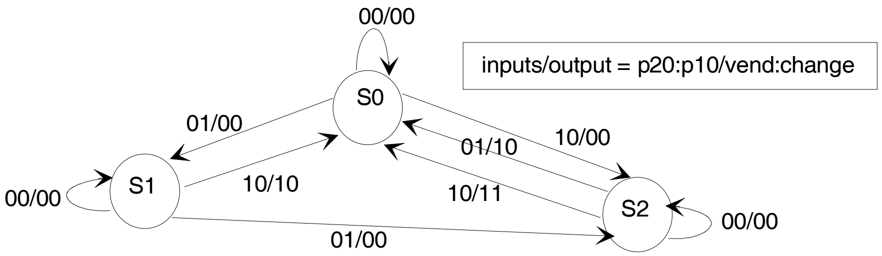
- Assumptions:
  - One coin at a time
  - Generate pulse p10 or p20 lasting for one clock cycle when coin inserted
  - vend = 1 for one clock period to deliver a coke
  - change = 1 for one clock period to return a 10p coin

Step 2: Draw State Diagram



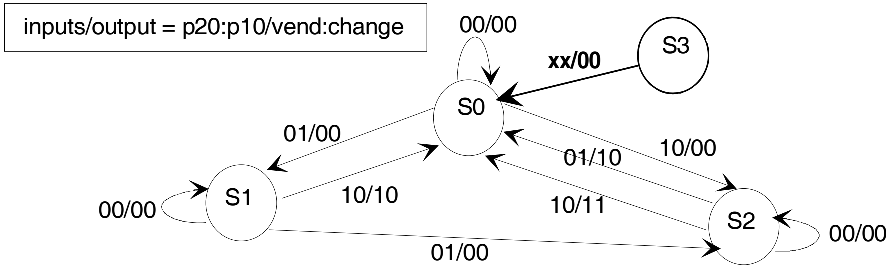
- FSM remains in S0 until there is a *p20* or *p10* input
- S1 represents 10p credit
- S2 represent 20p credit
- S3 represent 40p credit

Step 3: Reduce state diagram



- Two states are said to be **equivalent** if they have
  - identical next states
  - Identical outputs
- Therefore S3≡S2

Step 4: Ensure no undefined states



Step 5: Draw State Transition Table

Next State

Outputs
vend: change

Inputs p20: p10

Current State	00	01	11	10
S0p	S0p,00	S10p,00	xx,xx	S20p,00
S10p	S10p,00	S20p,00	xx,xx	S0p,10
S20p	S20p,00	S0p,10	xx,xx	S0p,11
Sbad	S0p,00	S0p,00	xx,xx	S0p,00

Step 6: Assign binary value to states

S0p = 00

S10p = 01

S20p = 11

Sbad = 10

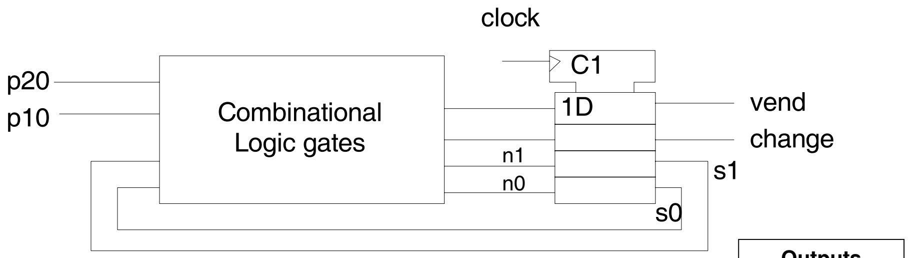
Next State
n1: n0

Outputs
vend: change

Inputs p20: p10

Current State	00	01	11	10
S0p	**00**,00	**01**,00	**xx**,xx	**11**,00
S10p	**01**,00	**11**,00	**xx**,xx	**00**,10
S20p	**11**,00	**00**,10	**xx**,xx	**00**,11
Sbad	**00**,00	**00**,00	**xx**,xx	**00**,00

Step 7: Hardware design



Next State
n1: n0

Outputs
vend: change

Inputs p20: p10

Current State s1:s0	00	01	11	10
00	**00**,00	**01**,00	**xx**,xx	**11**,00
01	**01**,00	**11**,00	**xx**,xx	**00**,10
11	**11**,00	**00**,10	**xx**,xx	**00**,11
10	**00**,00	**00**,00	**xx**,xx	**00**,00

Step 8: Draw Karnaugh Map for each output variable

Inputs p20: p10

Current State s1:s0

n1	00	01	11	10
00	0	0	x	1
01	0	1	x	0
11	1	0	x	0
10	0	0	x	0

Inputs p20: p10

Current State s1:s0

n0	00	01	11	10
00	0	1	x	1
01	1	1	x	0
11	1	0	x	0
10	0	0	x	0

$$n1 = s1\, s0\, \overline{p20}\, \overline{p10} + \overline{s1}\, s0\, p10 + \overline{s1}\, \overline{s0}\, \overline{p20}$$

$$n0 = s0\overline{p20}\, \overline{p10} + \overline{s1}\, p10 + \overline{n1}\, \overline{n0}\, p20$$

Current State s1:s0 / Inputs p20: p10

vend	00	01	11	10
00	0	0	x	0
01	0	0	x	1
11	0	1	x	1
10	0	0	x	0

$$vend = s1\ s0\ p10 + s0\,p20$$

Current State s1:s0 / Inputs p20: p10

change	00	01	11	10
00	0	0	x	0
01	0	0	x	0
11	0	0	x	1
10	0	0	x	0

$$change = s1\ \overline{s0}\ p20$$