Computer Architecture and Organization

Lecture 2

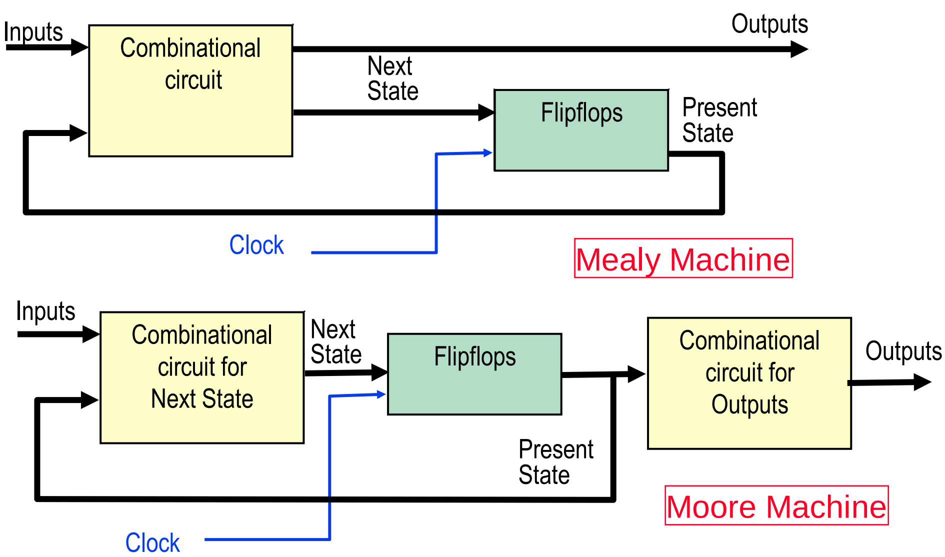
Finite State Machines
(Moore/Mealy)

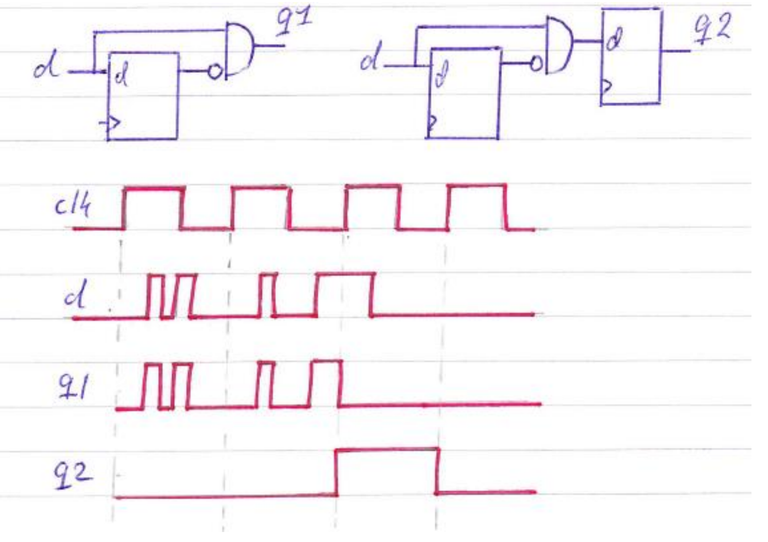
Bert Molenkamp

Learning objectives:

- **You can design synchronous sequential logic using FSM's**
- **You can analyze synchronous sequential logic**

Model of Moore and Mealy FSM





Synthesis of an FSM 1

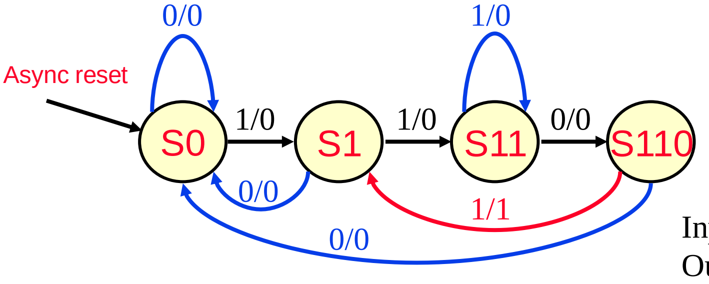
1. **Create from a given description in words (informal specification) a state diagram (formal specification)**
   - This is the most important and most difficult step, which has to done by the designer "by hand“.
   - For the exam, students should also be able to design simple state machines by hand.

Synthesis of an FSM 2

2. Simplify if necessary the state diagram (not part of this course)
3. Create a not encoded two-dimensional state table
4. Choose number of flip-flops, encode the states and create a two-dimensional state table
5. Create a one-dimensional state table
6. Choose type of flip-flops (D flip-flops; EE students other type of flip-flops in “Digital Hardware”)
7. Determine equations for the inputs of the flip-flop
8. Determine equations for the outputs of the system
9. Draw schematic

Example 1. state diagram

- Input x, output z
- z=1 when last three inputs of x are 110 and the current input of x=1
- Asynchronous reset to the state that no ones are detected

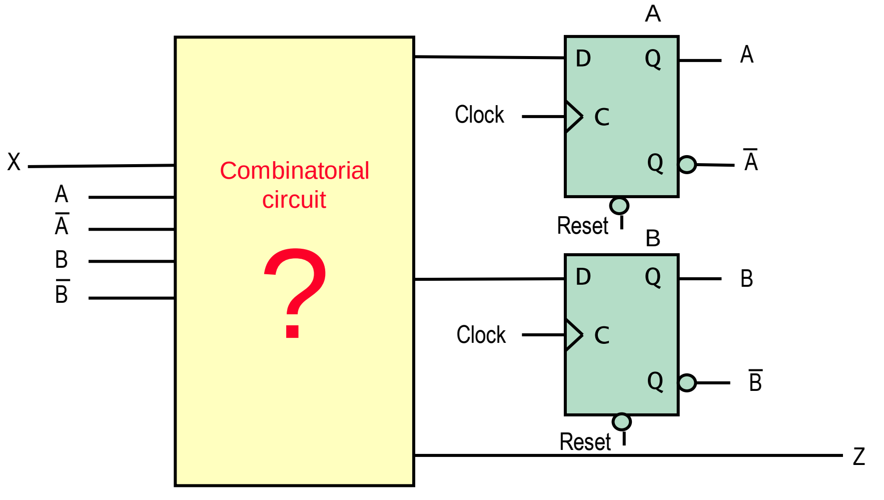


Use meaningful names for the states (in stead of A, B, C and D)

Input: x
Output: z
At edge x/z

Realization

Find a minimal SOP for combinational circuit and 2 flip-flops

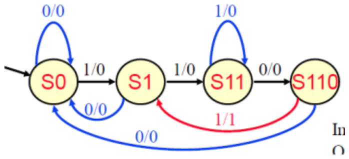


2. Simplify state diagram

- **There are systematic methods to simplify the state machine to a minimum number of states, these methods are beyond the scope of this course**
- **The given state machine is minimal**

3. Not encoded state table

The initial state is not explicit marked in this table



Present State	Next State		Output	
	$\overline{X}$	X	$\overline{X}$	X
			Z	Z
S0	S0	S1	0	0
S1	S0	S11	0	0
S11	S110	S11	0	0
S110	S0	S1	0	1

4. Encoding of the states (1)

- **Any encoding, as long as the states are different encoded always work, however encoding has an influence on the size and speed of the system**
- **For the encoding of *m* states at least *n* flip-flops with $2^n \geq m$ are required. The minimal number of flip-flops does not always result in the smallest circuit**
- **Finding the minimal encoding in reasonable time, especially for large state machines, is a very difficult task for which no algorithms exist**
- **Rules of thumb (heuristics) are often used.**
- **Finding an optimal encoding is not part of this course**

4. Encoding of the states (2)

° **Example: choose minimum number of flip-flops (A and B) with arbitrary encoding S0→00, S1→01, S11→11, S110→10**

Present State	Next State $\bar{X}$	Next State X	Output $\bar{X}$ Z	Output X Z
S0	S0	S1	0	0
S1	S0	S11	0	0
S11	S110	S11	0	0
S110	S0	S1	0	1

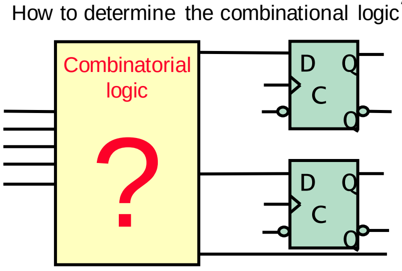
Present State		Next State				Output	
		$\bar{X}$		X		$\bar{X}$	X
A	B	A+	B+	A+	B+	Z	Z
0	0	0	0	0	1	0	0
0	1	0	0	1	1	0	0
1	1	1	0	1	1	0	0
1	0	0	0	0	1	0	1

5. one-dimensional state table

Copy info from two-dimensional state table

Present State		In-put	Next State		Out-put
A	B	X	A+	B+	Z
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	0
0	1	1	1	1	0
1	0	0	0	0	0
1	0	1	0	1	1
1	1	0	1	0	0
1	1	1	1	1	0

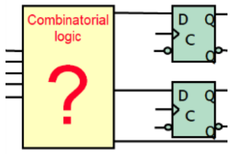
How to determine the combinational logic?



6. Choose type of flip-flop

- Choose the type of flip-flop; most used are **D flip-flops**
- Add to table columns for the **inputs of the flip-flops**
- For D-type flip-flops, this is simple: **D-inputs** of the flip-flops are the **same as** the **desired Next State** of that flip-flop

Present State		In-put	Next State		Inputs flipflops		Out-put
A	B	X	A+	B+	D_A	D_B	Z
0	0	0	0	0	0	0	0
0	0	1	0	1	0	1	0
0	1	0	0	0	0	0	0
0	1	1	1	1	1	1	0
1	0	0	0	0	0	0	0
1	0	1	0	1	0	1	1
1	1	0	1	0	1	0	0
1	1	1	1	1	1	1	0



Same for D flip-flop

7/8. equations

D_A:

		B	B	
	0	0	1	0
A	0	0	1	1
		X	X	

D_B:

		B	B	
	0	1	1	0
A	0	1	1	0
		X	X	

Z:

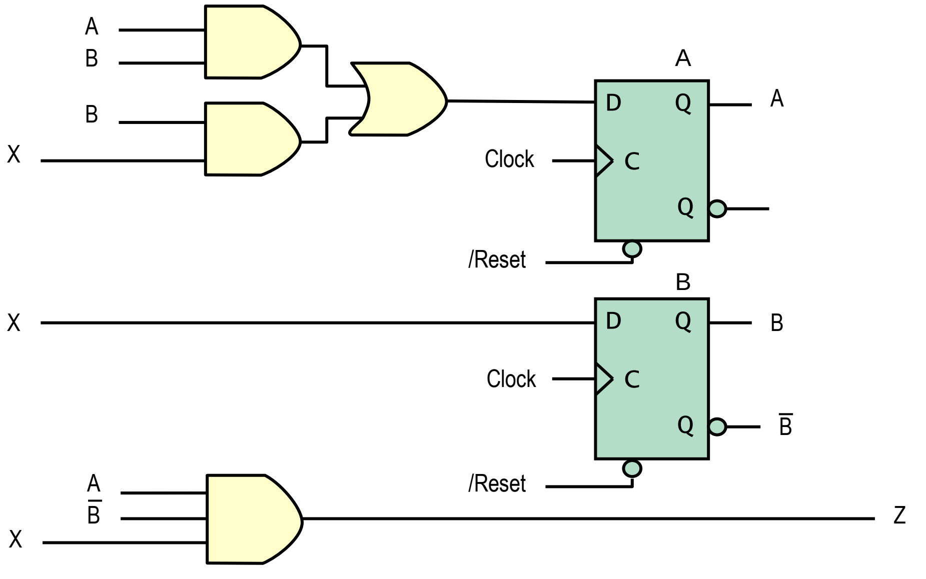
		B	B	
	0	0	0	0
A	0	1	0	0
		X	X	

$$D_A = A \bullet B + B \bullet X$$

$$D_B = X$$

$$Z = A \bullet \overline{B} \bullet X$$

9. Draw schematic



Remarks

° If another state encoding is used the circuit is different but the behavior is the same.

° The signal names `Clock` and the asynchronous `reset` are not in the table

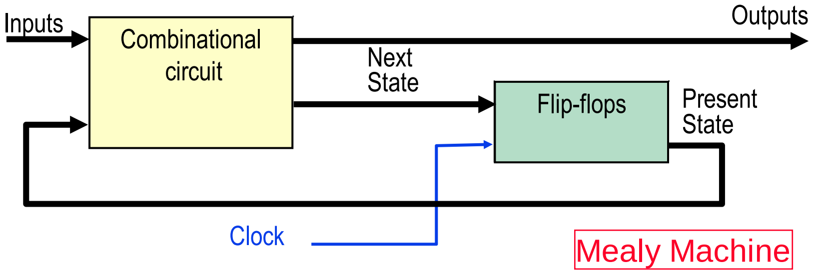
Unused states

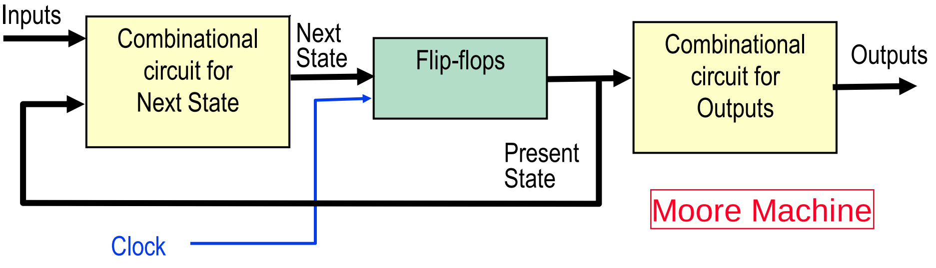
- If number of states is not a power of 2, there are unused states
- Due to a fault or power on the system be in an unused state.
- There may or may not be something prescribed for these unused states

Initial state

- After switching on the power the circuit can be in an arbitrary state.
- With an asynchronous reset/preset the circuit is forced in a defined initial state
- The initial state or Reset State is often marked with an arrow “from nowhere”

Model Mealy and Moore





Mealy ←→ Moore

Specification of a Mealy machine

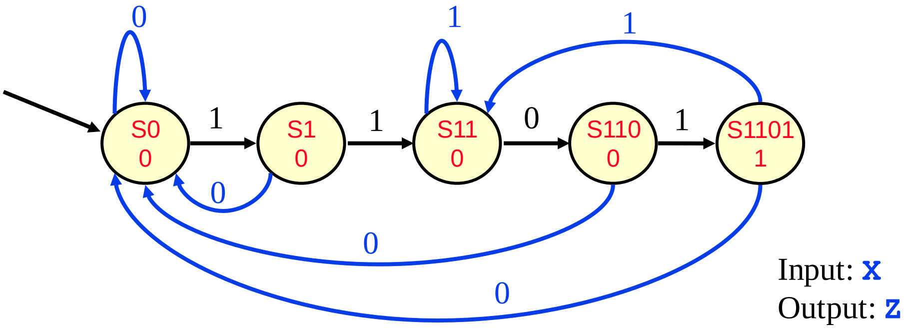
- Input **X**, output **Z**
- **Z=1** when last three inputs of **X** are **110** and the current input of **X=1**
- Asynchronous reset to the state that no ones are detected

Specification of a Moore machine

- Input **X**, output **Z**
- **Z=1** when last four inputs of **X** are **1101**
- Asynchronous reset to the state that no ones are detected

State diagram of a Moore machine

Output in the states.



Example: two-dimensional state table (Moore)

Present State	Next State $\overline{X}$	Next State X	Output Z
S0	S0	S1	0
S1	S0	S11	0
S11	S110	S11	0
S110	S0	S1101	0
S1101	S0	S11	1

1 output column

Encoding of the states

Present State	Next State		Out-put
	$\overline{X}$	X	
			Z
S0	S0	S1	0
S1	S0	S11	0
S11	S110	S11	0
S110	S0	S1101	0
S1101	S0	S11	1

	F2 F1 F0
S0	0 0 0
S1	0 0 1
S11	0 1 0
S110	0 1 1
S1101	1 0 0

Realize this with D flip-flops.

Analysis - Synthesis

Schematic

Equations

One-dimensional state table

Two-dimensional state table

not encoded state table

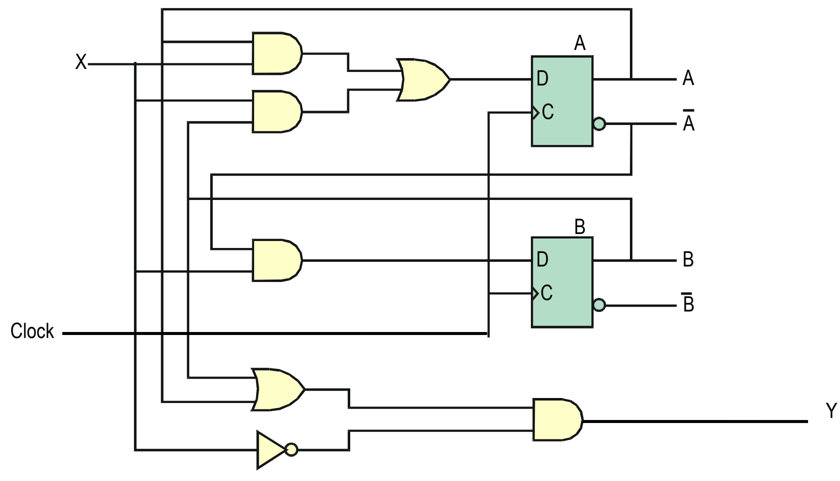
Description in words

Analysis

Synthesis

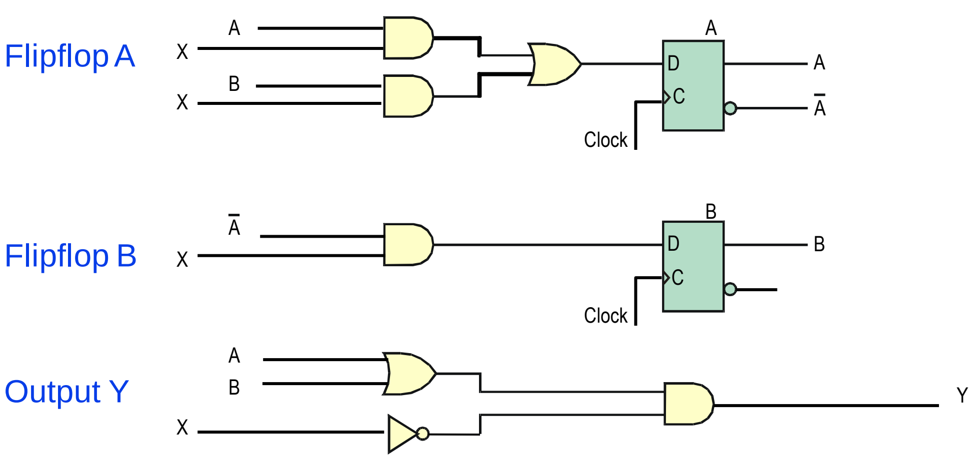
Analysis of a synchronous sequential circuit

System has input X and output Y; what is the behaviour?



Alternative schematic

Schematics are often more clear when not all connections are drawn



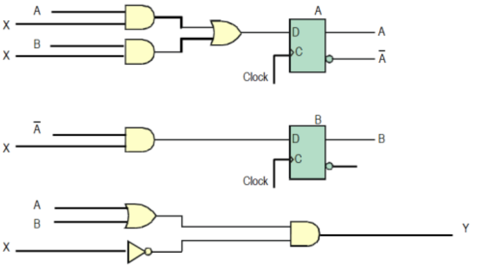
Inputs of flip-flops and outputs of the system

Determine the input equations of the flip-flops and the output equations of the system

$$D_A = A \cdot X + B \cdot X$$

$$D_B = \overline{A} \cdot X$$

$$Y = (A + B) \cdot \overline{X}$$



One-dimensional state table

Create a one-dimensional state table with a row for each combination of state variabels and inputs

- **The Next State column is equal to D (a D flip-flop is used)**

Present State		Input	Next State		Output
A	B	X	A+	B+	Y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	0

$$D_A = A \cdot X + B \cdot X$$

$$D_B = \overline{A} \cdot X$$

$$Y = (A + B) \cdot \overline{X}$$

Two-dimensional state table

Create a two-dimensional state table with a row for each state and for each input combination separate columns for the Next State and Output

- **The information is exactly the same as the one-dimensional state table**

S0=00
S1=01
S2=10
S3=11

Present State		Next State				Output	
		$\overline{X}$		X		$\overline{X}$	X
A	B	A+	B+	A+	B+	Y	Y
0	0	0	0	0	1	0	0
0	1	0	0	1	1	1	0
1	0	0	0	1	0	1	0
1	1	0	0	1	0	1	0

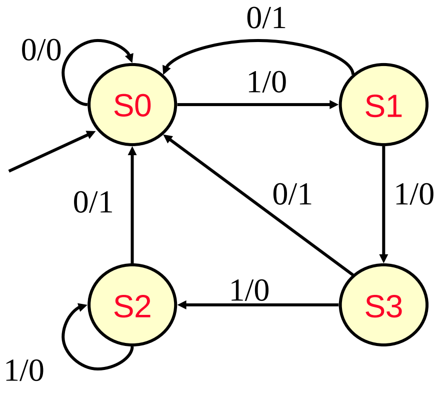
State table with not encoded states

Replace the bit patterns of the state by freely chosen symbolic names

Present State	Next State		Output	
	$\overline{X}$	X	$\overline{X}$	X
			Y	Y
S0	S0	S1	0	0
S1	S0	S3	1	0
S2	S0	S2	1	0
S3	S0	S2	1	0

State diagram

Draw FSM



Input: **X**

Output: **Y**

Initial state: **S0**

Behaviour in words

° **Circuit has input X and output Y**

° **Y=1 when X=1 in the previous clock period and the current value of X=0**

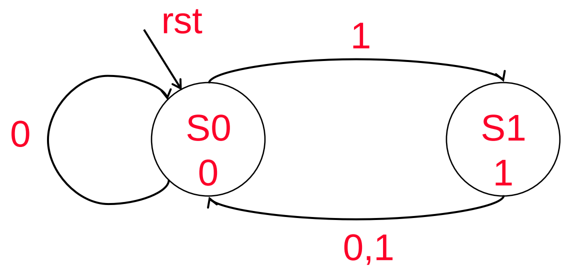
Note

Given circuit is not a minimum solution for this behavior, because the states S2 and S3 are not necessary!

° What is the simplified state machine?

° And how is it realized?

Give a synchronous sequential logic for this FSM



- Input: X
- Output: Y
- Asynchronous low active reset RST

- **What did you learn:**
  - You can design synchronous sequential logic using FSM's
  - You can analyze synchronous sequential logic
- **Next lecture:**
  - Number representations