Task 2

Digital Electronics & Processors

Team 12

Abdulrahman Shawky	sec: 1	B.N: 34
Ahmed Kamal	sec: 1	B.N: 4
Amgad Atef	sec: 1	B.N: 9
Abdallah Mohammed	sec: 1	B.N: 38

Under the supervision of:

Dr. Mohamed Islam

Eng. Samar Taher

➢ Idea:

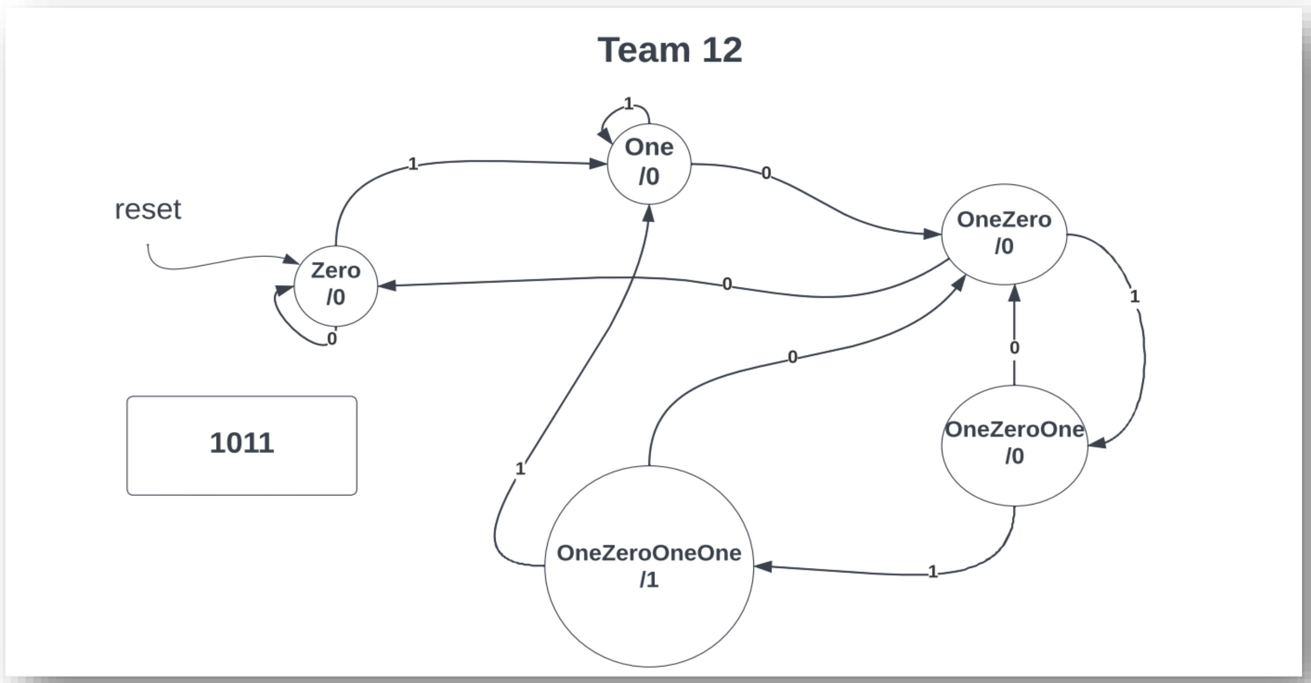
Our project is a sequence detector that detects a specific sequence such as 1011 and that is used in many applications such as:

- **Security and Cryptography:** Sequence detectors play a crucial role in security and cryptography applications. They can be used to detect specific sequences or patterns in cryptographic algorithms, ensuring the integrity and authenticity of encrypted data. Sequence detectors are also employed in intrusion detection systems to identify malicious patterns or suspicious activities.
- **Communication Protocols:** Sequence detectors are commonly used in communication protocols to identify specific sequences of bits or symbols. For example, in serial communication protocols like UART or SPI, a sequence detector can be used to detect start and stop bits or specific command sequences.
- **Data Parsing and Processing:** Sequence detectors are employed in data parsing and processing tasks. They can be used to identify specific patterns or sequences in a data stream, enabling efficient extraction and manipulation of relevant data. This can be useful in applications such as data compression, error detection, or protocol parsing.
- **Pattern Recognition:** Sequence detectors find applications in pattern recognition tasks, where specific patterns or sequences need to be identified in input data. This can be used in image or speech recognition systems, where sequence detectors help recognize specific patterns or sequences of features or phonemes.
- **Control Systems:** Sequence detectors are utilized in control systems to detect specific sequences of events or states. They can be employed to trigger specific actions or transitions in the control

system based on the detected sequences. This can be useful in robotics, automation, or process control applications.

- **Signal Processing:** Sequence detectors are used in various signal processing applications. For instance, in audio or video processing, sequence detectors can be employed to detect specific patterns or sequences of signals, enabling features like voice or image recognition, data compression, or noise filtering.

➢ State diagram:



We detect the sequence **1011** or any other sequence we want to detect,

- First, we feed the reset input with logic one to start from the state **Zero**.
- Now, the current state is **Zero** as the input sequence is still **0**.
- Once, the input sequence becomes logic one, the next state changes into **One**.

- By using the same previous procedure, we monitor the value of the input sequence and the current state for every clock cycle.
- if the input sequence is **1** and the current state is **One**, the next state still at **One.** Otherwise, the next state changes to become **OneZero**.
- As the clock moves to the next cycle, if the input sequence is **1** and the current state is **OneZero**, the next state changes to become **OneZeroOne.** Otherwise, the next state goes back to a previous state of **Zero**.
- As we can see the next state has not been **1011** yet, the previous criteria will continue until the input sequence is **1** and the current state becomes **OneZeroOneOne**.
- Once, the current state becomes **1011**, the output will be **high**.

Basically, in case of detecting a new sequence and starting a new loop, all we can do, we activate the input reset to become high, and all parameters of the system will be initialized to zero.

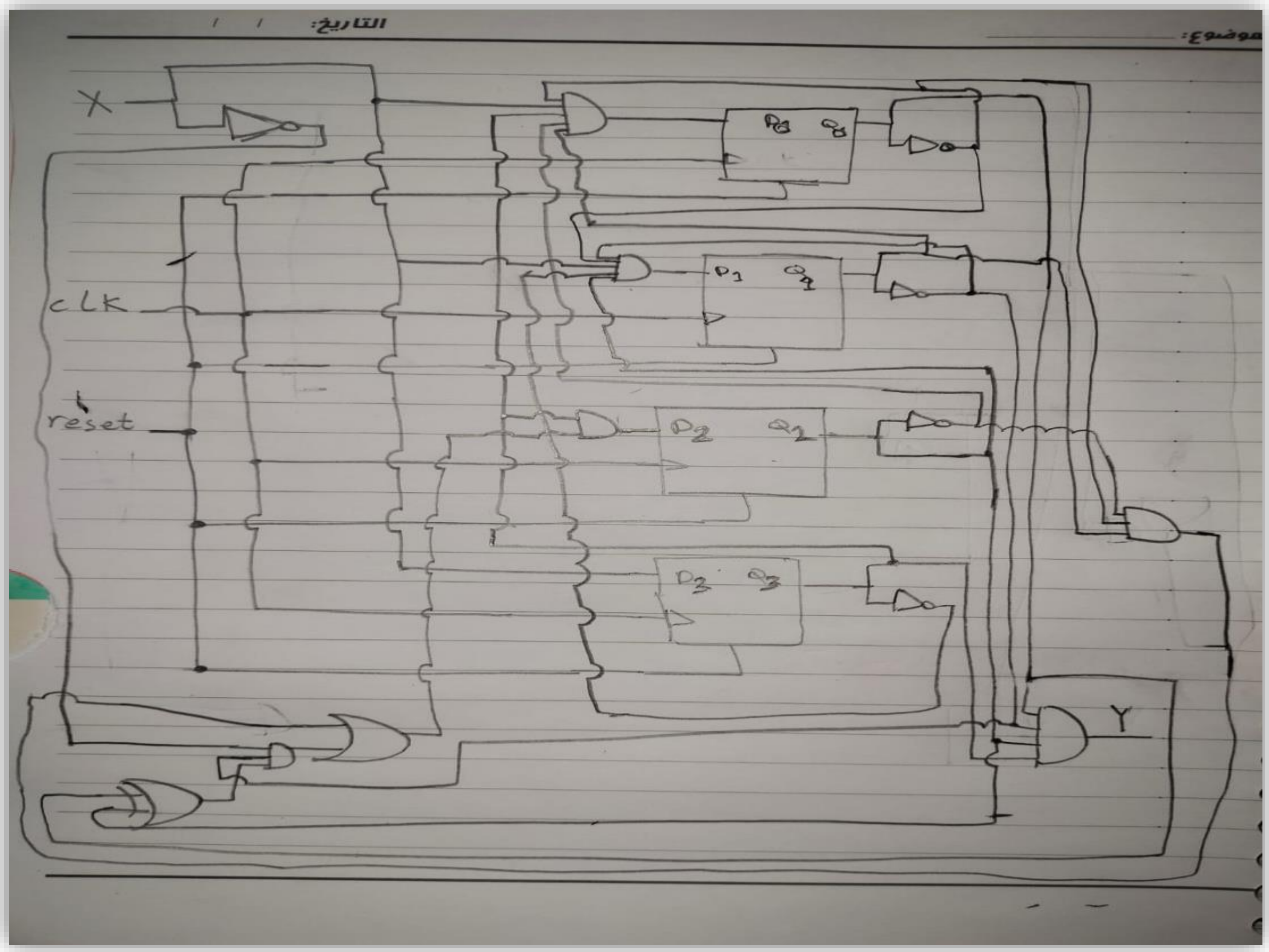
➢ State table:

Present state	Next state		Output	
	X = 0	X = 1	X = 0	X = 1
a (Zero)	a	b	0	0
b (One)	c	b	0	0
c (OneZero)	a	d	0	0
d (OneZeroOne)	c	e	0	1
e (OneZeroOneOne)	c	b	0	0

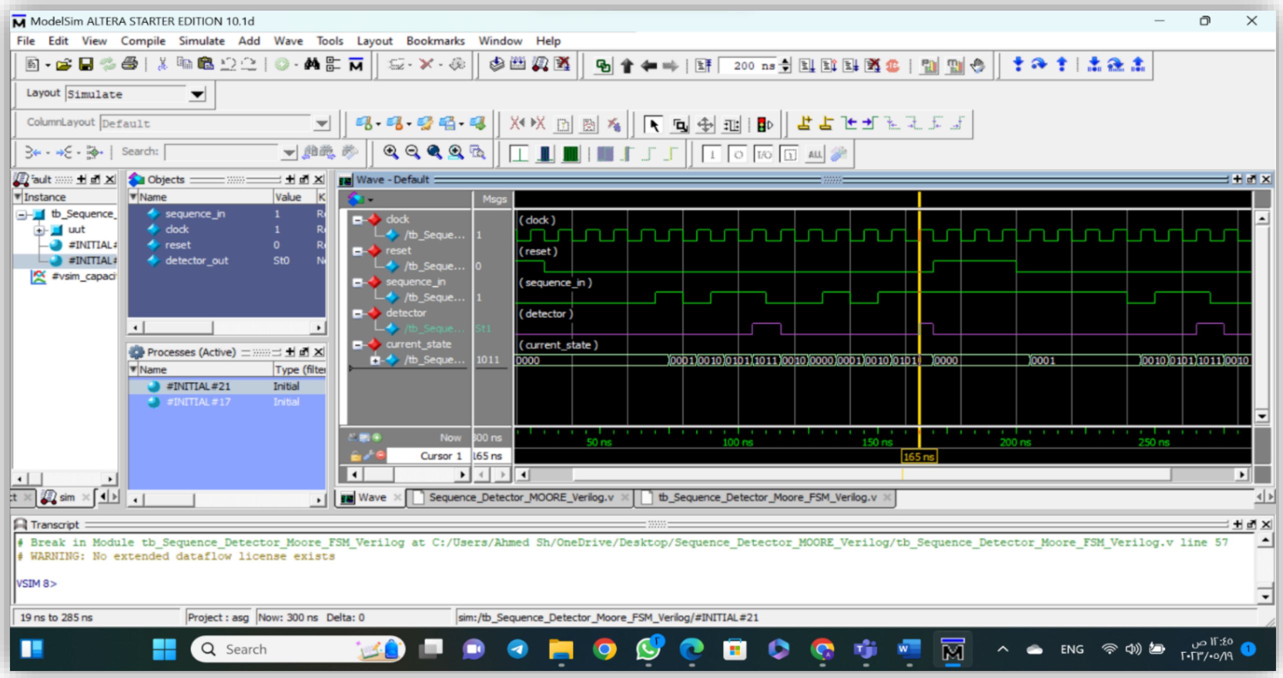
➢Excitation table:

reset	clk	X	D0	D1	D2	D3		D0+1	D1+1	D2+1	D3+1		Y
1	↑	x	x	x	x	x		0	0	0	0	a	0
0	↑	0	0	0	0	0	a	0	0	0	0	a	0
0	↑	0	0	0	0	1	b	0	0	1	0	c	0
0	↑	0	0	0	1	0	c	0	0	0	0	a	0
0	↑	0	0	1	0	1	d	0	0	1	0	c	0
0	↑	0	1	0	1	1	e	0	0	1	0	c	1
0	↑	1	0	0	0	0	a	0	0	0	1	b	0
0	↑	1	0	0	0	1	b	0	0	0	1	b	0
0	↑	1	0	0	1	0	c	0	1	0	1	d	0
0	↑	1	0	1	0	1	d	1	0	1	1	e	0
0	↑	1	1	0	1	1	e	0	0	0	1	b	1

➢ Circuit diagram:



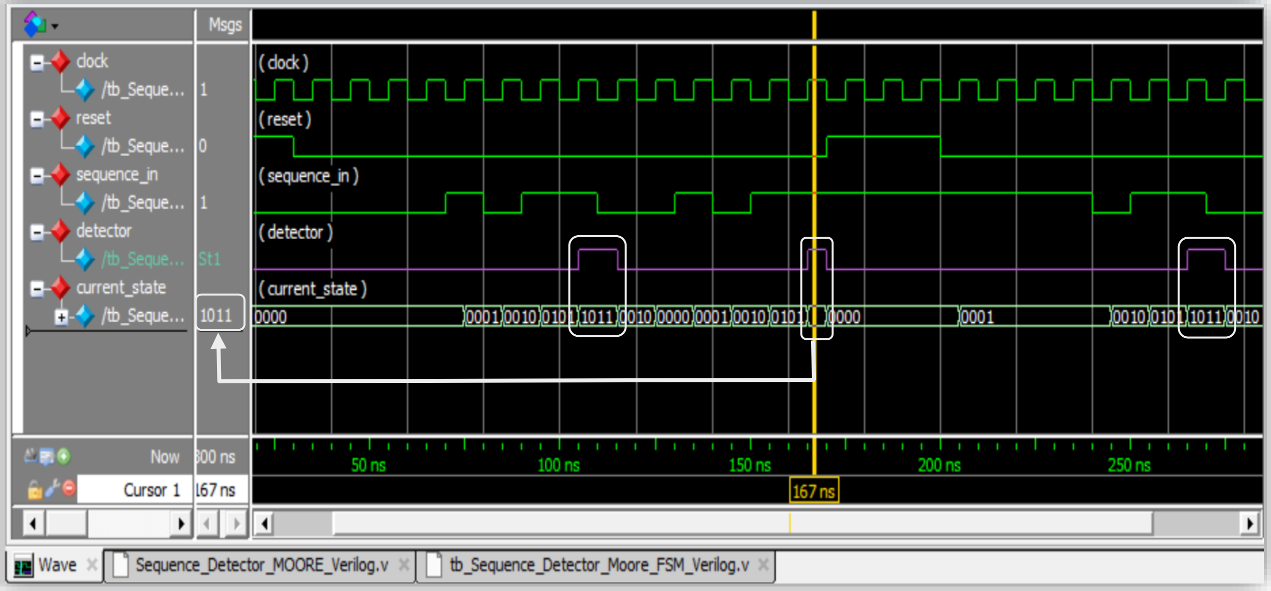
• Output diagram:



```
# Break in Module tb_Sequence_Detector_Moore_FSM_Verilog at C:/Users/Ahmed Sh/OneDrive/Desktop/Sequence_Detector_MOORE_Verilog/tb_Sequence_Detector_Moore_FSM_Verilog.v line 57
# WARNING: No extended dataflow license exists

VSIM 8>
```

ModelSim ALTERA STARTER EDITION 10.1d
File Edit View Compile Simulate Add Wave Tools Layout Bookmarks Window Help
Layout Simulate
ColumnLayout Default
Objects
Processes (Active)
Wave - Default
Transcript
19 ns to 285 ns
Project : asg
Now: 300 ns Delta: 0
sim:/tb_Sequence_Detector_Moore_FSM_Verilog/#INITIAL#21
Wave
Sequence_Detector_MOORE_Verilog.v
tb_Sequence_Detector_Moore_FSM_Verilog.v
Cursor 1
165 ns



We notice from the figures above that there are five signals which indicate the **clock** (input square wave), the **reset** signal that initialize the other signals to zero, the **sequence_in**, the **detector**, and the **current_state**.

As we can see from the above figure that the output will be an active once the current state becomes the same bits of the desired signal we are detecting.

The sequence in and the detector follow the procedure discussed in the above section of the state diagram.

➢Challenges we faced and how we overcame them:

1. Learning and using Verilog for the first time.
2. Algorithm and combination of states.
3. Complexity of selecting appropriate number of bits to be detected which falls in the range of four bits.
4. As the number of states, inputs, and outputs in the system increases, the complexity of fsm also increases. This can make it a bit difficult to design, analyze, and debug the system.