# **Experiment** **7**

# **Sequential Circuit Design**

**Basic Concepts**

In combinational circuits the outputs at any instant of time are entirely dependent upon the inputs present at that time. In sequential circuits the outputs depend on the inputs and the state of the sequential circuits which depend on the memory elements of the circuits. Figure 1 is a block diagram of a sequential circuit. The storage elements are circuits that are capable of storing binary information. The binary information stored in these elements at any given time defines the *state* of the sequential circuit at that time. The block diagram demonstrates that the next state of storage elements is a function of the inputs and the present state, and the output of the sequential circuit is a function of the present state of storage elements and (sometimes) the inputs.

**Combinational Logic**

**Inputs**

**Outputs**

**Storage Elements**

**Next State**

**State**

Figure 1: Block Diagram of a Sequential Circuit

**Sequential Circuit design:**

A synchronous sequential circuit is made up of flip-flops and combinational gates. The design of the circuit consists of choosing the flip-flops and finding a combinational circuit structure which, together with the flip-flops, produces a circuit that fulfills the stated specifications. The combinational circuit is derived from the state table by finding the flip-flop input equations and output equations.

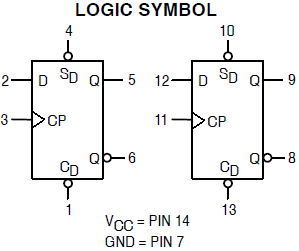
**Design Procedure:**

The following procedure for the design of sequential circuits is similar to that for combinational circuits but has some additional steps:

* Specification
* Formulation - Obtain a state diagram or state table
* State Assignment - Assign binary codes to the states
* Flip-Flop Input Equation Determination - Select flip-flop types and derive flip-flop equations from next state entries in the table
* Output Equation Determination - Derive output equations from output entries in the table
* Optimization - Optimize the equations
* Technology Mapping - Find circuit from equations and map to flip-flops and gate technology
* Verification - Verify correctness of final design

**DualD-Type PositiveEdge-Triggered Flip-Flop:**

The 74LS74A offers dual edge-triggered flip-flops. Each flip-flop has individual clear and set inputs, and also complementary Q and Q outputs. Information at input D is transferred to the Q output on the positive-going edge of the clock pulse. The logic symbol and truth table of 74LS74A is shown in Figure 2.



**TRUTH TABLE**

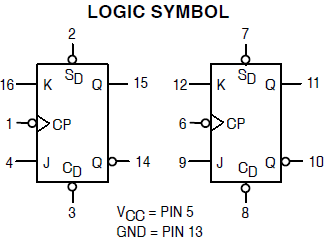
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SD'** | **CD'** | **D** | **Q** | **Q'** | **Description** |
| **L** | **H** | **X** | **H** | **L** | **Asynchronously Set** |
| **H** | **L** | **X** | **L** | **H** | **Asynchronously Clear** |
| **L** | **L** | **X** | **H** | **H** | **Undetermined** |
| **H** | **H** | **H** | **H** | **L** | **Load 1 (Set)** |
| **H** | **H** | **L** | **L** | **H** | **Load 0 (Reset)** |

Figure 2: 74LS74A

**Dual JK Flip-Flop With Set and Clear:**

The 74LS76A offers individual J, K, Clock Pulse, Direct Set and Direct Clear inputs. These dual flip-flops are designed so that when the clock goes HIGH, the inputs are enabled and data will be accepted. The Logic Level of the J and K inputs will perform according to the Truth Table as long as minimum set-up times are observed. Input data is transferred to the outputs on the HIGH-to-LOW clock transitions. The logic symbol and truth table of 74LS76A is shown in Figure 9.

**TRUTH TABLE**

****

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SD'** | **CD'** | **J** | **K** | **Q** | **Q'** | **Description** |
| **L** | **H** | **X** | **X** | **H** | **L** | **Asynchronously Set** |
| **H** | **L** | **X** | **X** | **L** | **H** | **Asynchronously Clear** |
| **L** | **L** | **X** | **X** | **H** | **H** | **Undetermined** |
| **H** | **H** | **L** | **L** | **Q** | **Q'** | **No Change** |
| **H** | **H** | **L** | **H** | **L** | **H** | **Load 0 (Reset)** |
| **H** | **H** | **H** | **L** | **H** | **L** | **Load 1 (Set)** |
| **H** | **H** | **H** | **H** | **Q'** | **Q** | **Complement** |

Figure 9: 74LS76A

**Procedure**

**Part 1 (Workbench and Implementation):**

You are requested to design and implement a sequential circuit with two D flip-flops A and B and one input X. When X = 0, the state of the circuit remains the same. When X = 1, the circuit goes through the state transitions from 00 to 10 to 11 to 01, back to 00, and then repeat.

**State Table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Present State** | | **Inputs** | **Next State** | |
| ***A*** | ***B*** | ***X*** | ***A*** | ***B*** |
| 0 | 0 | 0 |  |  |
| 0 | 0 | 1 |  |  |
| 0 | 1 | 0 |  |  |
| 0 | 1 | 1 |  |  |
| 1 | 0 | 0 |  |  |
| 1 | 0 | 1 |  |  |
| 1 | 1 | 0 |  |  |
| 1 | 1 | 1 |  |  |

**Part 2 (Workbench and Implementation):**

You are requested to design and implement a sequential circuit that recognizes a subsequence **1 1 0 1**. The sequential machine accepts one input (*X*) which represents the received serial data. When the machine recognizes the subsequence, the output (*Z*) is set to one. Assume the following:

1. The output equals 0 for all other states.
2. The subsequences can be overlapped.
3. Gray-code state assignment.
4. D-type positive edge-triggered flip-flops are used.

**State Diagram:**

**State Table:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Present State** | | | **Input** | **Next State** | | **Output** |
|  | ***Q1*** | ***Q2*** | ***X*** | ***Q1*** | ***Q2*** | ***Z*** |
| **A** | 0 | 0 | 0 |  |  |  |
| 0 | 0 | 1 |  |  |  |
| **B** | 0 | 1 | 0 |  |  |  |
| 0 | 1 | 1 |  |  |  |
| **C** | 1 | 1 | 0 |  |  |  |
| 1 | 1 | 1 |  |  |  |
| **D** | 1 | 0 | 0 |  |  |  |
| 1 | 0 | 1 |  |  |  |

**K-Map:**

0

1

00 01 11 10

*D*1 =

Q

1

Q X

2

0

1

00 01 11 10

*D*2 =

Q

1

Q X

2

0

1

00 01 11 10

*Z* =

Q

1

Q X

2

**Part3 (Workbench and Implementation):**

You are requested to design and implement a sequential circuit that recognizes a subsequence **1 1 0 1**. The sequential machine accepts one input (*X*) which represents the received serial data. When the machine recognizes the subsequence, the output (*Z*) is set to one. Assume the following:

1. The output equals 0 for all other states.
2. The subsequences can be overlapped.
3. Gray-code state assignment.
4. JK positive edge-triggered flip-flops are used.

**State Diagram:**

**State Table:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Present State** | | | **Input** | **Next State** | | **Output** | **Inputs of JK flip-flops** | | | |
|  | ***Q1*** | ***Q2*** | ***X*** | ***Q1*** | ***Q2*** | ***Z*** | ***JQ1*** | ***KQ1*** | ***JQ2*** | ***KQ2*** |
| **A** | 0 | 0 | 0 |  |  |  |  |  |  |  |
| 0 | 0 | 1 |  |  |  |  |  |  |  |
| **B** | 0 | 1 | 0 |  |  |  |  |  |  |  |
| 0 | 1 | 1 |  |  |  |  |  |  |  |
| **D** | 1 | 0 | 0 |  |  |  |  |  |  |  |
| 1 | 0 | 1 |  |  |  |  |  |  |  |
| **C** | 1 | 1 | 0 |  |  |  |  |  |  |  |
| 1 | 1 | 1 |  |  |  |  |  |  |  |

**K-Map:**

0

1

00 01 11 10

*J*Q1 =

Q

1

Q X

2

0

1

00 01 11 10

*K*Q1 =

Q

1

Q X

2

0

1

00 01 11 10

*J*Q2 =

Q

1

Q X

2

0

1

00 01 11 10

*K*Q2 =

Q

1

Q X

2

0

1

00 01 11 10

*Z* =

Q

1

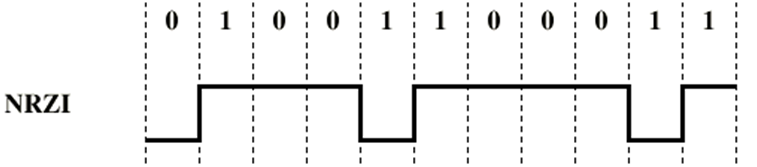
Q X

2

**Part4 (Workbench and Implementation):**

In many communication and networking systems, the signal transmitted on the communication line uses NRZI signal. You are requested to design and implement a circuit that converts any message sequence of 0s and 1s to a sequence in the NRZI format. The mapping for such a circuit is as follows:

1. If the message bit is a 1, then the NRZI format message contains an immediate change from 1 to 0 or from 0 to 1, depending on the current NRZI value.
2. If the message bit is a 0, the NRZI format message remains unchanged at 0 or 1, depending on the current NRZI value.

****

**No Change**

**Complement**

**1/1**

**State Diagram:**

**0/1**

**0/0**

**No Change**

**Complement**

**1/0**

**State Table:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Present State** | | **Input** | **Next State** | **Output** | **Inputs of JK flip-flop** | |
|  | ***Q*** | ***X*** | ***Q*** | ***Z*** | ***JQ*** | ***KQ*** |
| **A** | 0 | 0 |  |  |  |  |
| 0 | 1 |  |  |  |  |
| **B** | 1 | 0 |  |  |  |  |
| 1 | 1 |  |  |  |  |

**K-Map:**

*Z* =

0

1

0 1

Q

X

*K*Q =

0

1

0 1

Q

X

*J*Q =

0

1

0 1

Q

X