



ENCS2110

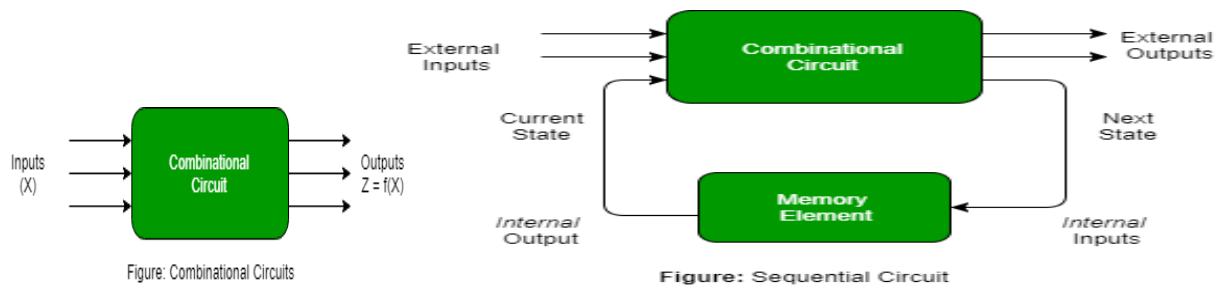
- Department of Electrical and Computer Engineering.
- Digital electronics and computer organization laboratory.
- Report: Experiment No. 5 - Sequential Logic Circuits
- Student name: Abdulrahman Shaheen.
- ID#: 1211753.
- Instructor Name: Dr. Qadri Mayyala
- Date: 20/5/2023.
- Section: 2.

Abstract:

The purpose of this experiment is twofold: firstly, to explore and analyze the disparities between sequential and combinational logic circuits, and secondly, to gain a comprehensive understanding of the functionality of flip-flops, specifically the D flip-flop and JK flip-flop.

Brief theoretical review:

Combinational circuits, which implement Boolean functions, and sequential circuits are the two types of digital circuits. Sequential circuits, also known as two-valued networks, differ from combinational circuits as they consider not only the present inputs but also the past inputs.



*Latches:

Latches are a type of memory unit commonly used in sequential circuits. They serve various purposes, such as sorting binary data and designing asynchronous circuits.

One example of a latch is the set-reset latch, which can be constructed using NOR or NAND gates. Here's an example of how it can be implemented:

1- S-R latch Using NOR gates:

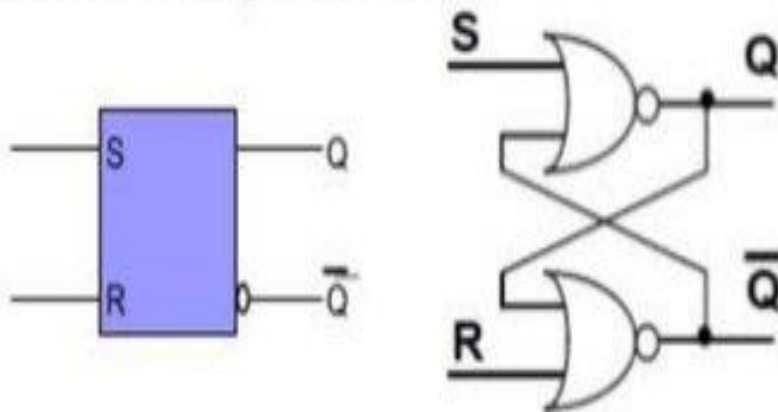
The latch described is an active-high latch. In this latch, when the input S (set) is set to 1, the latch will be set, and the output Q will be 1. On the other hand, when the input R (reset) is set to 1, the latch will be reset, and the output Q will be 0. When both S and R are set to 0, the latch will store the previously set value, acting as a memory. It's important to note that when both S and R are set to 1 simultaneously, this results in an undefined condition.

2- S-R latch Using NAND gates:

In the case of an active-low latch, certain conditions determine its behavior. When the input S (set) is set to 0, the latch will be activated, resulting in the output Q being set to 1. Conversely, when the input R (reset) is set to 0, the latch will be reset, causing the output Q to become 0. When both inputs S and R are set to 1, it signifies the memory state, where the latch retains its current value. However, when both S and R are set to 0 simultaneously, an undefined condition arises.

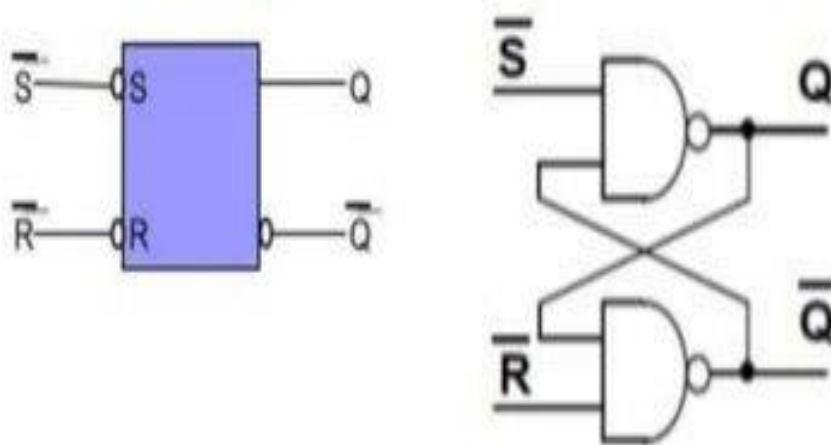
Symbol & Truth Table

i. Active HIGH i/put S-R Latch



S	R	Q	\bar{Q}	Comments
0	0	NC	NC	No Change (remain present state)
0	1	0	1	RESET
1	0	1	0	SET
1	1	0	0	INVALID

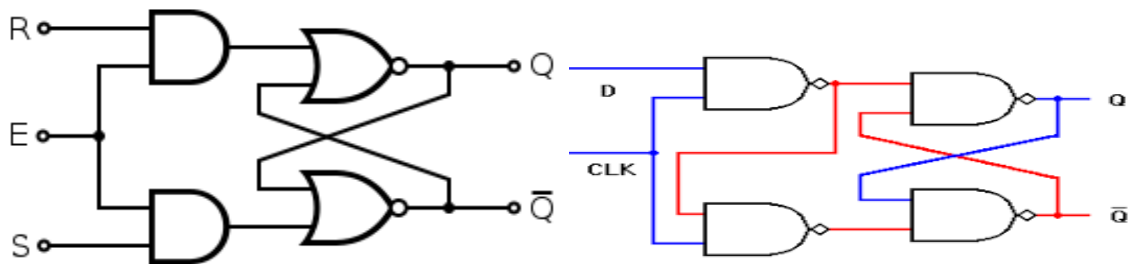
ii. Active LOW i/put S-R Latch



\bar{S}	\bar{R}	Q	\bar{Q}	Comments
0	0	1	1	INVALID
0	1	1	0	SET
1	0	0	1	RESET
1	1	NC	NC	No Change (remain present state)

3- S-R latche with control input

The addition of input control to the normal S-R latches addresses a potential issue known as the "racing" or "race around" problem. In a basic S-R latch, when both the S and R inputs change simultaneously, there is a brief moment where the outputs become unpredictable. This is due to the propagation delay within the gates, which can lead to conflicting feedback signals. By introducing control inputs to the S-R latch, the racing problem can be resolved. The control inputs enable a mechanism to disable or enable the latch's behavior, ensuring controlled and predictable operation. When the control input is activated, it overrides the S and R inputs and prevents the latch from responding to changes in those inputs.



CLK = E = C



Table 5.2: RS latch with control truth table.

INPUT			OUTPUT		State
C	S	R	Q_{n+1}	Q'	
0	X	X	Q_n	Q'	No Change/ Memory
1	0	0	Q_n	Q'	No Change/ Memory
1	0	1	0	1	RESET
1	1	0	1	0	SET
1	1	1	0	0	Indeterminate

4-The D Latch:

To overcome the issue of an undefined state in the RS latch, the D latch was developed. It features a single "D" input and an enable input. When the enable input is active, the D latch responds to changes in the D input by either setting or resetting the output accordingly. In contrast, when the enable input is inactive, the latch maintains its current state, preserving a stable output. This eliminates the problem of an undefined state and provides a dependable mechanism for data storage. The D latch finds extensive application in digital systems and sequential logic designs, ensuring reliable and controlled operation.

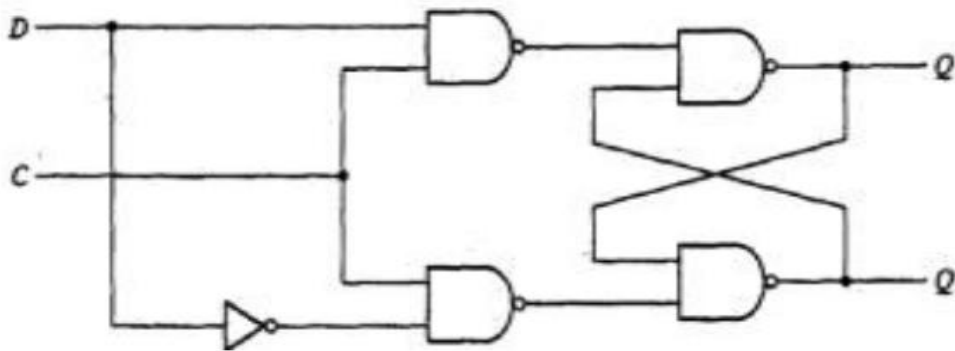


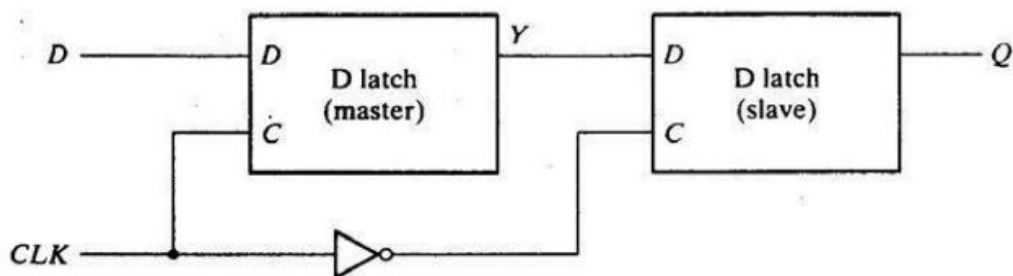
Table 5.3: D latch truth table.

INPUT		OUTPUT		State
C	D	Q_{n+1}	Q'	
0	X	Q_n	Q'	No Change/ Memory
1	0	0	1	RESET
1	0	1	0	SET

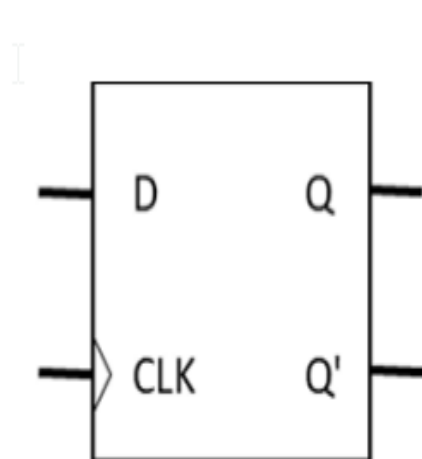
*Flip-Flops:

Flip-flops were introduced as an improvement over latches in digital circuits due to their enhanced functionality and reliability. With synchronous behavior, precise timing control, separate set and reset inputs, and stable data storage, flip-flops are better suited for sequential logic applications, providing a more robust and predictable solution compared to latches.

The implementation of a flip-flop can be achieved by using two separate latches. By utilizing two D latches, a D flip-flop can be constructed. This arrangement allows for the storage and manipulation of digital data.

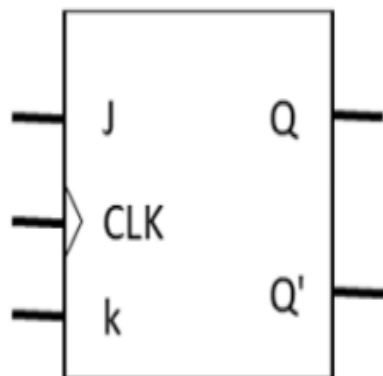


The most commonly used types of flip-flops in digital circuits are the D, T, and JK flip-flops. These flip-flops serve different purposes and have distinct characteristics that make them suitable for various applications in sequential logic design.



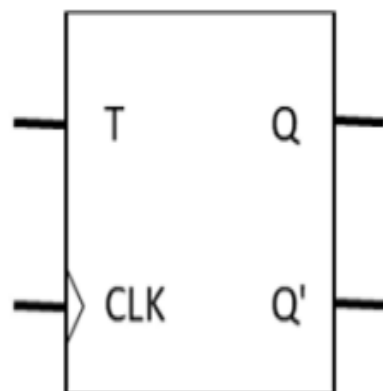
$$Q(n+1) = D$$

INPUT		OUTPUT		State
D		Q_{n+1}	Q_n'	
0		0	1	RESET
1		1	0	SET



$$Q(n+1) = Q(n) \oplus T$$

INPUT		OUTPUT	State
J	K	Q_{n+1}	
0	0	Q_n	No Change
0	1	0	RESET
1	0	1	SET
1	1	Q_n'	Complement



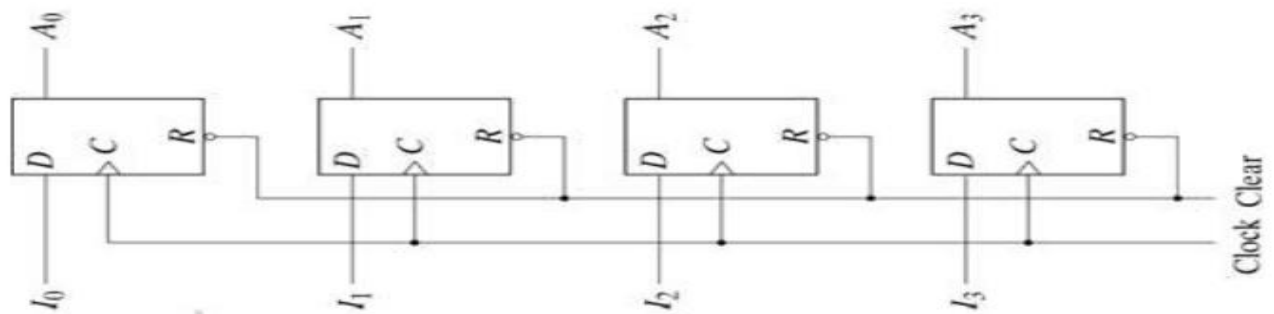
$$Q(n+1) = JQ'(n) + K'Q(n)$$

INPUT	OUTPUT	State
T	Q_{n+1}	
0	Q_n	No Change
1	Q_n'	Complement

Figure 5.6: D, JK, and T flip flops.

*Registers:

In digital systems, registers are commonly used to store binary data. A register is a collection of flip-flops, where an N-bit register consists of N individual flip-flops. These flip-flops are driven by a common clock signal, ensuring synchronous operation. Registers provide a means to store and manipulate digital information in various applications.



A shift register is a group of flip-flops that are interconnected in a chain, with the output of one flip-flop connected to the input of the next flip-flop. This sequential connection enables the shifting of data within the register. Shift registers are commonly used in various applications such as data storage, data manipulation, and serial-to-parallel or parallel-to-serial conversion.

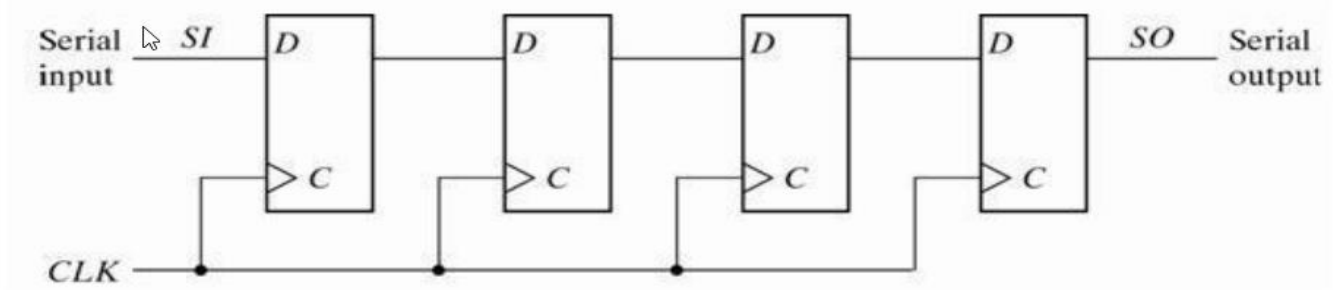


Figure 5.8: 4-bit shift- right register.

* Counters:

Counters are special-purpose registers that follow a prescribed sequence of states. They are classified as ripple counters and synchronous counters.

Ripple counters do not have a common clock; instead, the transition of one flip-flop triggers the next. Synchronous counters, on the other hand, receive a common clock signal for simultaneous state changes. Counters are essential in digital circuits for tasks such as frequency division, event counting, and sequence generation.

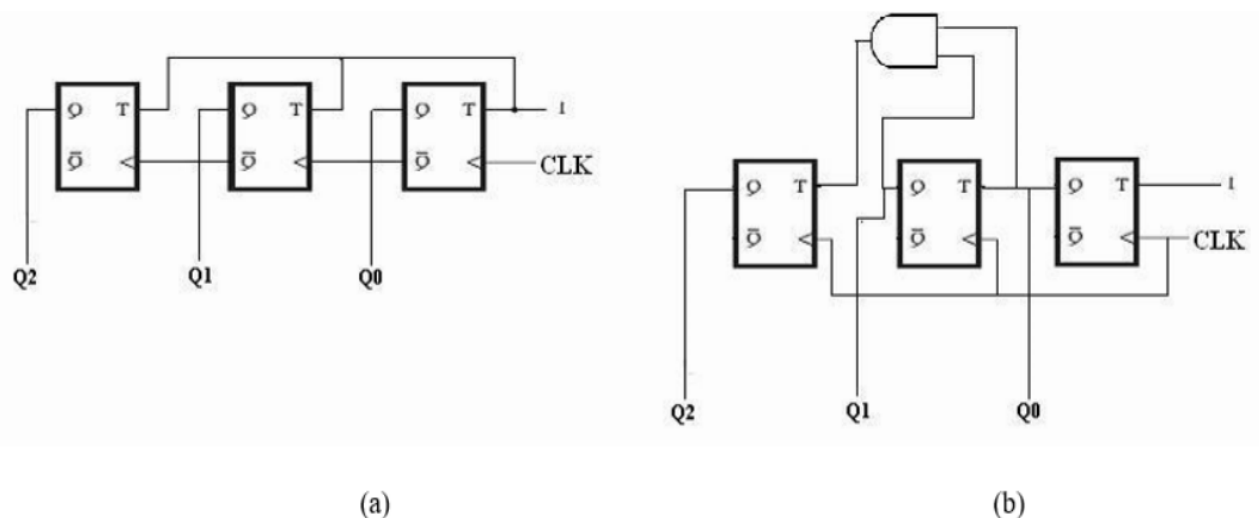
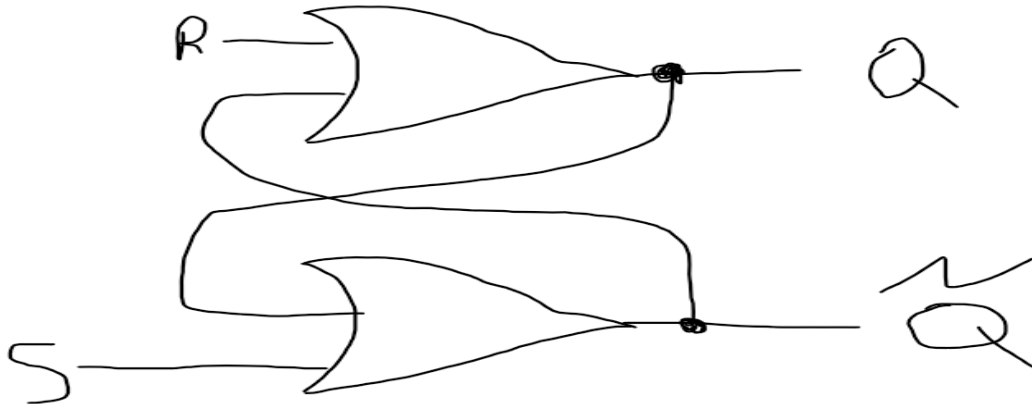


Figure 5.9: (a) 3-bit ripple counters, (b) 3-bit synchronous counter.

*Pre-Lab:

For each used IC (Integrated Circuits), search for its datasheet that explains exactly what a component does and how to use it:



The $R = S = 1$ combination is called a restricted combination or a forbidden state because, as both NOR gates then output zeros, it breaks the logical equation $Q = \bar{Q}$. The combination is also inappropriate in circuits where both inputs may go low simultaneously.

INPUT		OUTPUT		State
S	R	Q	\bar{Q}	
1	0	1	0	SET
0	0	1	0	No Change/ Memory
0	1	0	1	RESET
0	0	0	1	No Change/ Memory
1	1	0	0	Invalid

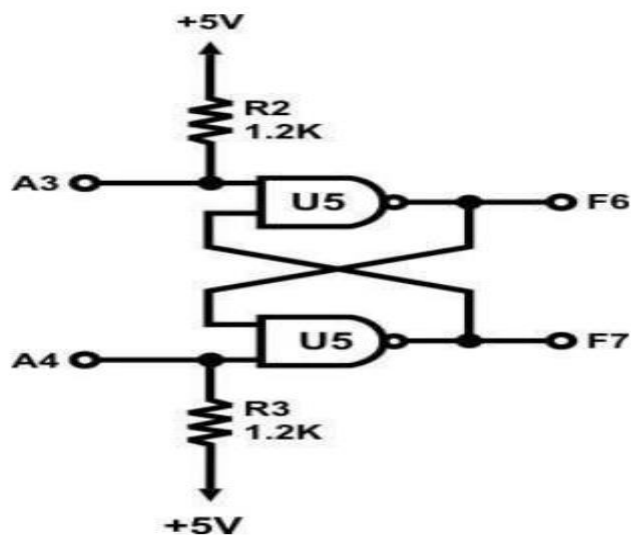
*Procedure, Data and results:

At the beginning of every section, we inserted the +5 V and the ground.

5.6.1 Latches and Flip flops:

a) Constructing RS latch with Basic Logic Gates:

To conduct the experiment, begin by constructing the circuit illustrated in Figure 5.10 using the IT-3008 module. Next, establish connections between the inputs A3 and A4 of the module and the corresponding switches SW1 and SW2. Finally, connect the outputs F6 and F7 of the module to the logic indicators L1 and L2, respectively.

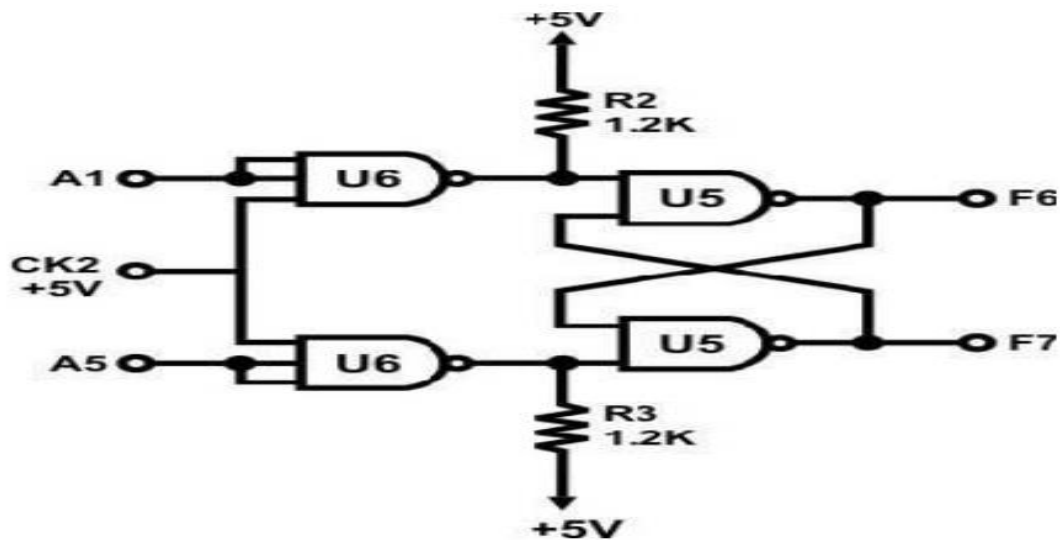


This

INPUT		OUTPUT	
A3	A4	F6	F7
0	0	1	1
0	1	0	1
1	0	1	0
1	1	1	0

b) Constructing RS latch with control input:

To conduct the experiment, utilize the IT-3008 module to establish the circuit as depicted in Figure 5.11. Connect the inputs A1 and A5 of the module to the respective switches SW1 and SW2.

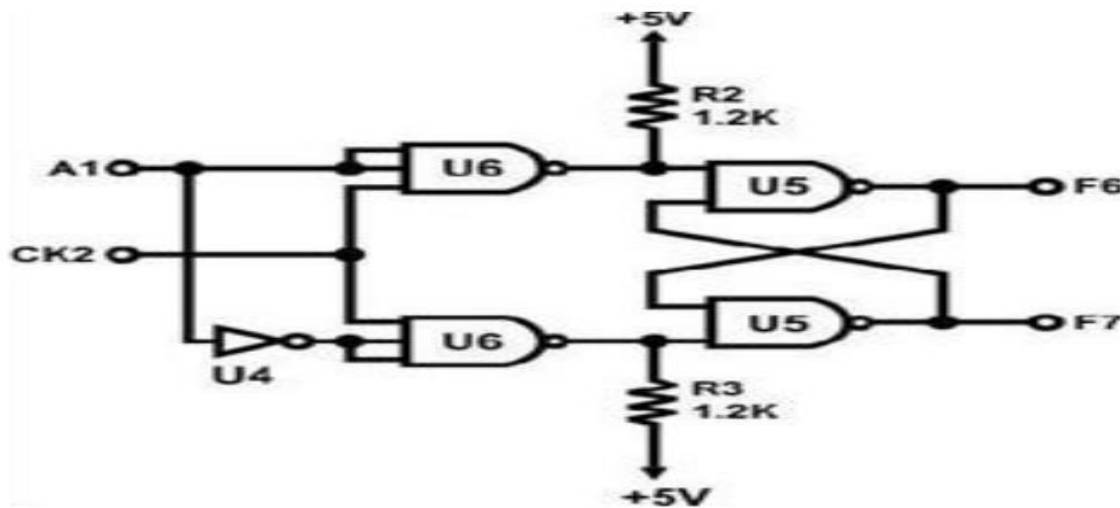


this

INPUT		OUTPUT	
A1	A5	F6	F7
0	0	1	0
0	1	0	1
1	0	1	0
1	1	1	1

c) Constructing D latch with RS latch”

For this experiment, employ the IT-3008 module to construct the circuit illustrated in Figure 5.12. Connect the input A1 of the module to the switch SW1. Connect the input CK2 of the module to the switch SWA A. Additionally, connect the output F6 of the module to the logic indicator L1.

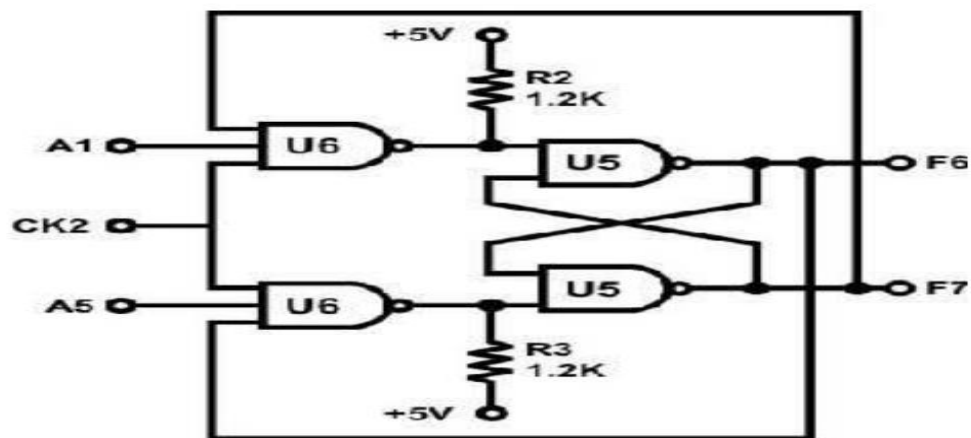


This

INPUT		OUTPUT
CK2	A1	F6
0	0	0
0	1	1
1	0	0
1	1	1

d) Constructing JK latch with RS latch:

To perform the experiment, begin by constructing the circuit depicted in Figure 5.13 using the IT-3008 module. Connect the input CK2 of the module to the output SWB B. Connect the input A1 to the output SW0 and the input A5 to the output SW1. Furthermore, connect the output F6 of the module to the logic indicator L1.

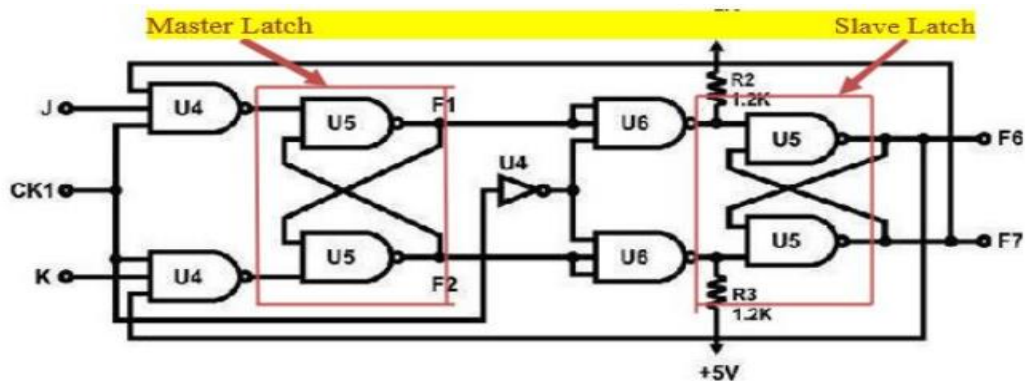


this

INPUT			OUTPUT	Status
CK2	A1	A5	F6	
	1	0		set
	0	0		No change
	1	1		change
	1	0		change
	0	0		No change
	0	1		change
	1	1		change

e) Constructing JK Flip-flop with master- slave RS latches:

The master-slave flip-flop addresses timing issues by utilizing two SR flip-flops. The "Master" flip-flop triggers on the leading edge of the clock pulse, while the "Slave" flip-flop triggers on the falling edge. This ensures that the master and slave sections are enabled during opposite half-cycles of the clock signal, effectively canceling timing problems. To perform this experiment, start by constructing the circuit illustrated in Figure 5.14 using the IT-3008 module. Connect CK1 to SWA A output, J to SW1, K to SW0, and F1, F2, F6, and F7 to L3, L2, L1, and L0, respectively.



This

INPUT			OUTPUT			
CK2	K	J	F1	F2	F6	F7
	0	0	0	1	0	1
	0	1	1	0	1	0
	1	0	0	1	0	1
	1	1	1	0	1	0
	1	1	0	1	0	1

5.6.2 Registers

a) Constructing Shift Register with D Flip-Flops:

To construct the circuit using the Block Shift Register 1 of the IT-3008 module, connect the clear input B to SW0 and the input A to SW1. The CK input should be connected to the SWA A output. Additionally, connect the outputs F1, F2, F3, and F4 to the logic indicators L1, L2, L3, and L4, respectively. Set SW0 to "0" to clear the B input, and then set it to "1". Now, follow the input sequence for the A input as described below:

a) When A = "1", send a CK signal from SWA. b) When A = "0", send a CK signal from SWA. c) When A = "0", send a CK signal from SWA. d) When A = "1", send a CK signal from SWA.

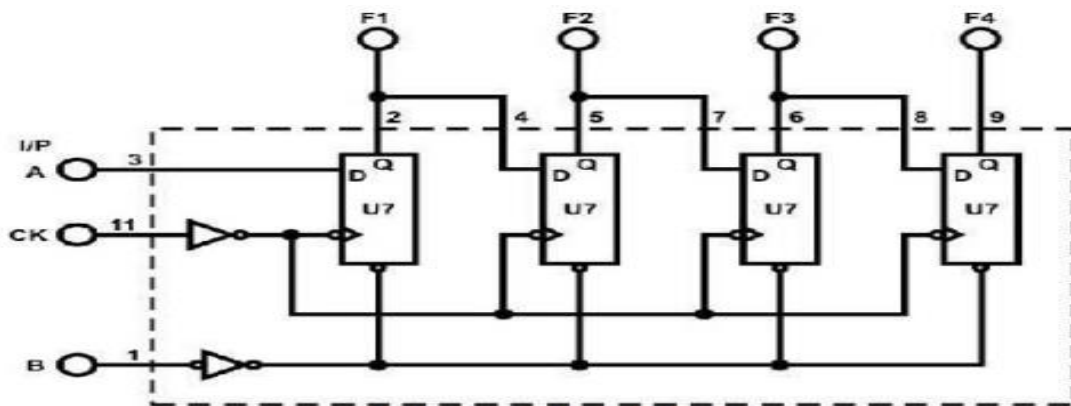


Figure 5.15: Shift Right Register.

INPUT		OUTPUT			
A	CK	F1	F2	F3	F4
1					
0					
0					
1					

b) 4-Bit Shift Register with serial and parallel load:

To utilize the Shift Register 2 module in the IT-3008, which is a 4-Bit Shift Register with serial and parallel synchronous operating modes, follow these steps:



4- Make the following connections:












- Connect inputs A, B, C, and D to SW0, SW1, SW2, and SW3, respectively.
- Connect outputs F1, F2, F3, and F4 to logic indicators L0, L1, L2, and L3, respectively.
- Connect B1 (input) to DIP2.0.
- Connect A1 (MODE) to DIP2.1 according to Table 5.10.

5- Connect CK (C1) to a clock generator with a TTL level output set at 1Hz. Change the data at B1 using DIP2.0. Follow the input sequences for A1 as specified in Table 5.11. Observe and record the corresponding outputs.






By following these steps, you can effectively use the Shift Register 2 module and analyze its behavior based on the provided input sequences.

Table 5.10: A1 modes.

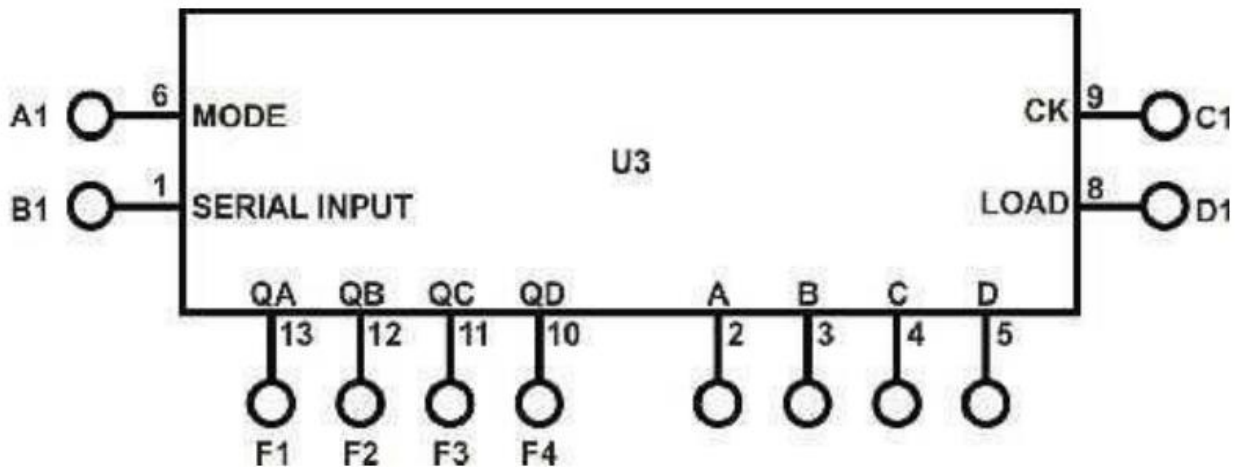
MODE Control (A1)	Input	
	CK	
	C1	D1
L		X
H	X	

INPUT		OUTPUT			
A1	C1	L3	L2	L1	L0
0					1
0				1	1
0			1	1	1
1			1	1	1

To proceed with the next step, connect the LOAD (D1) input of the Shift Register 2 module to the clock generator's TTL level output set at 1Hz. Set A1 to "1" as instructed. Follow the input sequences for inputs A, B, C, and D, as provided in Table 5.12. While following the sequences, observe and record the corresponding outputs generated by the module. This will allow you to analyze the behavior and response of the shift register under different input configurations.

Input					Output			
D1	D	C	B	A	L3	L2	L1	L0
	0	0	1	0	0	0	1	0
	1	0	1	0	1	0	1	0
	1	1	1	0	1	1	1	0
	0	1	1	1	0	1	1	1
	0	1	1	0	0	1	1	0

This



5.6.3 Counters:

a) 2-bit Synchronous Counter:

To implement the 2-bit synchronous counter shown in Figure 5.17, you will need to use the IT-3007 module. Begin by connecting the +5V output of the fixed power supply to the +5V input of the IT-3007 module. Similarly, connect the GND output of the power supply to the GND input of the module. This ensures proper power supply connections.

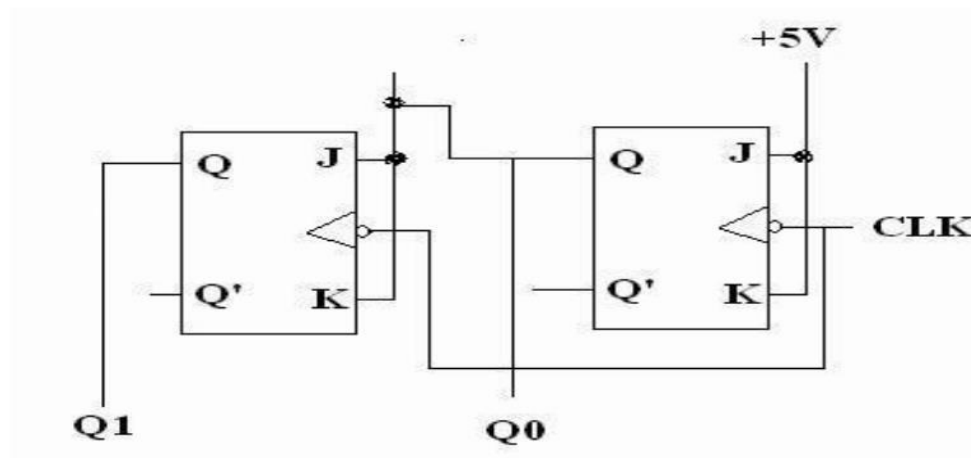
Next, connect the CLK input of the counter to the pulser switch SWA. This will provide the clock pulses necessary for the counter operation.

To observe the counter outputs, connect the counter outputs Q1 and Q0 to the indication lamps L1 and L2, respectively. This will visually indicate the binary outputs of the counter.

Now, apply clock pulses to the CLK input by activating the pulser switch SWA. Observe and record the outputs (in binary format) of the counter in Table 5.13(a) as the clock pulses are applied.

To further analyze the counter outputs, you can connect the counter outputs Q1 and Q0 to a seven-segment display. This will display the decimal representation of the counter outputs. Observe and record the decimal outputs in Table 5.13(b) as the counter advances.

By following these steps and recording the outputs, you will be able to understand the behavior and functionality of the 2-bit synchronous counter.

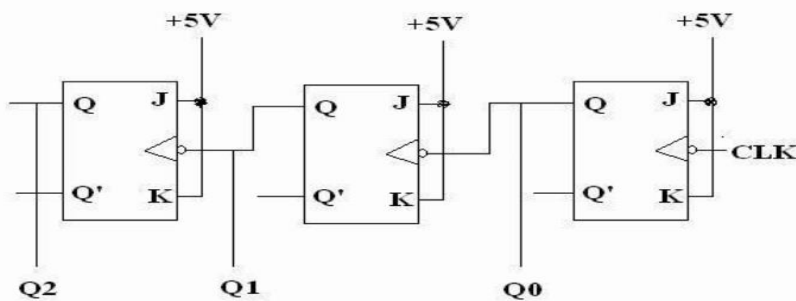


Input	OUTPUT	
CLK	Q1	Q0
	0	1
	1	0
	1	1
	0	0
	0	1
	1	0
	1	1
	0	0

Input	OUTPUT
CLK	D
	1
	2
	3
	0
	1
	2
	3
	0

b) 3-bit (divide-by-eight) Ripple Counter:

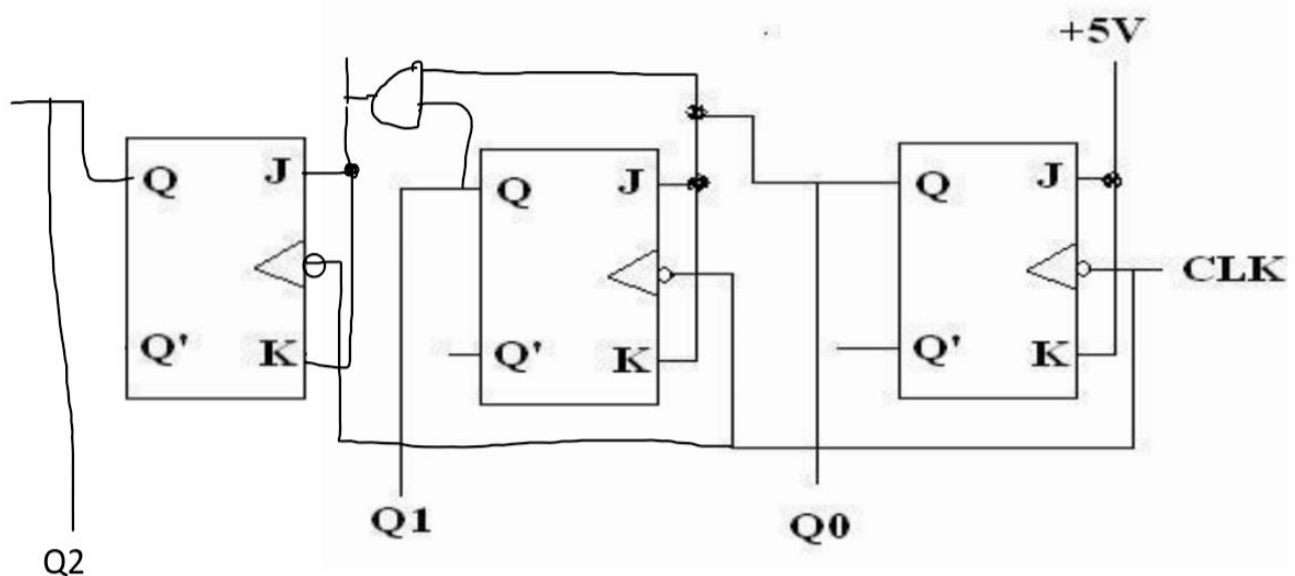
To implement the 3-bit divide-by-8 ripple counter using the IT-3007 module, follow these steps. First, connect the CLK input to a pulser switch to provide clock pulses. Next, connect the counter outputs Q2, Q1, and Q0 to indication lamps to display the binary outputs. Apply clock pulses to the CLK input and observe the outputs as they cycle from 0 to 7, recording the binary values in Table 5.14(a). For a more convenient representation, connect the counter outputs to a seven-segment display to show the decimal values. Observe and record the decimal outputs in Table 5.14(b) as the counter advances. These steps will allow you to understand how the 3-bit divide-by-8 ripple counter operates and counts from 0 to 7.



Input	OUTPUT		
CLK	Q2	Q1	Q0
	0	0	0
	0	0	1
	0	1	0
	0	1	1
	1	0	0
	1	0	1
	1	1	0
	1	1	1
	0	0	1
	0	0	0

Input	OUTPUT
CLK	D
	0
	1
	2
	3
	4
	5
	6
	7
	0
	1

Task 2 : Modify the circuit in Figure 5.17 to be 3-bit Synchronous Counter. Attach the design with this experiment report.



c) BCD Counter:

To locate the BCD counter (IC 7490) on the IT-3008 module and understand its functionality, follow these steps. Firstly, connect the +5V of the IT-3008 module to the +5V output of a fixed power supply, and connect the GND of the module to the GND output of the power supply. Next, make the following connections: connect C3 and C4 to SW0 and SW1 respectively, connect D1 and D2 to SW2 and SW3 respectively, connect F1 to L1, F2 to L2, F3 to L3, and F4 to L4. Finally, connect A2 to the SWA A output. To measure and record the outputs F1, F2, F3, and F4, set F1 to B2, set C3, C4, D1, and D2 to ground, and set A2 to SWA A pulse. Lastly, set SW2 and SW3 to 0 to observe the effect on the outputs.

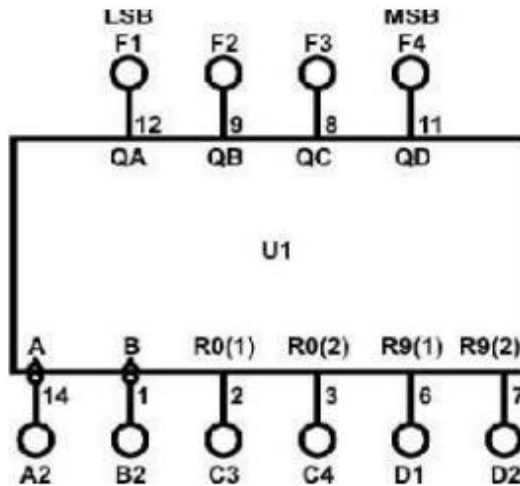
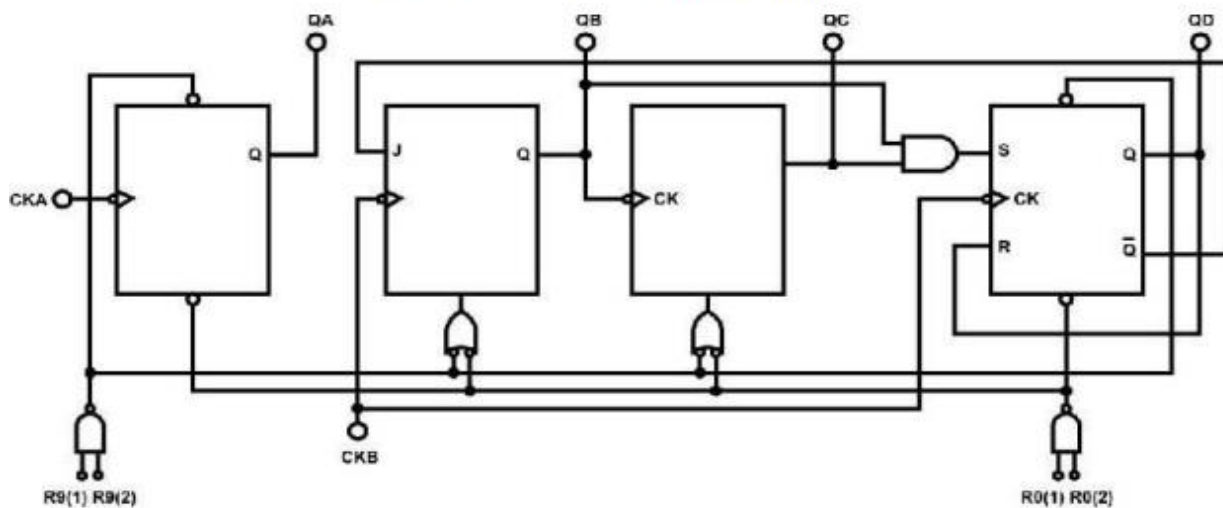


Figure 5.19: IC 7490 BCD Counter.



* Discussion and Evaluation:

1. Although latches are useful for storing binary information, they are rarely used in sequential circuit design, why?

1- Timing Issues: Latches are level-sensitive circuits, which means they are sensitive to the input level while the “enable” signal is active. Sometimes this causes timing glitches. Flip-flops, on the other hand, are edge-triggered, so they respond one time when the transaction happens.

2- Design Constraints: latches are built with more complex control signals compared with flipflops. Since they have an enable signals that have to be managed carefully.

3-power consumption: Latches are generally more power-hungry compared to flip-flops. This is because latches continuously evaluate their inputs as long as the enable signal is active.

2- What is the disadvantage of the RS flip flop?

Invalid State: The RS flip-flop can have an invalid state when both inputs (S and R) are both set to logic 1. This condition is known as the "forbidden" state. In this state, the outputs of the flip-flop can rapidly oscillate, making it impossible to determine the stable output.

3. What is the difference between “synchronous” and “ripple” counters?

Synchronous counters are designed to operate in synchronization with an external clock signal. They use a common clock signal to trigger the state transitions of all the flip-flops within the counter simultaneously. However, in Ripple counters, also known as asynchronous counters, don't rely on a common clock signal for state transitions. Instead, each flip-flop within the counter triggers the next flip-flop in a cascading manner. The output of each flip-flop serves as the clock input for the next flip-flop in the sequence.

*Conclusions:

In conclusion, our series of experiments involving various digital circuits and components provided us with valuable insights into the world of digital electronics. We explored different types of flip-flops, including the D flip-flop, JK flip-flop, and master-slave flip-flop, each serving unique purposes and addressing specific timing challenges.

Through the construction and analysis of registers, shift registers, and counters, we observed how these components store and manipulate binary data. The utilization of clock signals in synchronous circuits ensured precise timing and coordinated operation, minimizing timing issues and enhancing reliability.

We also gained hands-on experience in connecting modules, switches, indicators, and other hardware components to construct functional circuits. By following input sequences and recording output observations, we were able to analyze and understand the behavior of these circuits.

Furthermore, our experiments emphasized the significance of synchronization in digital systems. Synchronous circuits, driven by a common clock signal, offer advantages such as improved timing accuracy, reduced power consumption, and enhanced noise immunity. They play a vital role in complex digital systems that require precise timing and coordinated operations.

Overall, our exploration of digital circuits and synchronous components provided a solid foundation for understanding the principles and practical aspects of digital electronics. We acquired knowledge about the construction, operation, and significance of various digital components, enabling us to design and analyze robust and efficient digital systems.

*References:

http://www.ee.surrey.ac.uk/Projects/CAL/seq-switching/General_seq_circ.htm

<http://hyperphysics.phy-astr.gsu.edu/hbase/electronic/flipflop.html>

<http://electronics.stackexchange.com/questions/61530/how-to-understand-the-sr-latch>

<http://www.ee.nthu.edu.tw/jcliao/Logic98/chap06/showch06.html>

Feedback:

The experiments provided hands-on learning in digital circuits, allowing practical implementation and observation of circuit behavior. Visual aids and additional context could enhance the understanding and application of the concepts.