CONTENTS

Introduction:

In today's technologically driven world, the integration of digital systems is ubiquitous, with digital clocks being a prime example of their application. Digital clocks offer precision, reliability, and versatility, making them essential timekeeping devices in various settings, from household appliances to industrial processes. Behind the sleek and straightforward display of time lies a complex system of sequential circuits, orchestrating the precise sequencing of signals to accurately display time.

This introduction serves as a gateway to exploring the intricate workings of digital clocks, focusing specifically on the underlying sequential circuits that drive their functionality. Through an exploration of sequential circuits, we aim to unravel the intricate mechanisms that govern the operation of digital clocks, shedding light on how these devices synchronize and display time with remarkable accuracy.

This discussion will delve into the fundamental concepts of sequential circuits, their classifications, and their role in the design and operation of digital clocks. By understanding the principles governing sequential circuits, we can grasp the inner workings of digital clocks and appreciate the engineering marvels that enable us to keep time efficiently in the digital age.

Key Components used in the Design:

- NAND gate
- Flip-Flops
- Mod-N Counters

NAND Gate:

The NAND gate or “Not AND” gate is the combination of two basic logic gates, the AND gate and the NOT gate connected in series. The NAND gate and NOR gate can be called the universal gates since the combination of these gates can be used to accomplish any of the basic operations. Hence, NAND gate and NOR gate combination can produce an inverter, an OR gate or an AND gate.

The output of a NAND gate is high when either of the inputs is high or if both the inputs are low. In other words, the output is always high and goes low only when both the inputs are high. The logic NAND function is given by the Boolean expression.

$$Y = \overline{A.B} = (AB)'$$

The Boolean expression given for a NAND gate is that of logical addition and it is opposite to AND gate. The Boolean expression is given by a single dot (.) with an overline over the expression to show the NOT or the logical negation of the NAND gate. The symbol of the NAND gate is represented as a combination of AND gate and NOT gate.

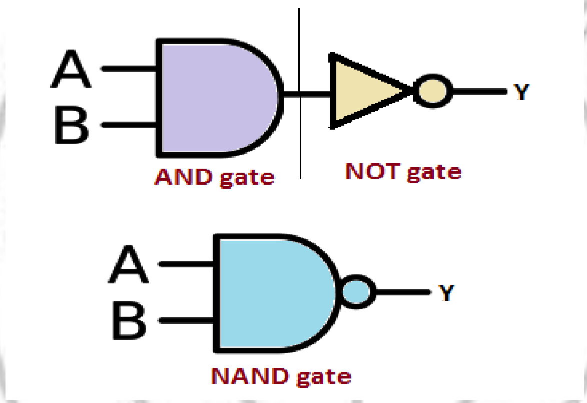

Figure 1: NAND gate in digital circuits.

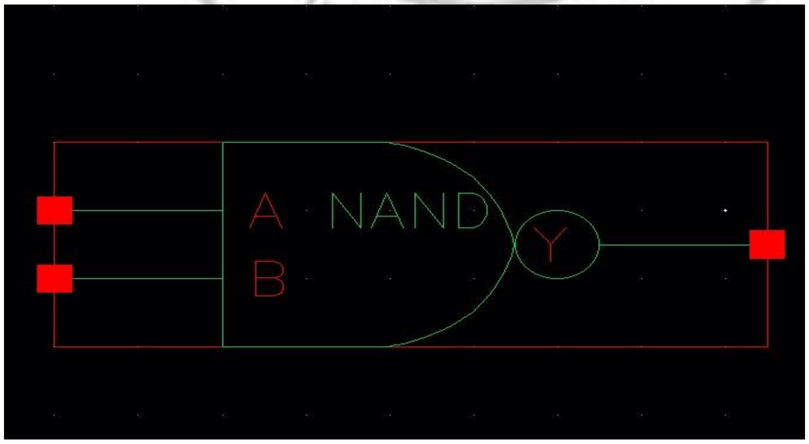

Figure 2: Implementation of NAND gate in Cadence Software.

The truth table of a NAND gate is given below

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

Table 1: Truth Table of NAND gate

NAND Gates in MOSFET Representation:

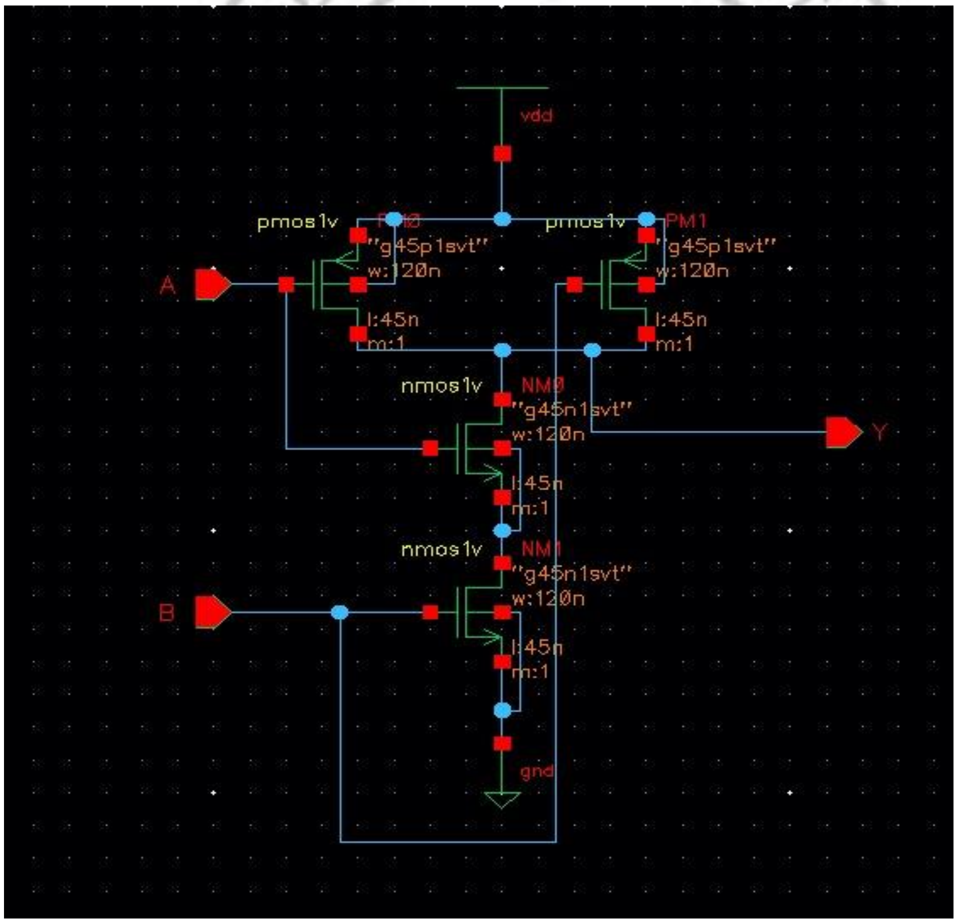

Figure 3: NAND Implementation Using MOSFETs in Cadence Software

This gate is composed of two NFETs in series to pull the output low, and two PFETs in parallel to pull the output high, which satisfies rule 1.

The operation of the two input NAND gate is as follows. When the A and B inputs are both high, the two NFETs turn on and pull the output low. At this point, the two PFETs are both off, so neither is pulling the output high and rule 2 is satisfied.

If either A or B is low, then at least one of the NFETs will turn off, preventing the output from being pulled low. Simultaneously, at least one of the PFETs will turn on and pull the output high, satisfying rule 3.

The two input NAND gate can be extended to three inputs by placing three NFETs in series and three PFETs in parallel. The operation is the same as for the two input NAND gate, satisfying all three basic rules. Continuing the process, the NAND gate can be further extended to more three inputs as well.

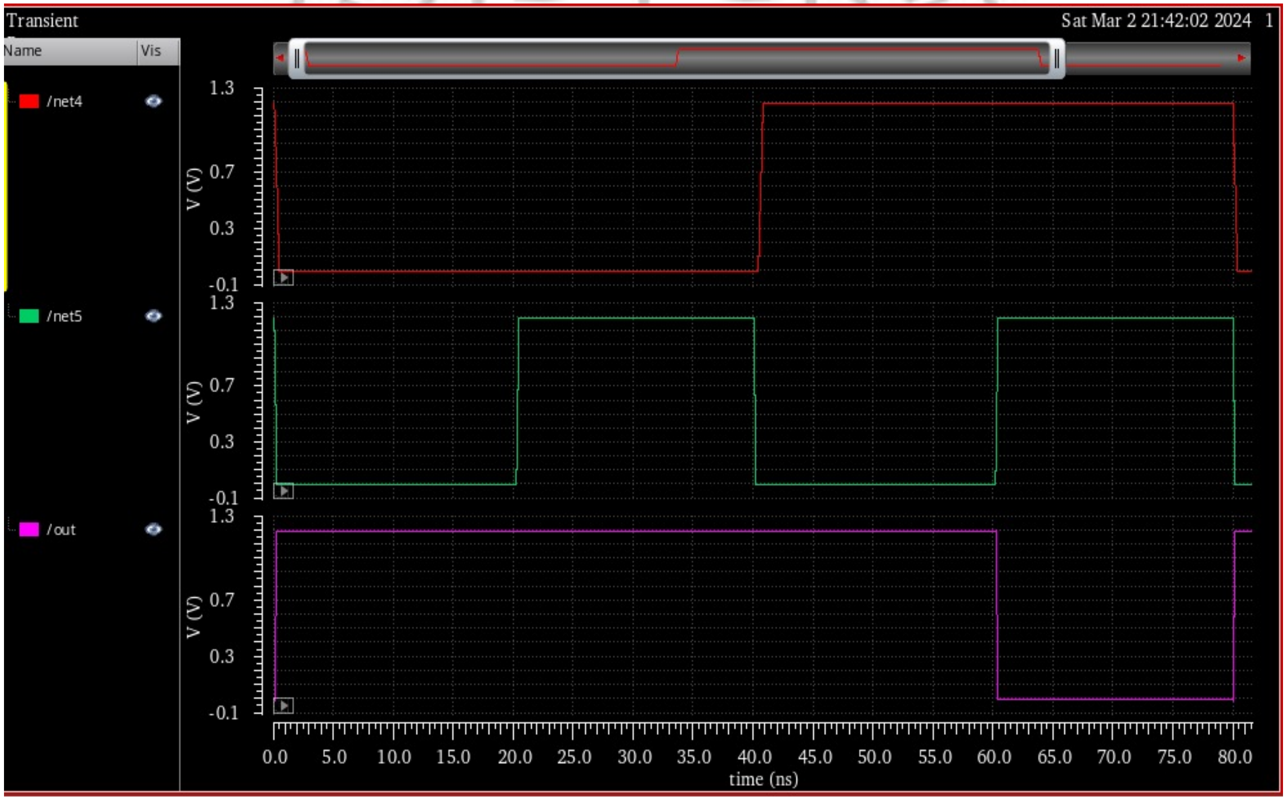

Figure 4: Output of NAND gate in Cadence Software

Flip-Flops

Flip-flop digital circuits are fundamental building blocks for storing binary or state information. They are bistable multivibrators, which means they have two stable states and can be used to represent and store a single binary bit of information (0 or 1). Flip-flop digital circuits are critical components in sequential logic circuits, such as memory elements, registers, and various digital counters.

Flip-flops play a crucial role in the design of digital systems, including microprocessors, memory devices, and various digital controllers, enabling the processing and storage of binary information. Different types of flip-flops are chosen based on a given application's specific requirements and timing constraints. There are several types of flip-flops, each with its own characteristics.

Types of Flip-Flop Digital Circuits	
Set-Reset (SR) Flip-Flop	It has two inputs: Set(S) and Reset (R) • When S=1 and R=0, it sets the flip-flop to state Q=1 • When S=0 and R=1, it resets the flip-flop to state Q=0 • When both S and R are 0, the flip-flop retains its previous state • When both S and R are 1, it can lead to an unpredictable state
Toggle (T) Flip-Flop	It has a toggle input (T) and a clock input (CLK) • When T=1 and a clock pulse is applied, the flip-flop toggles its state • When T=0 and a clock pulse is applied, the flip-flop maintains its current state
JK Flip-Flop	It has three inputs: J (set), K (reset) and clock input (CLK) • When J=1 and K=0, it sets the flip-flop to state Q=1 • When J=0 and K=1, it resets the flip-flop to state Q=0 • When both J and K are 0, the flip-flop retains its previous state • When both J and K are 1, the flip-flop toggles(inverts) its state
Data or Delay (D) Flip-Flop	It has a data input (T) and a clock input (CLK) • The flip-flop stores the value of D at the rising or falling edge of the clock signal, depending on its type (positive or negative edge-triggered)

Table 2: Types of Flip-Flops

Negative Edge-Triggered JK Flip-flop:

Master Slave JK flip flop – The Master-Slave Flip-Flop is basically a combination of two JK flip-flops connected together in a series configuration. Out of these, one acts as the "master" and the other as a "slave". The output from the master flip flop is connected to the two inputs of the slave flip flop whose output is fed back to inputs of the master flip flop. In addition to these two flip-flops, the circuit also includes an inverter. The inverter is connected to clock pulse in such a way that the inverted clock pulse is given to the slave flip-flop. In other words, if CP=0 for a master flip-flop, then CP=1 for a slave flip-flop and if CP=1 for master flip flop then it becomes 0 for slave flip flop.

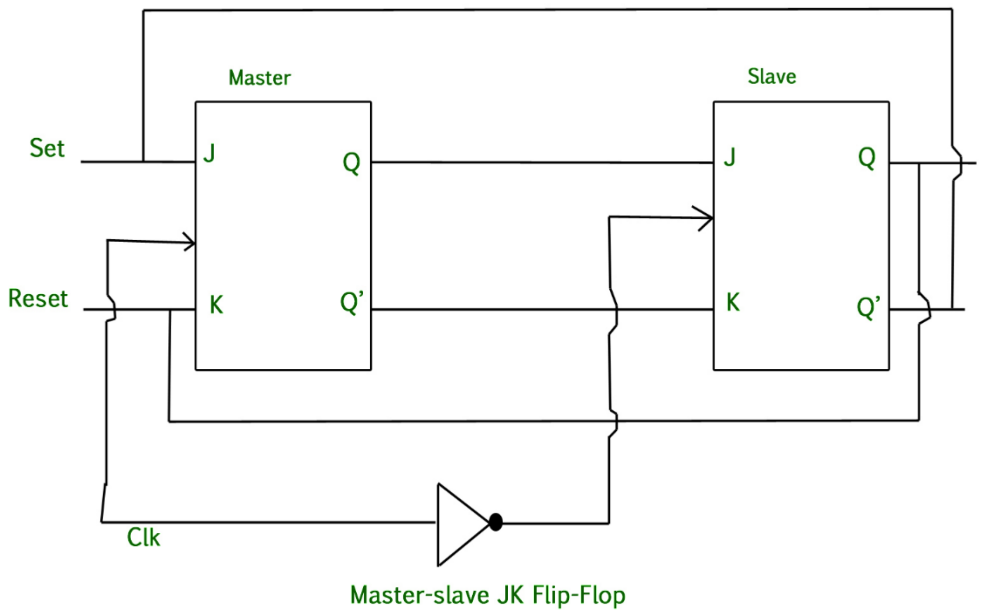

Figure 5: Negative Edge triggered Master-Slave JK Flip-Flop

Working of a master slave flip flop

1. When the clock pulse goes to 1, the slave is isolated; J and K inputs may affect the state of the system. The slave flip-flop is isolated until the CP goes to 0. When the CP goes back to 0, information is passed from the master flip-flop to the slave and output is obtained.

2. Firstly, the master flip flop is positive level triggered and the slave flip flop is negative level triggered, so the master responds before the slave.

3. If J=0 and K=1, the high Q' output of the master goes to the K input of the slave and the clock forces the slave to reset, thus the slave copies the master.

4. If J=1 and K=0, the high Q output of the master goes to the J input of the slave and the Negative transition of the clock sets the slave, copying the master.

5. If J=1 and K=1, it toggles on the positive transition of the clock and thus the slave toggles on the negative transition of the clock.

6. If J=0 and K=0, the flip flop is disabled and Q remains unchanged.

Timing Diagram of a Master Slave flip flop

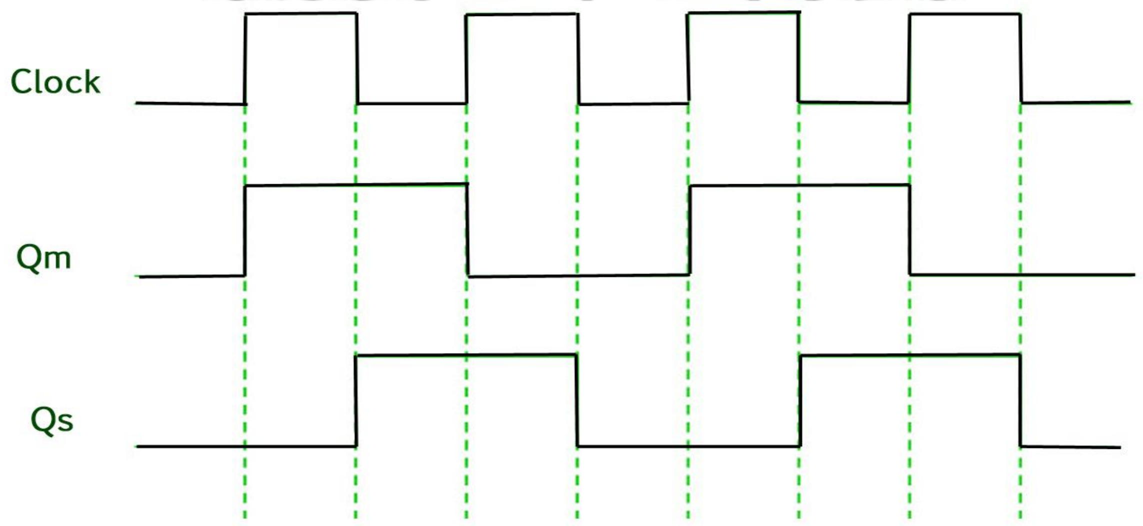

Figure 6: Timing Diagram of Master-Slave JK Flip-Flop

- When the Clock pulse is high the output of master is high and remains high till the clock is low because the state is stored.

- Now the output of master becomes low when the clock pulse becomes high again and remains low until the clock becomes high again.

- Thus, toggling takes place for a clock cycle.

- When the clock pulse is high, the master is operational but not the slave thus the output of the slave remains low till the clock remains high.
- When the clock is low, the slave becomes operational and remains high until the clock again becomes low.
- Toggling takes place during the whole process since the output is changing once in a cycle.

Preset and Clear Facility in JK Flip-flop

Commercially available JK flip-flops (which are available in ICs form) contain two additional inputs i.e. preset (PS) and clear (CLR), which are known as asynchronous inputs. These asynchronous inputs relinquish or reject synchronous inputs (J & K) as and when required. Normally, PS and CLR are capable of active low status. It means that they are normally high however, whenever the circuit is desired to be preset or cleared, they are momentarily turned low. By means of pressing low preset push button, Q value becomes 1 and pressing low clear button, Q value turns out 0. It has to be remembered that whenever a computer is logged on, operator press a master reset button, which transmits a clear (reset) signal on all flip-flops.

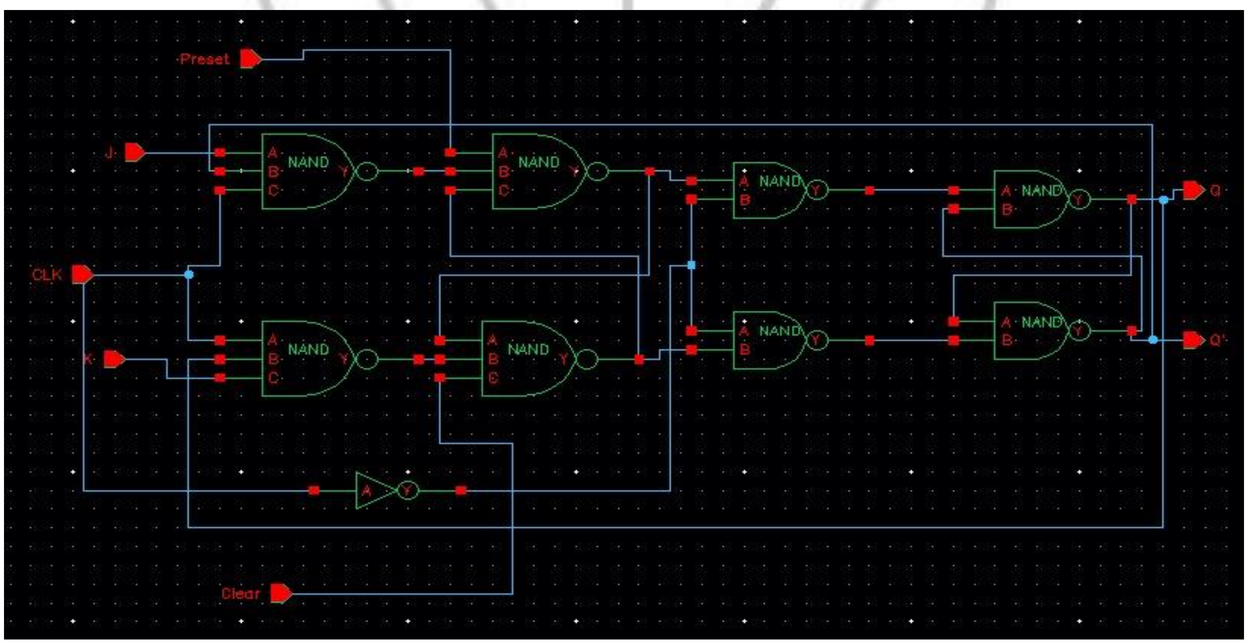

Figure 7: Negative Edge Triggered JK Flip-Flop with Preset and Clear in terms of NAND gate in Cadence Software

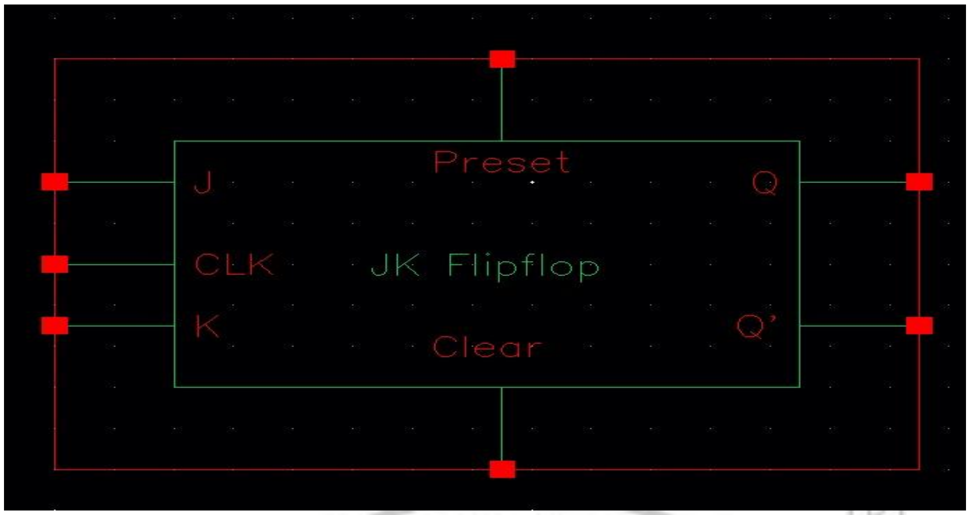

Figure 8: Symbol of JK Flip-Flop in Cadence Software

Counters

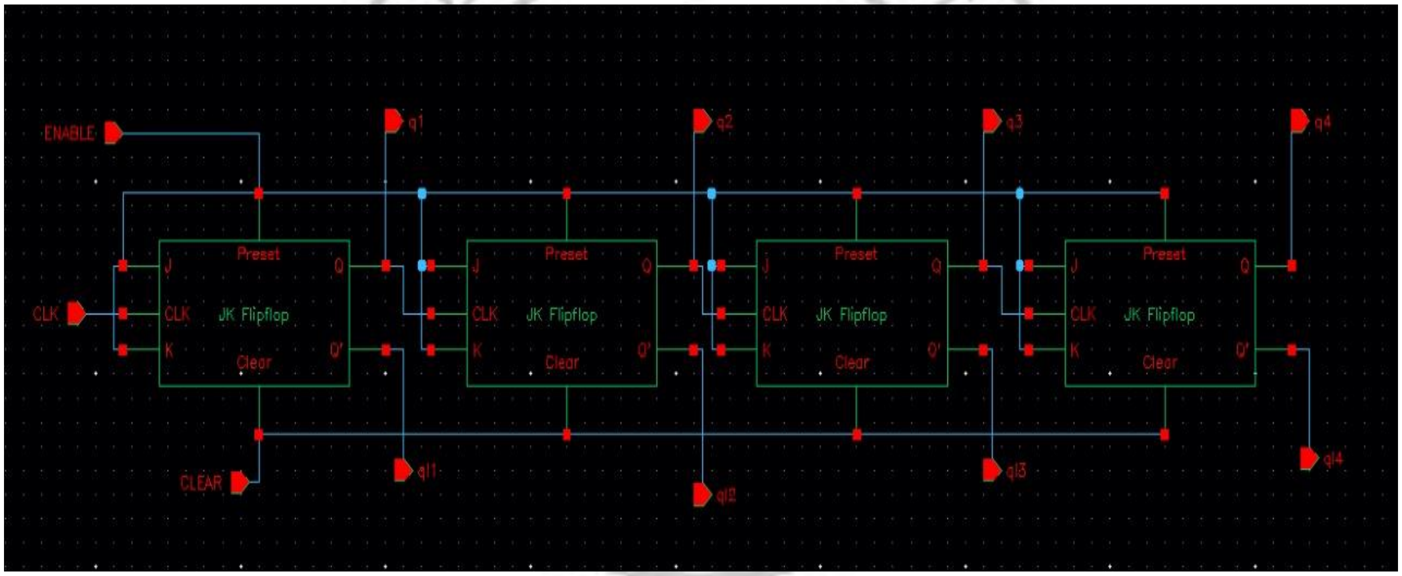

Figure 9: 4-Bit Asynchronous Counters in Cadence Software

The first flip-flop (the one with the Q0 output), has a negative-edge triggered clock input, so it toggles with each falling edge of the clock signal.

I've shown the signal in this manner for the purpose of demonstrating how the clock signal need not be symmetrical to obtain reliable, "clean" output bits in our four-bit binary sequence.

Using one J-K flip-flop for each output bit, however, relieves us of the necessity of having a symmetrical clock signal, allowing the use of practically any variety of high/low waveform to increment the count sequence.

As indicated by all the other arrows in the pulse diagram, each succeeding output bit is toggled by the action of the preceding bit transitioning from “high” (1) to “low” (0).

This is the pattern necessary to generate an “up” count sequence.

A less obvious solution for generating an “up” sequence using positive-edge triggered flip-flops is to “clock” each flip-flop using the Q’ output of the preceding flip-flop rather than the Q output.

Since the Q’ output will always be the exact opposite state of the Q output on a J-K flip-flop (no invalid states with this type of flip-flop), a high-to-low transition on the Q output will be accompanied by a low-to-high transition on the Q’ output.

In other words, each time the Q output of a flip-flop transitions from 1 to 0, the Q’ output of the same flip-flop will transition from 0 to 1, providing the negative-going clock pulse we would need to toggle a negative-edge triggered flip-flop at the right moment:

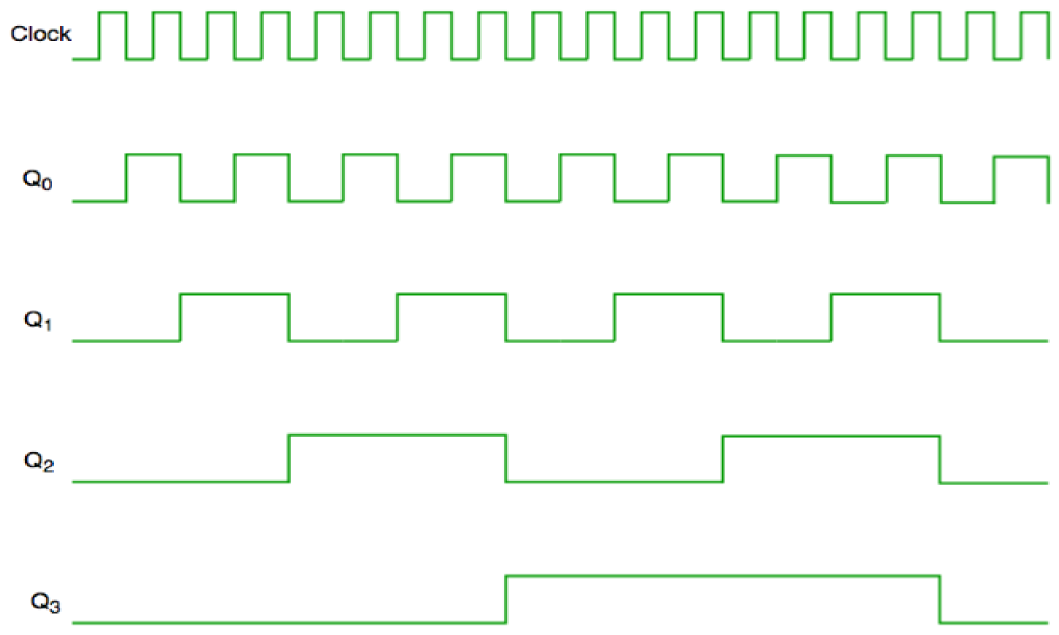

Figure 10: Timing Diagram of 4-bit Asynchronous Counters.

MOD-6 Counter

A modulo 6 (MOD-6) counter circuit, known as divide-by-6 counter. The circuit design is such that the counter counts from 0 to 5, and then on the 6th count it automatically resets to begin the count again. Since we are using the sixth count itself to cause a reset, it is unstable.

The trick is to start with a counter and then look for the binary sequence 110, which is 6 in decimal. Since this binary sequence is unique, we look for the sequence of 1's and feed them into a NAND gate. The output from the AND gate is then used to control the RESET function on all three flip-flops. Remember that binary is read from right to left (LSB to MSB), whilst the counter output is left to right. Students are likely to make mistakes here, and end up using the incorrect outputs to feed the NAND gate.

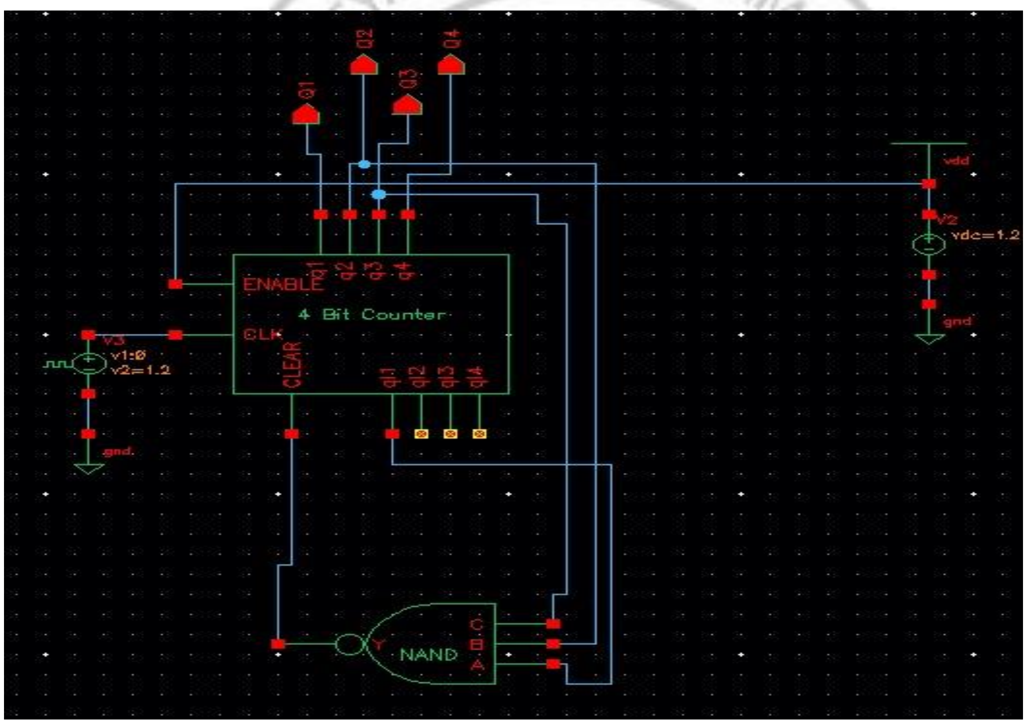

Figure 11: MOD-6 Counter in Cadence Software.

DESIGN OF DIGITAL CLOCK

The Digital Clock mainly consist of 3 major key aspects, namely Seconds, Minutes and Hours. A MOD-6 Counter and A MOD-10 Counter is used to compute Seconds, while same set of components are used to represent the Minutes in the Digital Clock which is shown in Figure

12. The output of the Minutes and Seconds are shown in figure 13. The clock signal triggers the Seconds counter. When the Seconds reach 59, the carry signal increments the Minutes counter. Similarly, the carry from the Minutes counter updates the Hours.

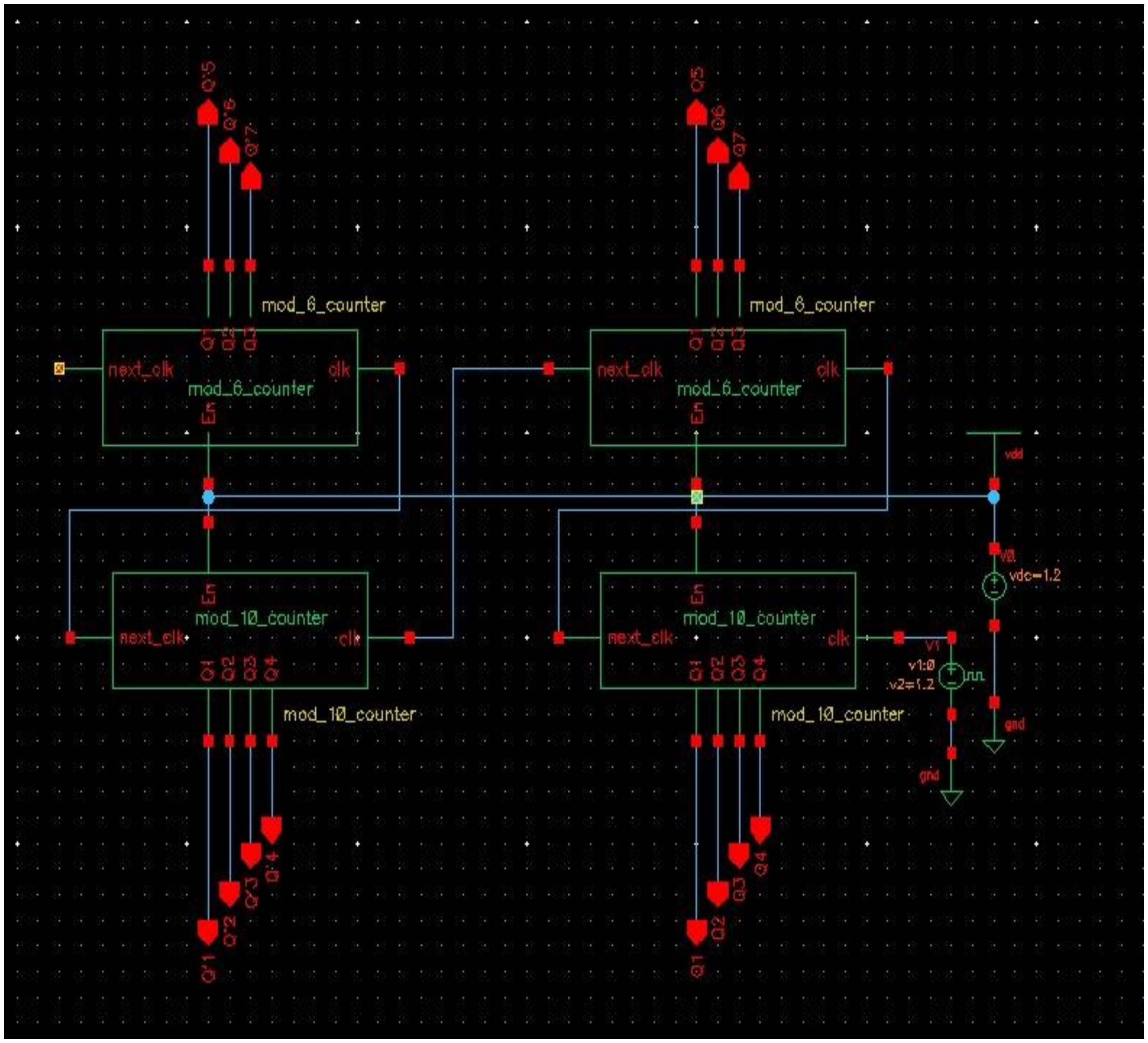

Figure 12: Seconds and Minutes representation in Digital Clock Using MOD-6 & MOD-10 Counters in Cadence Software.

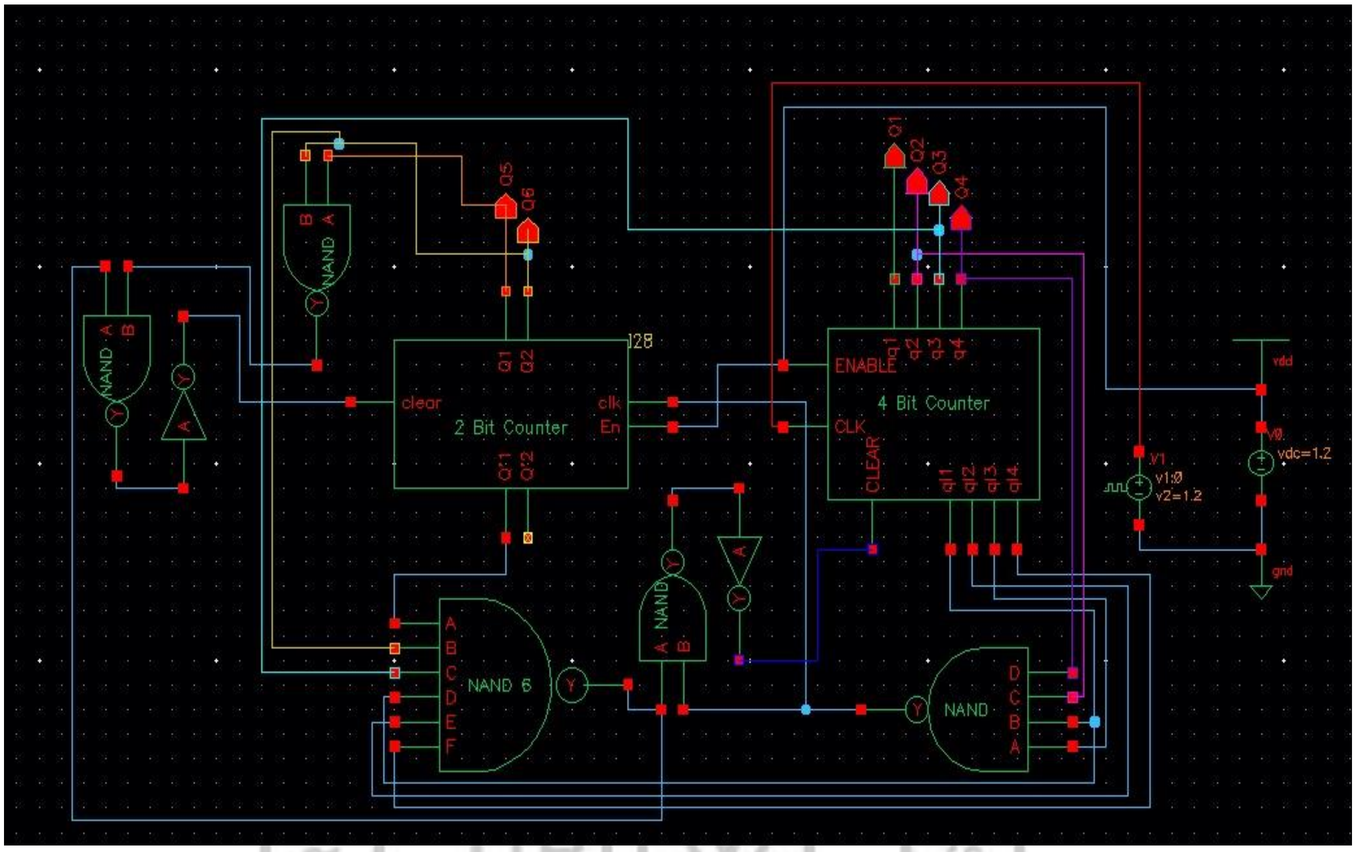

Figure 13: Digital Clock Schematics in Cadence Software.

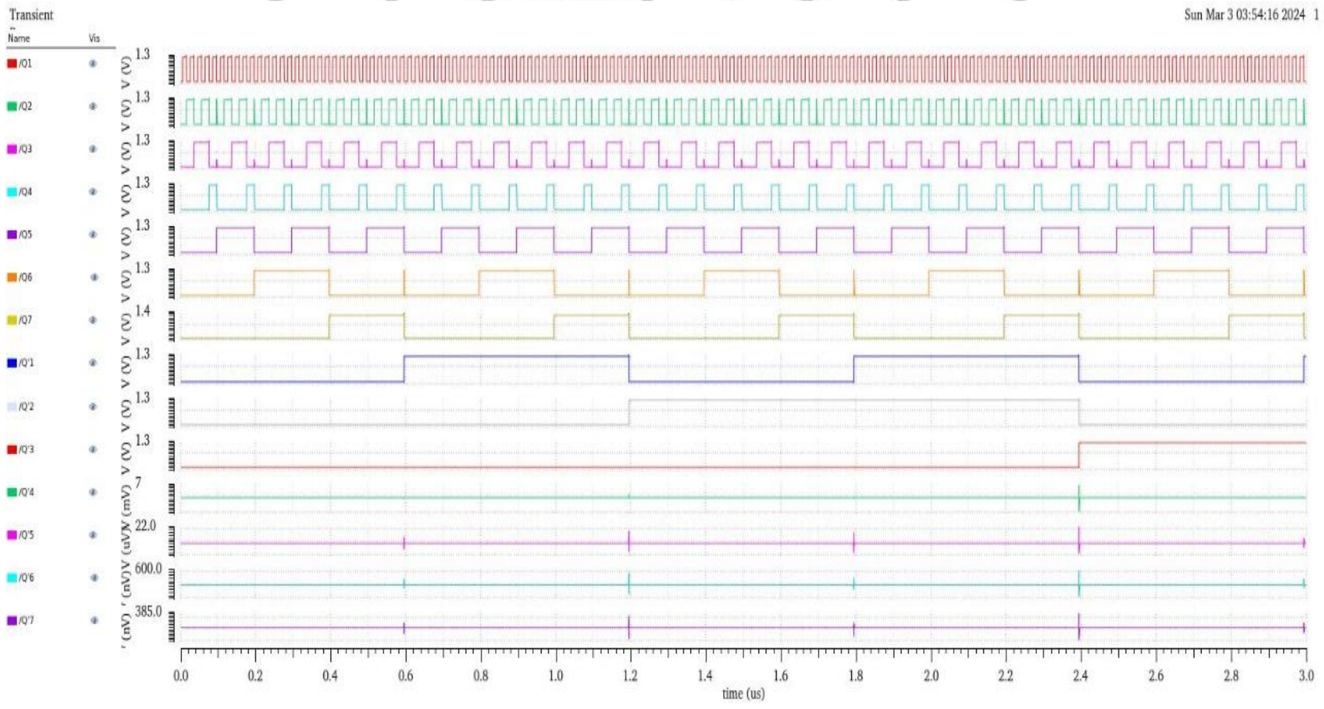

Figure 14: Output obtained for the Digital Clock in Cadence Software

LEARNING OUTCOMES WITH RESPECT TO COURSE OUTCOMES

The implementation and simulation of the elevator controller circuit using Cadence provided valuable learning outcomes in various aspects of electronics and control systems. Here are some key learning outcomes from this experiment:

1. Practical Application of Components:

   Participants gained hands-on experience in incorporating essential electronic components, such as Flip-Flops, Latches, counters, and logical gates, into a real-world application. This practical exposure deepened their understanding of component functionalities and applications in control systems.

2. System Integration and Interfacing:

   The experiment enhanced participants' skills in integrating multiple components into a cohesive system. Understanding how the encoder communicates with the comparator, how the comparator influences the counter, and how these components interface with the motor control system provided insights into system-level design.

3. Signal Processing and Feedback Mechanisms:

   Participants learned about signal processing in the context of the elevator controller. The comparator's role in comparing signals from the encoder with predefined thresholds for floor detection illustrated the importance of feedback mechanisms in control systems, ensuring accurate floor tracking.

4. Simulation and Analysis Techniques:

   The use of Cadence for simulation equipped participants with valuable skills in setting up simulations, defining parameters, running analyses, and interpreting results. This experience is transferable to other electronic design and simulation tools, enhancing their proficiency in virtual prototyping.

5. Troubleshooting and Debugging:

The simulation process provided an opportunity for participants to identify and troubleshoot issues within the circuit design. This skill is crucial in real-world scenarios where debugging and refining a design are common steps in the engineering process.

6. Application of Control Logic:

   The use of logical gates to control the elevator's movement based on user input and floor detection showcased the application of control logic in real-time scenarios. This knowledge is fundamental for designing intelligent and responsive control systems.