A black background with white text

Description automatically generated

**SCSR1013 – DIGITAL LOGIC**

**SEMESTER I, SESSION 2024/2025**

**Project Digital Logic**

**Network Packet Transmission Monitoring System**

**Lecturer: Dr. Zuriahati Binti Mohd Yunos**

**Section: 05**

|  |  |
| --- | --- |
| **NAME** | **MATRIC NO** |
| LOOI YU XIANG | A24CS0107 |
| EDWIN OO MING HAO | A24CS0245 |
| YEOH KENG WEI | A24CS0316 |
| CHING QI YAN | A24CS0235 |

**DEDICATION AND ACKNOWLEDGE**

We dedicate this project to our incredible team members, whose hard work, collaboration, and commitment have been instrumental in bringing this endeavour to life. Their dedication, creativity, and teamwork have made this journey both productive and rewarding, and their contributions will always be deeply appreciated.

This project is dedicated to our esteemed lecturer, Dr. Zuriahati Binti Mohd Yunos, whose invaluable guidance, expertise, and unwavering support have been crucial to the successful completion of this work. We would like to sincerely thank Dr. Zuriahati Binti Mohd Yunos for her guidance and support throughout this project. Her advice and encouragement have been invaluable in helping us stay on track and complete our work successfully. We deeply appreciate the time and effort she has dedicated to mentoring us, and we are truly grateful for her contributions to this project.

We also dedicate this work to our families and friends, who have provided unwavering support—both technically and emotionally—and encouraged us through every challenge along the way.

**Introduction**

Modern computing system consists of complex system and technologies. These technologies are built upon some fundamental simple logics known as digital logic. By using digital logic gates, we can develop complex logical circuit for various purposes like data storing, data manipulation or simply data representation (GeeksforGeeks, 2023). This Digital Logic project is to implement knowledge gained from this course by simulating the real case outside classroom which is simulate a Network Packet Transmission Monitoring System. In this scenario, a group of computers located in Lab 1 is connected to another group of computers in Lab 2 through a single cable. The objective is to transmit a specific number of data packets, defined by the user, from a source computer in Lab 1 to a destination computer in Lab 2. This setup aims to facilitate efficient data transfer between the two labs while ensuring smooth communication across the connected systems.

The circuit for transmitting packets between Lab 1 and Lab 2 is designed and simulated using Deeds (Digital Electronics Education and Design Suite). The computers in Lab 1 and Lab 2 are connected via a single cable, and a multiplexer (MUX) is used at the source in Lab 1 to select and send data packets. At the destination in Lab 2, a demultiplexer (DEMUX) routes the received packets to the correct computer. This Count-up counter Function built using flip-flops like JK, D, or T, keeps track of the number of packets being sent and matches it to the number specified by the user. A comparator is also used to compare the transmitted packet count with the user-defined value, ensuring the system stops transmitting once the target is reached. Finally, the circuit includes a clock enabler, implemented using an AND or NAND gate, to synchronize the overall operation. These components work together to ensure smooth, efficient, and precise data transfer between the two labs.

As a student, we could improve our soft skills such as communication, teamwork, time management and critical thinking through this project. By using DEEDS in this project, students gain hands-on experience that allows them to apply theoretical concepts in a practical, real-world setting.

Lastly, this project gives students a great chance to put their Digital Logic knowledge into practice using DEEDS. It helps develop important skills like teamwork, communication and time management skils, while encouraging students to come up with innovative solutions for real-world challenges. By working hands-on, students not only strengthen their understanding of theory but also gain valuable experience that will help them tackle future challenges.

**Problem background**

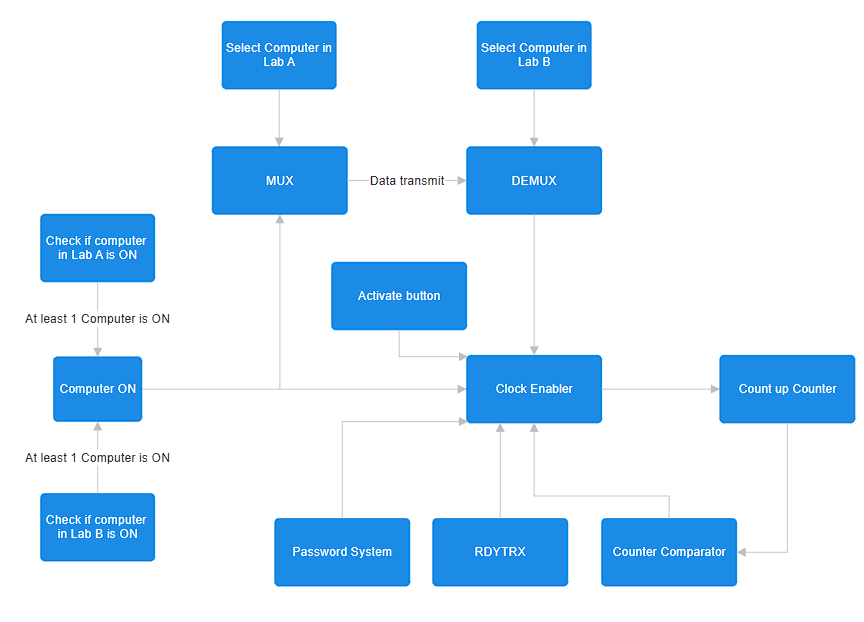
In today's fast-paced technological environment, efficient and seamless interaction between devices is vital. The demand for faster project completion has driven the need for advanced data transmission and monitoring systems.

This project addresses these challenges by establishing a connection between a group of computers in Lab 1 and a different group of computers in Lab 2 using a single cable. The system will facilitate the transmission of user-defined numbers of data packets from a source computer in Lab 1 to a destination computer in Lab 2.

A system for monitoring and managing the packet transmission is requiring to be developed. This system aims to optimize data flow, detect potential issues, and provide real-time insights to enhance overall performance.

A Network Packet Transmission Monitoring System is being designed and simulated by using DEEDS, a digital circuit simulator to fulfil this requirement. Some key component involved in this architecture such as clock, multiplexers (MUX), demultiplexers (DEMUX), JK flip-flop.

**Suggested Solution**



To activate the clock enabler that can make the count up counter add 1, there are a few conditions to achieve.

1. Computer is ON.
   1. Both Lab A and Lab B should have at least one computer is switched ON.
   2. If there are no computer is ON, but the user chooses one of computer from both sides, the system will not work.
   3. This system can help technician to check whether a computer is connected to the system properly or not.
2. Select computer from Lab A and Lab B.
   1. User should choose one of the computers from Lab A to send data.
   2. User should choose one of the computers from Lab B to receive data.
   3. User can only choose one computer for each sides.
   4. User cannot choose the computer which is not ON yet.
3. PIN system
   1. The PIN system guarantee the protection of data.
   2. It ensures that only a user with authentic which is the correct password can operate the system.
   3. A predefined password is set by the moderator and will be compare with the password input by the user to grant or deny the access.
4. RDYTRX
   1. RDYTRX (Ready for transmission) is a button for user to select where the data is ready to transfer or not.
   2. When the Button is turn to 0, no data can be transfer. When the button is turn to 1, data can transfer.
   3. This button can ease the security operation and regular maintenance check-up.
5. Activate Button
   1. It is a button to activate the clock enabler.
   2. It is a push button. It can give one pulse to the clock when pushed.
   3. There will be count up by one when the button is pushed.
   4. It acts as a trigger of the system.
6. Counter Comparator
   1. This comparator is to avoid data overflow.
   2. The user can choose the maximum number of data packet to be transfer.
   3. When the data packet transfer is larger than the amount that user selected, the system will be terminated automatically.
7. Clock Enabler
   1. Clock Enabler is managing the timing of synchronization of the system
   2. It can activate the clock that will affect the timing of the data transmission.
   3. It mainly controls the count up system.
   4. It can only be activated by only the computer is ON, data transferred, password is correct, counter comparator functioning, RDYTRX is on, activate button is pushed.
8. MUX and DEMUX
   1. A system to transfer data from a selected computer in Lab A to a selected computer in LAB B.

1. Count up counter
   1. A counter that can record and summarize the data transfer from Lab A and Lab B
   2. It can ensure that no data overflow by cooperating with counter comparator.

**The Requirement**

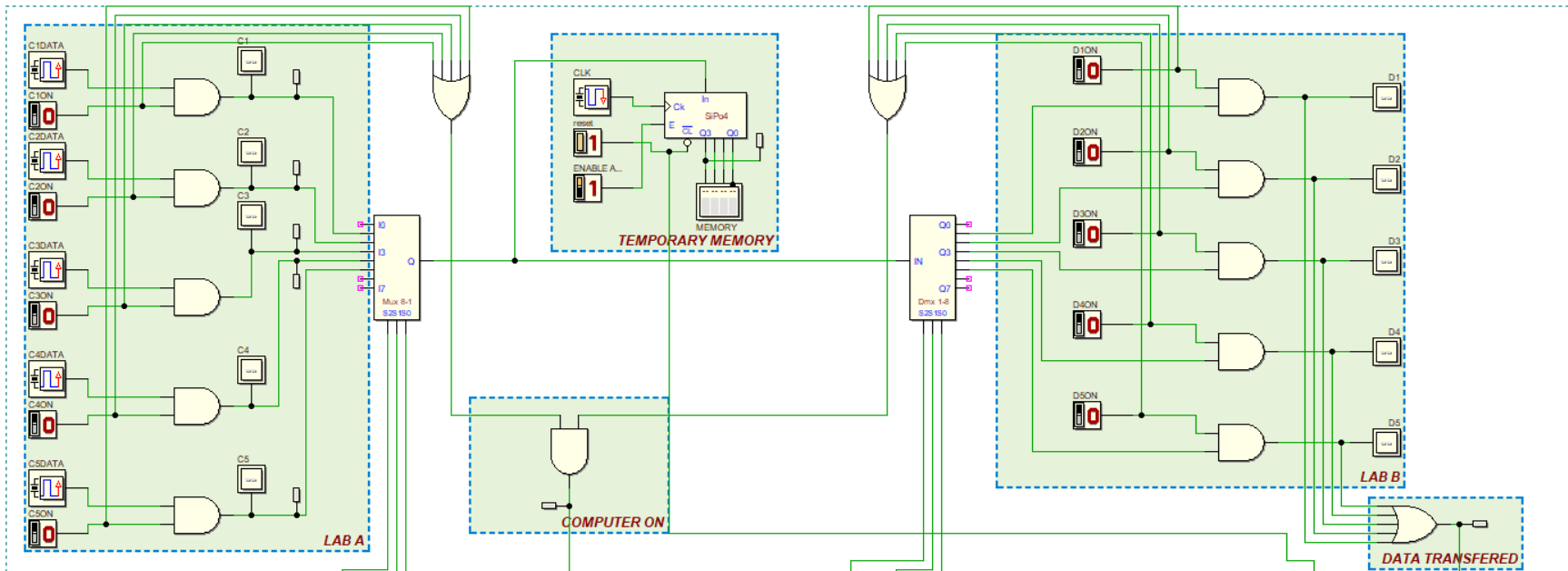
The requirements for the Network Packet Transmission Monitoring System are:

1. **Computer is ON**: At least one computer from both Lab A and Lab B need to be **ON** to operate the system.
2. **Select computer from Lab A and Lab B**: One computer from Lab A to send and one computer from Lab B to receive data.
3. **Password System**: Users must enter the correct 4-digit password to operate the system
4. **MUX and DEMUX**: MUX takes multiple input signals and provides a single output and DEMUX takes input from one source and converts the data to transmit towards various sources which will make the data transfer between Lab A and Lab B more efficient.
5. **Count up Counter**: The counter can tracks and summarize the total bits of data transfer from Lab A and Lab B.
6. **Counter Comparator**: The comparator will compare the data packet transfer with the maximum number of data packet set by the user to determine if data reception can continue.
7. **RDYTRX**: Button that let the user able to enable or disable the data transmission for security or maintenance purpose.
8. **Activate Button**: Button to activate the Clock Enabler. It will give one pulse to the clock when pushed.
9. **Clock Enabler**: It can only be activated by only the computer is ON, data transferred, password is correct, counter comparator functioning, RDYTRX is on, activate button is pushed.
10. **Serial-In Parallel-Out (SIPO) shift register:** The SIPO shift register can be used to temporarily save data transferred between Lab A and Lab B. This stored data can be reused as needed, ensuring efficient data handling.

These 9 requirements ensure the whole Network Packet Transmission Monitoring System operates quickly and effectively in an organized way.

**System Implementation**

**Lab A and Lab B**

Figure 5.1: Lab A and Lab B

**Lab A**: In Lab A, 5 computers are connected to a **multiplexer (MUX)**. Each computer has a **power-on button** that must be turned on before sending a signal. The **clock input**, which alternates between 0 and 1, acts as the data source. The data source and power-on button connected to an **AND gate**, resulting in 5 outputs—one from each AND gate. The MUX then consolidates these outputs into a single signal.

**Lab B**: The output from the MUX in Lab A is connected to the input of a **demultiplexer (DEMUX)** in Lab B. The DEMUX distributes the signal to 5 computers in Lab B. Each computer in Lab B has a **power-on button** that must be turned on to receive a signal. Each output from the DEMUX is connected to an **AND gate** along with the corresponding power-on button to ensure the computer is powered on and can receive the signal.

**Computer On**: Each lab includes a **5-input OR gate** connected to the power-on buttons of the computers. The outputs of these OR gates are then connected to an **AND gate** to ensure that at least one computer in each lab is powered on. This output is significant as it influences the input value of the **clock enabler**.

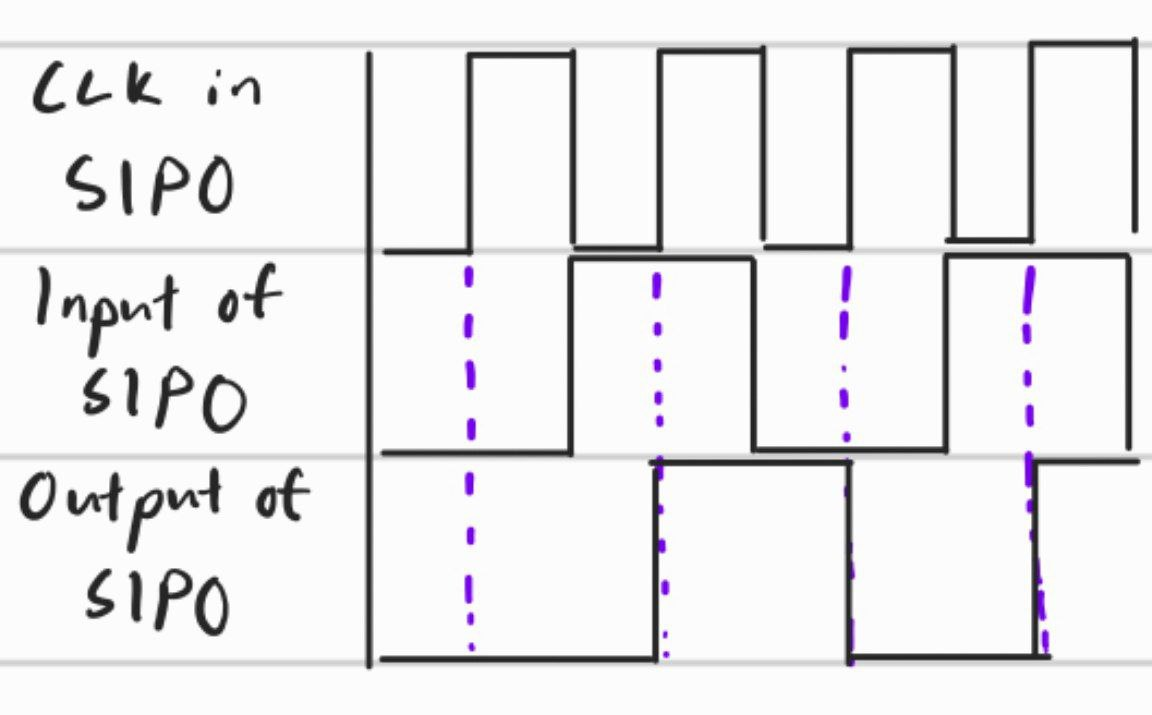


Figure 5.2: Timing diagram of SIPO

**Temporary Memory**: We use a **Serial-In Parallel-Out (SIPO) shift register** to save the data transferred from Lab A to Lab B. The saved data is displayed on an **LED array**, allowing users to confirm that their data has been stored. One LED bulb is connected to the Most Significant Bit (MSB) to indicate the data currently being saved. The user can decide whether to save the data by turning the **enable** function of the SIPO on (1) or off (0). Furthermore, user can **clear** the data saved also.

By implementing this memory system, we realized that the data was not being saved properly. To solve this issue, we increased the **clock speed of the SIPO** to exactly **twice** the speed of the data transfer in Lab A. This adjustment ensures that the data sent from Lab A is saved accurately, although it may introduce some **delay**.

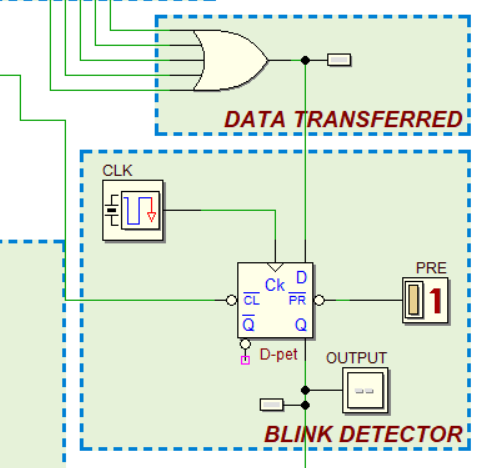


Figure 5.3: Data Transferred and Blink Detector

**Data Transferred**: The 5 outputs from the AND gates in Lab B are connected to a **5-input OR gate** to verify successful data transfer. If the data is successfully transferred, the LED connected to the output of the OR gate will **blink**, indicating the data is constantly changing between 0 and 1. This output is significant as it influences the input value of the **clock enabler**.

**Blink Detector**: Since this affects the **clock enabler**, the output **should not blink**. To stabilize the output, we use a **D flip-flop** to ensure that the output remains consistent, and blinks as intended.

**Validation for selectors**

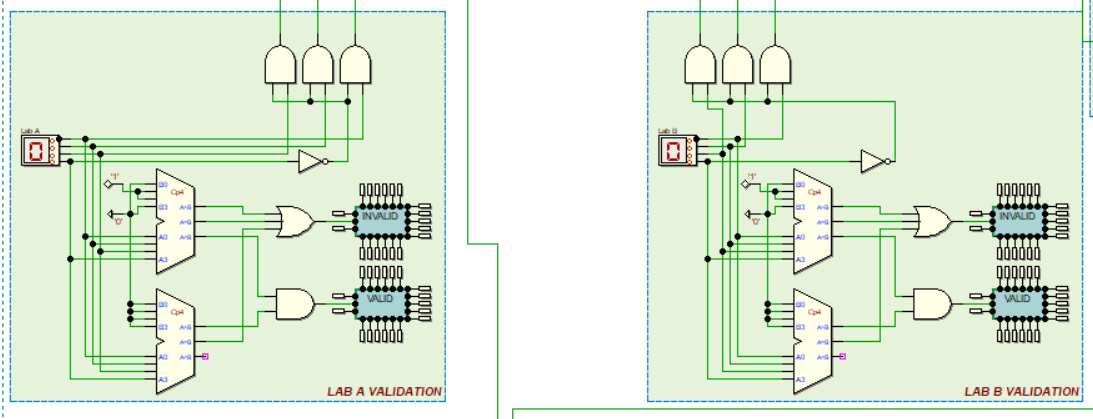


Figure 5.4: Validation for selectors in Lab A and Lab B

Each lab contains only 5 computers, assigned as I1 to I5 in Lab A for the MUX and Q1 to Q5 in Lab B for the DEMUX, users can only choose numbers between 1 and 5.

The **input Hex digit** allows users to easily select the computer they want to use.

* Lab A input Hex digit – The **selector of the MUX** determines which computer in Lab A will send the signal.
* Lab B input Hex digit – The **selector of the DEMUX** determines which computer in Lab B will receive the signal.
* Example: If you want to assign computer 4 in Lab A to send a signal to computer 1 in Lab B, you should select input 4 for the MUX selector and input 1 for the DEMUX selector.
* To prevent invalid inputs for the selectors from the Hex digit to the 3-input selectors of the MUX and DEMUX, we connect the MSB of the Hex digit and the remaining bits (3) to **AND gate**. This ensures only valid inputs are processed, as we only assign 5 computers (represented by binary values from 0001 to 0101), where the MSB is always 0.

The structure for both Lab A and Lab B validation is identical.

**Validation for Selector**: Two comparators are used to validate whether the input is greater than 0 and less than 6. The valid outputs (>0 and <6) are connected to an AND gate to confirm the user has entered a valid selector for the MUX or DEMUX. Any invalid inputs are sent to an OR gate to notify the user that they have entered an invalid selector for the MUX or DEMUX. There will be no values smaller than 0 in input Hex digit, so there is no need to wire it up.

**Password System**

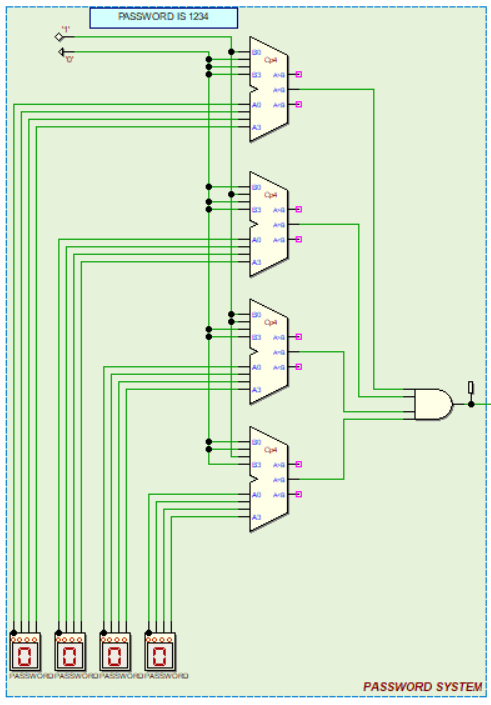
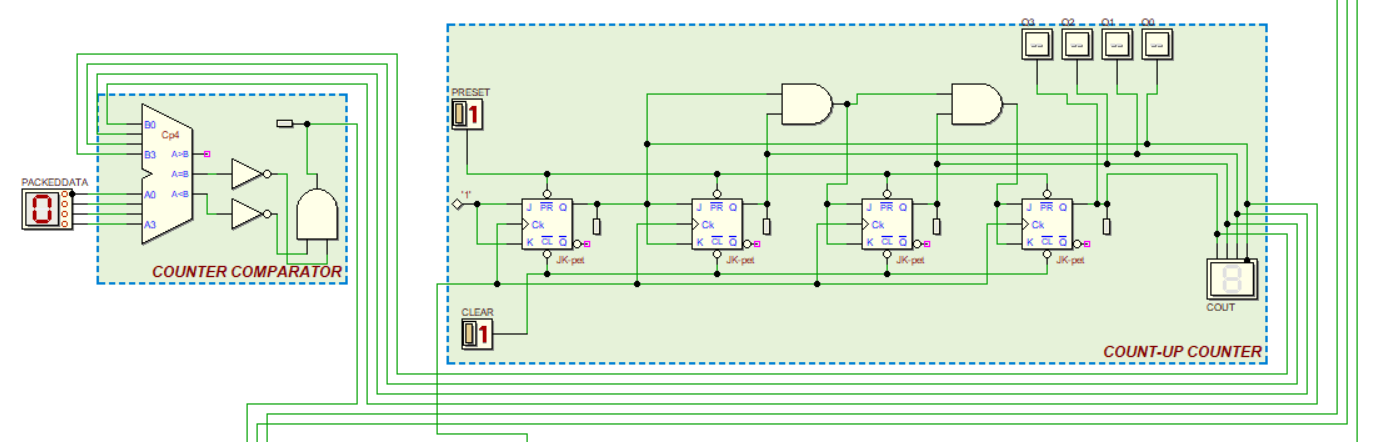


Figure 5.5: Password system

**Password System:** The password system is built using **4 comparators**, with each comparator checking one hexadecimal digit (0-F). The 4 equal outputs from the comparators are connected to a **4-input AND gate** to ensure that all four digits of the input are correct. This design gives many possible combinations, making it easy to create a strong and secure password. The system ensures that only the correct password allows access, protecting the system from unauthorized use. Each comparator checks if the input matches the correct value, and access is granted only when all 4 inputs are correct. This password system also controls the clock enabler. The output from the AND gate affects the **clock enabler**, making sure the system works only when the correct password is entered. This helps keep the system **secure** and reliable.

**Preset Password**: The password for this system is preset to **1234**.

**Counter comparator**

Figure 5.6: Counter Comparator and Count-up Counter

The counter comparator is to control the transfer of Packet Data from LAB A to LAB B. The user selects the number of packets to transfer, ranging from 0 to F (16 packets) using the Packet Data selector. This value represents the maximum number of packets to be sent.

When the transfer progresses, the count-up counter increments with each packet, displaying the current count (COUT). The counter comparator continuously compares the user-defined packet limit with the current count. When the current count is less than the user-defined limit, the indicator bulb lights up, meaning that it is ready to transfer data.

However, when the number of Packet Data is less than or equal to the current count, the bulb turns off, showing that the transfer should stop.

**Count-Up Counter**

Step 1: To design 4-bit up synchronous counter using **JK Flip-Flop (JK FF)**.

Step 2: Draw state diagram

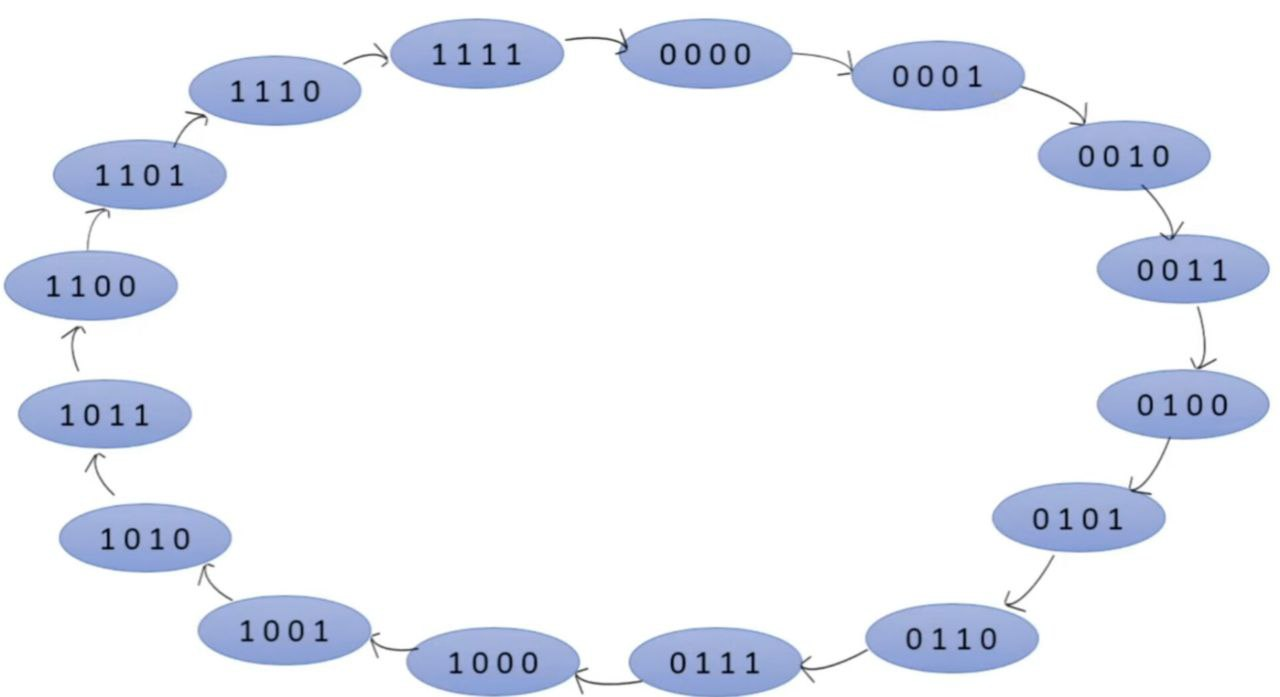


Figure 5.7: State Diagram (Raviraj Pawar, 2024)

Step 3: Next state table

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Present State | | | | Next state | | | |
| Q3 | Q2 | Q1 | Q0 | Q3+1 | Q2+1 | Q1+1 | Q0+1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |

Table 5.1: Next State Table (Raviraj Pawar, 2024)

Step 4: Transition table for JK FF

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Present State | | | | Next state | | | | JK Transition | | | | | | | | JK FF State | | | |
| Q3 | Q2 | Q1 | Q0 | Q3+1 | Q2+1 | Q1+1 | Q0+1 | J0 | K0 | J1 | K1 | J2 | K2 | J3 | K3 | FF3 | FF2 | FF1 | FF0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | X | 0 | X | 0 | X | 0 | X | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | X | 1 | 1 | X | 0 | X | 0 | X | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | X | X | 0 | 0 | X | 0 | X | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | X | 1 | X | 1 | 1 | X | 0 | X | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | X | 0 | X | X | 0 | 0 | X | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | X | 1 | 1 | X | X | 0 | 0 | X | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | X | X | 0 | X | 0 | 0 | X | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | X | 1 | X | 1 | X | 1 | 1 | X | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | X | 0 | X | 0 | X | X | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | X | 1 | 1 | X | 0 | X | X | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | X | X | 0 | 0 | X | X | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | X | 1 | X | 1 | 1 | X | X | 0 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | X | 0 | X | X | 0 | X | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | X | 1 | 1 | X | X | 0 | X | 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | X | X | 0 | X | 0 | X | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | X | 1 | X | 1 | X | 1 | X | 1 | 1 | 1 | 1 | 1 |

Table 5.2: JK Transition table

Step 5: Use K-Map to simplify the equation

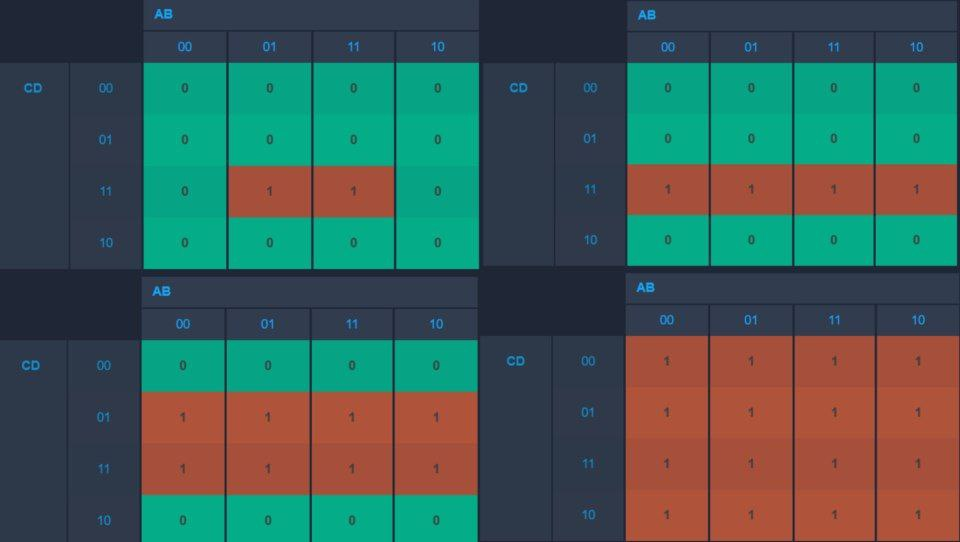


Figure 5.8: K-Map for FF3, FF2, FF1 and FF0 respectively

Let ABCD in Figure 5.9 correspond to Q3, Q2, Q1, and Q0, respectively. Based on this, the equations are as follows:

* FF3 = Q2 • Q1 • Q0
* FF2 = Q1 • Q0
* FF1 = Q0
* FF0 = 1

Step 6: Implement the circuit of the counter

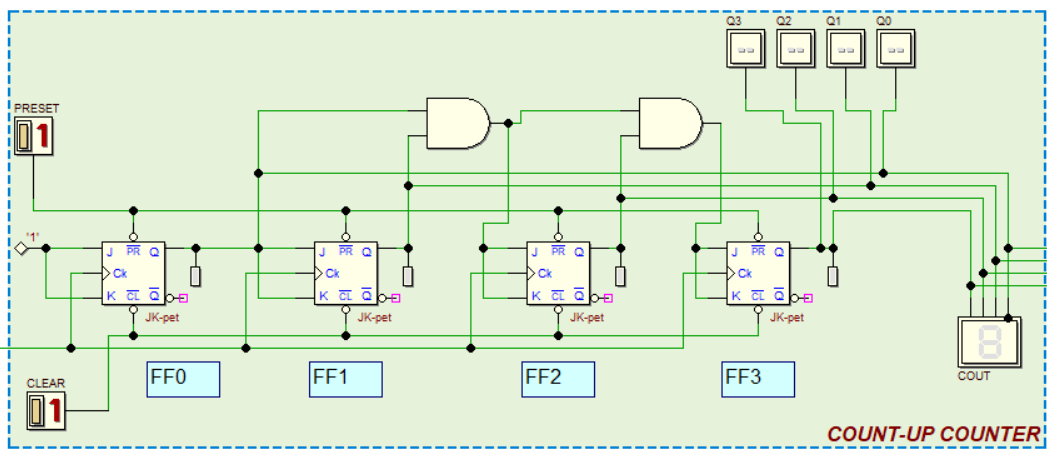
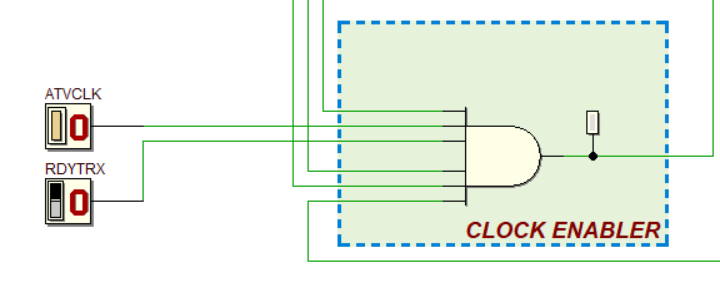


Figure 5.9: Binary 4-bit Synchronous Up Counter with JK Flip-Flop

The first AND gate connects **Q0** and **Q1** as inputs to **FF2**. The second AND gate takes the output from the first AND gate and **Q2** as inputs to **FF3**.

**Clock Enabler**



It ensures that the data transfer process is activated only under specific conditions by gating the clock signal through an AND gate. The inputs to the Clock Enabler include the **ATVCLK** (Active Clock) signal, which indicates that the clock is active and ready, and the **RDYTRX** (Ready to Transfer) signal, which confirms that the system is prepared for data transfer.

The Clock Enabler outputs the clock signal only when both inputs are high (logic 1), enabling the system to function only under the correct conditions.

**Conclusion and Reflection**

This project, Digital Logic Network Packet Transmission Monitoring

References

Abd. Bahrim Yusoff, Mazleena Salleh, Mohd Fo’ad Rohani, & Isamail Fauzi Isnin. (2024). *Digital Logic* (5th ed.). Faculty of Computing, Universiti Teknologi Malaysia.

**Appendices**

1. **Digital Circuit Simulator (Deeds):**

<https://www.digitalelectronicsdeeds.com/>

1. **Block Diagram & Flowchart Generator（smartdraw):**

<https://app.smartdraw.com/>

1. **Presentation Video (Record) :**

?(youtube link)

1. **Group Meeting Picture :**

?

<https://www.youtube.com/watch?v=mn5KnK9KrSU> this is reference do the APA

[jkfftransitiontable](https://www.google.com/url?sa=i&url=https%3A%2F%2Feees.in%2Fsynchronous-counters-using-jk-flipflop%2F&psig=AOvVaw25CdktxieKKXA2jz8jgWe1&ust=1737836536082000&source=images&cd=vfe&opi=89978449&ved=0CBQQjRxqFwoTCNjoi96Xj4sDFQAAAAAdAAAAABAJ)