**EXPERIMENT NO: 07**

**TITLE OF THE EXPERIMENT:** Shift Register

**AIM:** Design and investigation of all types of Shift Register

**APPARATUS/COMPONENTS REQUIRED:**

|  |  |  |  |
| --- | --- | --- | --- |
| Sl. no. | Name of the equipment /component | Specification | Quantity |
| 1 | Trainer board | Digital | 1 no |
| 2 | J-K Flip-Flop | IC 7476 | 2 no |
| 3 | Not gate | IC 7404 | 1 no |
| 4 | Connective wires | Hook up | As required |

**THEORY:** Shift registers are a type of logic circuits used basically for the storage and transfer of digital data. Storing of binary bit is done with the help of Flip-Flops. The storage capacity of a register is define as a no of bits of digital data it can store and retain. These registers are normally used for temporarily storage of data. The shifting of data from one registers to the other or within the registers can takes place in the following 4 ways.

1. Serial in Serial out (SISO)
2. Serial in Parallel out (SIPO)
3. Parallel in Serial out (PISO)
4. Parallel in Parallel out (PIPO)

**Serial-in Serial-out Shift Registers:** This type of shift registers accepts data serially, i.e. one bit at a time, and also outputs data serially.

The logical diagram of a 4-bit Serial-in, serial-out, shift-right, shift register is shown in figure below with four stages, i.e. four FFs, the register can store up to four bits of data.

|  |
| --- |
| Logic diagram: Serial-in Serial-out (SISO) Shift Registers using J-K Flip-Flop |
|  |

**Serial-in Parallel-out Shift Registers:** In this type of register, the data bits are entered into the register serially, but the data stored in the register is shifted out in parallel form.

Once the data bits are stored each bit appears on its respective output line and all bits are available simultaneously, rather than on a bit-by-bit basis as with the serial output.

|  |
| --- |
| Logic diagram: Serial-in Parallel-out(SIPO) Shift Register using D Flip-Flop |
|  |

**Parallel-in Serial-out Shift Register:** In such registers the data bits are entered simultaneously into their respective stages on parallel lines and not a bit-by-bit basis on one line as was the case with serial-in registers. Serial outputs of the registers are taken out in the same way as we explained for the serial-in and serial-output. After the data bits are properly stored once in the register the data bits are taken out one by one as was the case with serial-in serial-out registers.

|  |
| --- |
| Logic diagram: Parallel-in Serial-out(PISO) Shift Register using D Flip-Flop |
|  |

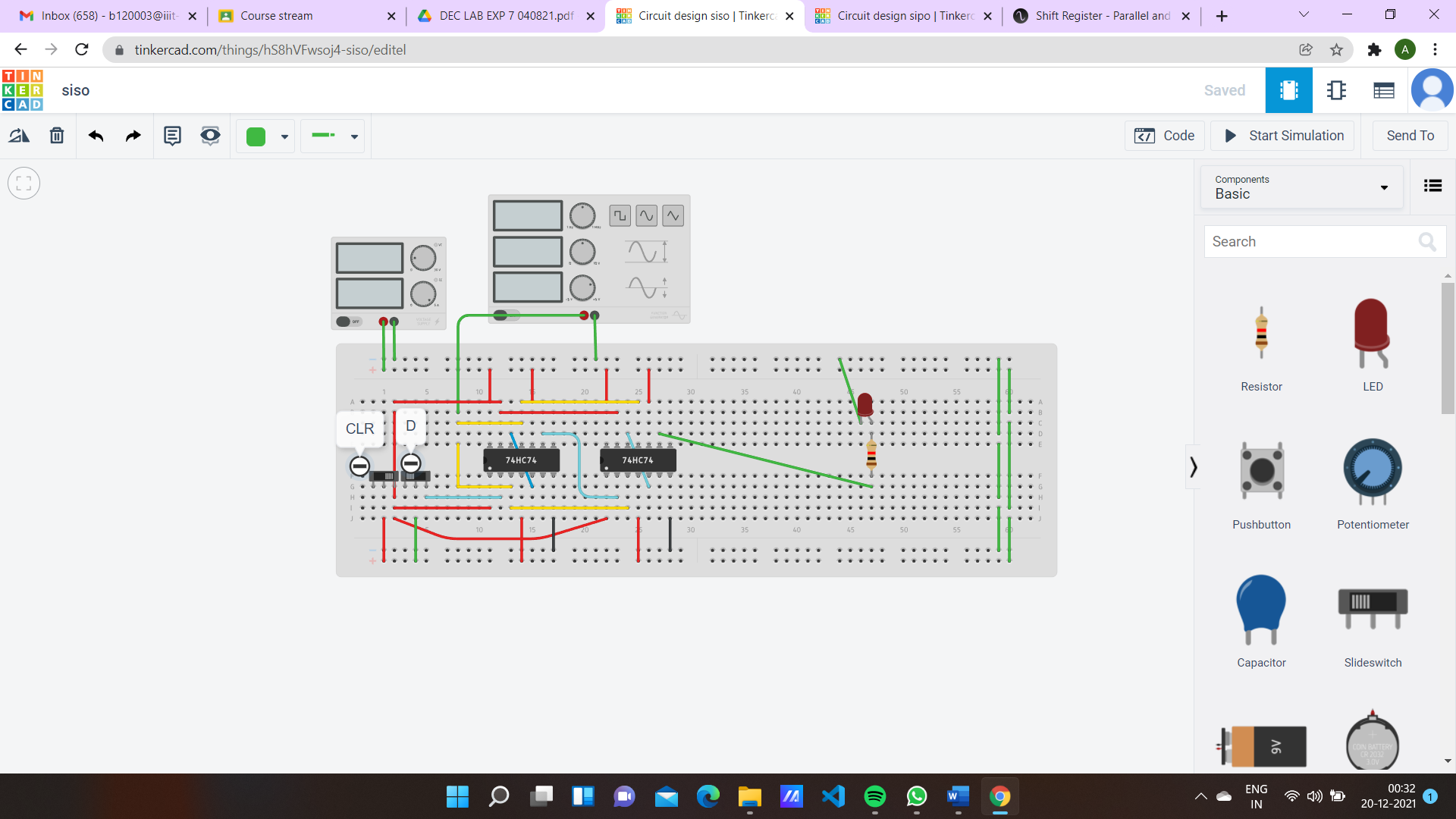
**Parallel-in Parallel-out Shift**

**Register:** In a Parallel-in, Parallel-out, shift register, the data is entered to the register in parallel form, and also the data is taken out of the register in parallel form. Immediately following the simultaneous entry of all data bits, the bits appear on the parallel outputs.

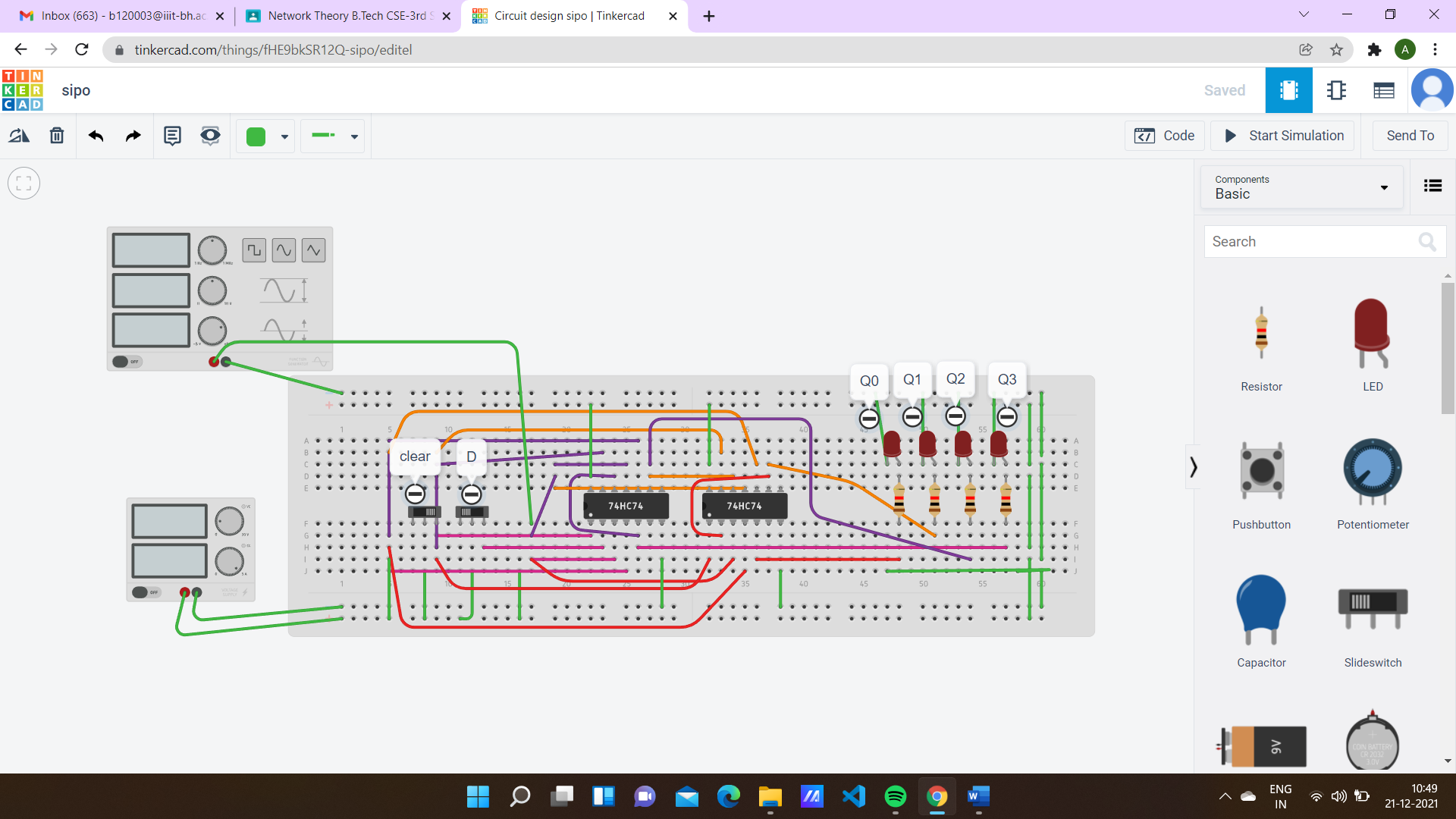
|  |
| --- |
| Logic diagram: Parallel-in Parallel-out (PIPO) Shift Register using J-K Flip-Flop |
|  |

**CIRCUIT DIAGRAMS:**

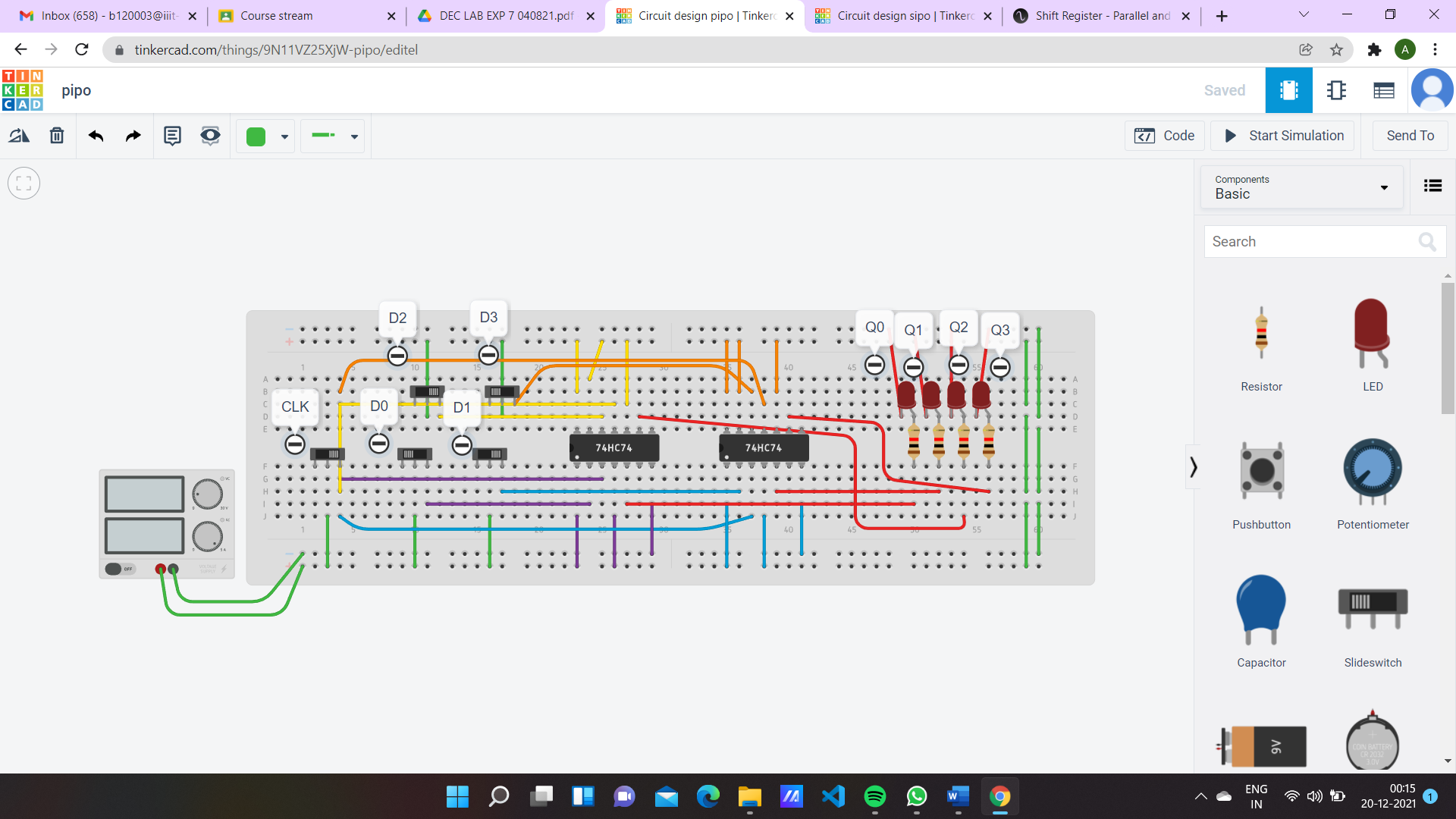
**Serial-in Serial-out Shift Registers:**



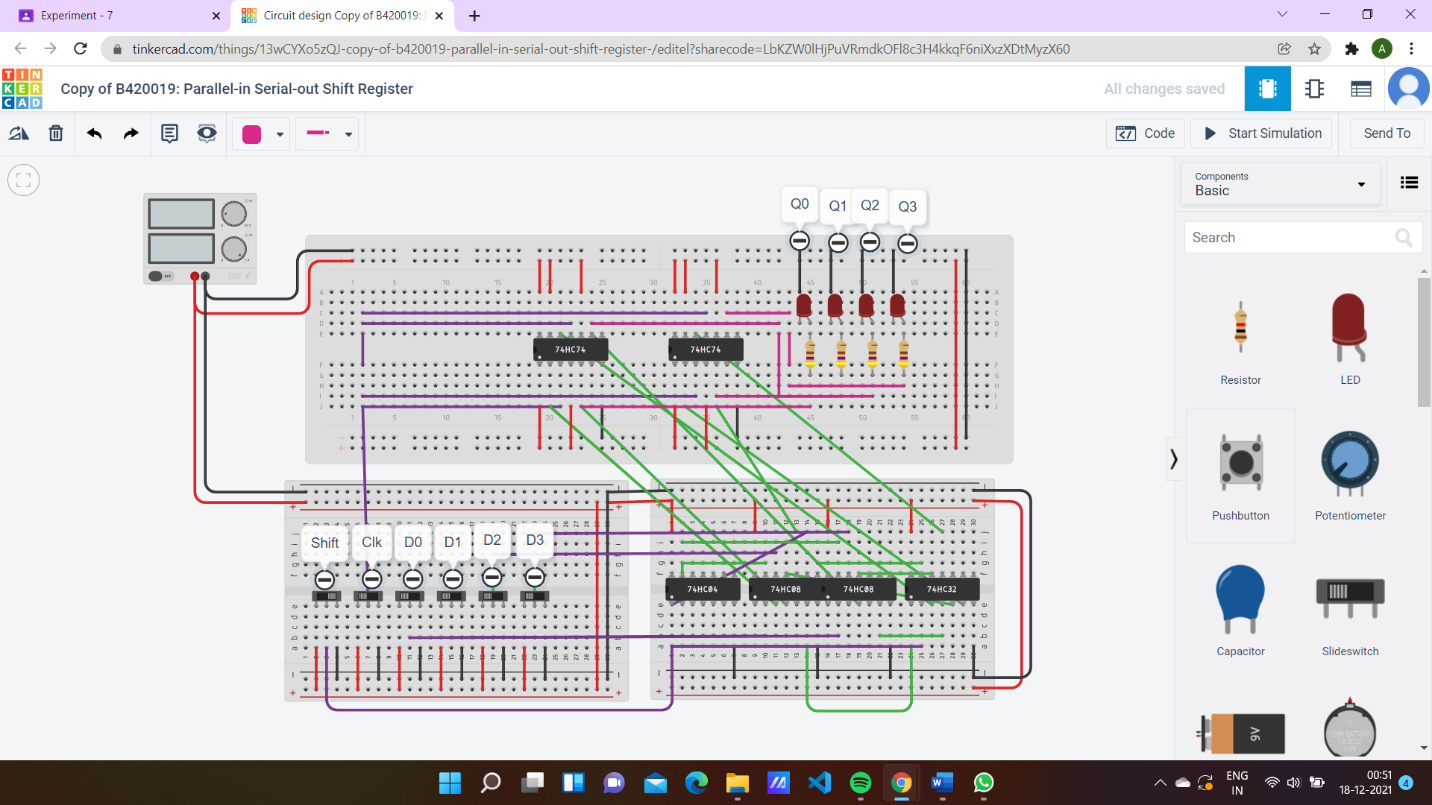
**Serial-in Parallel-out Shift Registers:**



**Parallel-in Parallel-out Shift Register:**



**Parallel-in Serial-out Shift Register:**



**OBSERVATION:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Clock Pulse |  | Serial-in Serial-out (SISO) | | | Serial-in Parallel-out (SIPO) | | | |
|  | Output | | | Output | | | |
| Q3 | Q2 | Q1 | Q0 | Q3 | Q2 | Q1 | Q0 |
| Initially | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1st Falling edge | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 2nd Falling edge | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 3rd Falling edge | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| 4th Falling edge | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

\*Apply input data serially and find the output.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | Observation table: Parallel-in Parallel-out (PIPO) | | | | | |  |
| Clock pulse |  | Input | | | Output | | |  |
| D3 | D2 | D1 | D0 | Q3 | Q2 | Q1 | Q0 |
| Initially | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 1st  Falling edge | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| 2nd  Falling edge | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 3rd  Falling edge | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| 4th  Falling edge | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Observation table: Parallel-in Serial-out (PISO) | | |  |  |  |
| Clock Pulse | Shift/Load’ | Output |  |  |  |
| Q3 | Q2 | Q1 | Q0 |
| Initially | 0 | 0 | 0 | 0 | 0 |
| 1st Falling edge | 1 | 1 | 0 | 0 | 0 |
| 2nd Falling edge | 1 | 1 | 1 | 0 | 0 |
| 3rd Falling edge | 1 | 0 | 1 | 1 | 0 |
| 4th Falling edge | 1 | 1 | 0 | 1 | 1 |

* Enter the input data simultaneously or parallelly and apply the clock.

**PROCEDURE:**

* 1. Design the logic circuit.
  2. Switch off the trainer board before assembling the circuit and doing the connection.
  3. Assemble the circuit on the breadboard as per the circuit diagram.
  4. Connect the positive supply and ground correctly to the circuit.
  5. Provide the logic inputs required to verify the truth table.
  6. Check whether all connections are correct before switch on the trainer board.
  7. Switch on the trainer board.
  8. Observe the output and verify the truth table according to the observation table.

**CONCLUSION:**

From this we learnt how to design and investigation of all types of Shift Register by using the tinkercad software

Name: ADITI ROUT

SID: B120003

Signature of Faculty Branch: CSE