EXPERIMENT NO: 9

Title: To implement :i) Serial In Parallel Out (SIPO) register and ii) Johnson counterusing simulator

EXPERIMENT NO: 9

Simulate a Serial In Parallel Out (SIPO) register and Johnson counter

Name:Hayden Cordeiro SECOMPS RollNo:05

AIM	To implement a Serial In Parallel Out (SIPO) register and Johnson counter using simulator
LEARNINGOBJECTIVE	To explore a simulation tool for computer organization components.
LEARNINGOUTCOME	Students can simulate the operation of multiplier unit .
LAB OUTCOME	CSL403.2: Ability to estimate the output of computer hardware operations using simulator.
PROGRAM OUTCOME	PO11, PO32, PO41, PO52, PO83, PO9-3,PO12-2,PSO11
BLOOM'STAXONOMYLEVEL	Evaluate
THEORY	Note for students: StudentsaresupposetowritetheoryrelatedtoSerialInParallelOut(SIPO)registerandJohnsoncounterandexplainitsworking. **SIPO:** A serial-in, parallel-out shift register is similar to the serial-in, serial-out shift register in that it shifts data into internal storage elements and shifts data out at the serial-out, data-out, pin. It is different in that it makes all the internal stages available as outputs. Therefore, a serial-in, parallel-out shift register converts data from serial format to parallel format. Working: In Serial In Parallel Out (SIPO) shift registers, the data is stored into the register serially while it is retrieved from it in parallel-fashion .The data word which is to be stored (Data in) is fed serially at the input of the first flip-flop (D1 of FF1). It is also seen that the inputs of all other flip-

flops (except the first flip-flop FF1) are driven by the outputs of the preceding ones say for example, the input of FF2 is driven by the output of FF1. In this kind of shift register, the data stored within the register is obtained as a parallel-output data word (Data out) at the individual output pins of the flip-flops (Q1 to Qn).In general, the register contents are cleared by applying high on the clear pins of all the flip-flops at the initial stage. After this, the first bit, B1 of the input data word is fed at the D1 pin of FF1.This bit (B1) will enter into FF1, get stored and thereby appears at its output Q1 on the appearance of first leading edge of the clock. Further at the second clock tick, the bit B1 right-shifts and gets stored into FF2 while appearing at its output pin Q2 while a new bit, B2 enters into FF1. Similarly at each clock tick the data within the register moves towards right by a single bit while a new bit of the input word enters into the register. Meanwhile one can extract the bits stored within the register in parallel-fashion at the individual flip-flop outputs.

JOHNSON COUNTER:

Johnson Counter. A Johnson counter is a modified ring counter in which the output from the last flip flop is inverted and fed back as an input to the first. It is also called as Inverse Feedback Counter or Twisted Ring Counter.

A Johnson counter is a modified ring counter, where the inverted output from the last flip flop is connected to the input to the first. The register cycles through a sequence of bit-patterns. The MOD of the Johnson counter is 2n if n flip-flops are used. The main advantage of the Johnson counter counter is that it only needs half the number of flip-flops compared to the standard ring counter for the same MOD.

It can be implemented using D-type flip-flops (or JK-type flip-flops).

Working:

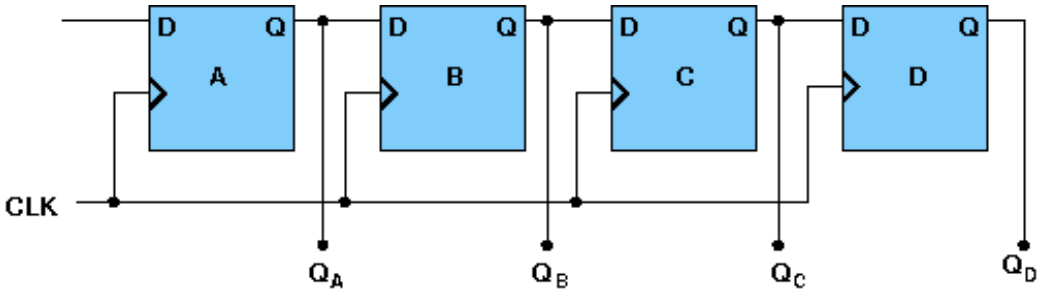
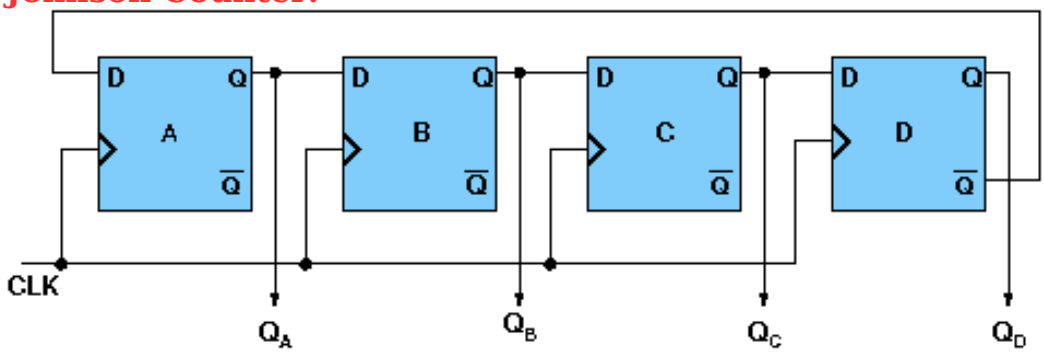
The default state of Johnson counter is 0000 thus before starting the clock input we need to clear the counter using clear input.

Whenever a clock edge hits the counter the output of each flip-flop will transfer to the next stage (flip-flop) but the inverted output of the last flip-flop will shift to the first stage making the state 1000.

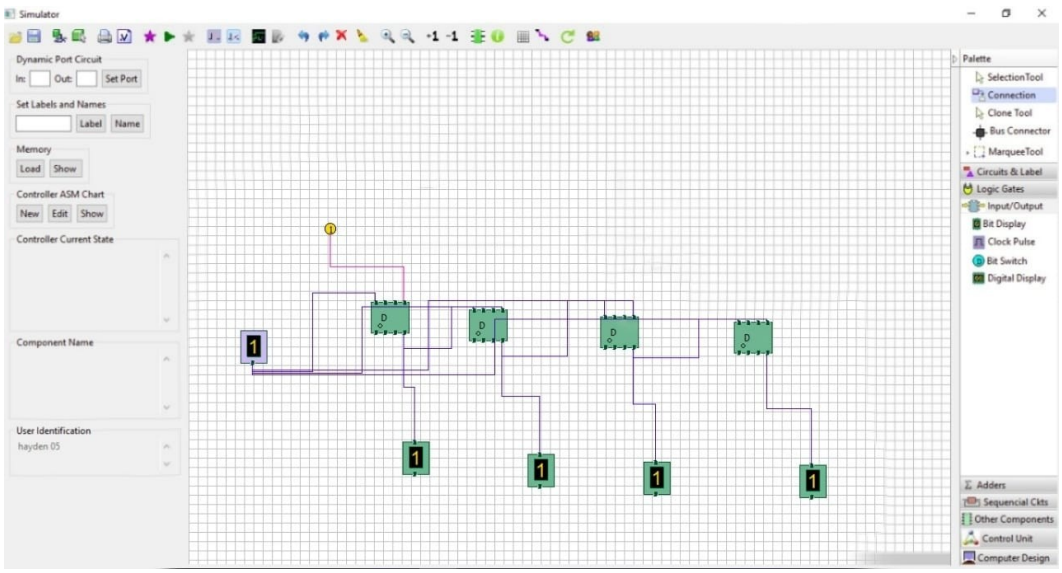
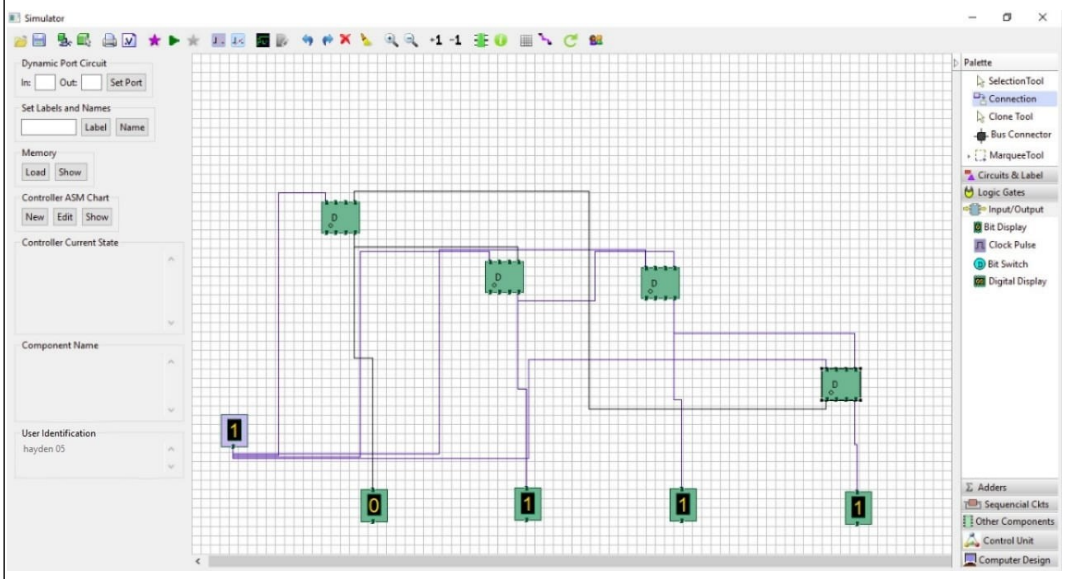
Upon next clock cycle, another ‘1’ will stack in from the left side as the inverted output of the last stage will be shifted to the first stage.

On next clock cycle, another ‘1’ will add in from left until the state becomes 1111.

Now that the last flip-flop’s output is ‘1’, the next clock cycle will shift the invert of the last flip-flop which is ‘0’ into the first flip-

	flop. It will result in stacking ‘0’ from the left side. This stacking of the first 0 will make the state 1111 into 0111. The next coming clock cycles will stack in 0’s from the left making the states 0011, 0001 & 0000 with each clock cycle. Eventually, it reaches its default state and it starts from the beginning again.
CIRCUITDIAGRAM	**SIPO register:** 
	Johnson Counter: 
COMPONENTS USED	**To build any register or counters, we need :** 1. FlipFlops. 2. LogicGates. 3. Wires toconnect.

STEPS TODESIGN THECIRCUIT	Startthesimulatorasdirected.Thissimulatorsupports5valuedlogic. 1. Todesigna4bitshiftregister(rightshift),weneed4MSDflipflop, 1freerunningclock,1Bitswitch(whichwillactasinputtotheleft mostflipflop),4Bitdisplay(toseetheoutputofindividualflipflopssoth attheshiftingcanbeseenwiththeclockinput),wires. 2. TheMSDflipflopcomponentisinthesequentialcircuitdrawerin thepallet.Thepinconfigurationisshownwheneverthemouseishover edonanycannedcomponentofthepaletteorpressthe'showpinconfig 'button.Pinnumberingstartsfrom1andfromthe bottomleftcorner(indicatingwiththecircle)andincreases anticlockwise. 3. ForMSDflipflopinputisinpin5,output(Q)isinpin4,clockisinpin8 4. clickontheMSDflipflopcomponentinthepalletandthenclickonthepo sitionoftheeditorwindowwhereyouwanttoaddthecomponent(nodr aganddrop,simpleclickwillservethepurpose),likewiseadd4MSDflip flops,1freerunningclock,1Bitswitchand 4bitDisplays(fromDisplayandInputdrawerofthepallet,ifitisnotseen scrolldowninthedrawer) 5. ToconnectanytwocomponentsselecttheConnectionmenuofPalette ,andthenclickontheSourceterminalandclickonthetargetterminal.c onnectallthecomponents,connecttheclocktothepin-8ofalltheMSDflipflops,connectabitswitchtothepin 5(Q)oftheleftmostMSDflipflop,connect4bitdisplaystothe pin-4of4MSDflipflops,connecttheQoutputofthepreviousflipfloptotheD(pin5)inputofthenextflipflop.

	6. Toseethecircuitworking,clickontheSelectiontoolinthepallet thengiveinputbydoubleclickingonthebitswitch,totheleftmost Dflipflopatpi-5(letitbe1),starttheclocknowchecktheoutputandseehowthe1isshifting fromlefttoright. 7. Similarwaycountercanbeimplemented.
SIMULATEDRESULTS	**Johnsons counter:**  **SERIAL IN PARALLEL OUT :** 

CONCLUSION	We have successfully implemented and simulated serial in parallel out register and johnsons counter
REFERENCES	**https://cse.iitkgp.ac.in/~chitta/coldvl/rca_design.html**