FR. Conceicao Rodrigues College of Engineering
Department of Computer Engineering

Experiment No 9: Design of Control unit

Course, Subject & Experiment Details

Academic Year	2023-24	Estimated Time	Experiment No. 9– 02 Hours
Course & Semester	S.E. (Computers) – Sem. III	Subject Name	Digital Logic & Computer Organization and Architecture
Chapter No.	4	Chapter Title	Control Unit Design
Experiment Type	Software	Subject Code	CSC304

Rubrics

Timeline (2)	Practical Skill & Applied Knowledge (4)	Output (4)	Total (10)

1. Aim of Experiment

To Design and Implement controller for Booth's Multiplier to generate the required control signals using Finite State Machine.

2. Objective of Experiment

1. Understanding behaviour of Booth's multiplication algorithm from working module and the module designed by the student as part of the experiment
2. Designing Booth's multiplier with a controller and a datapath. This will also help in the learning of control unit design as a finite state machine
3. Understanding the advantages of Booth's multiplier
    - It can handle signed integers in 2's complement notion
    - It decreases the number of addition and subtraction
    - It requires less hardware than combinational multiplier
    - It is faster than straightforward sequential multiplier

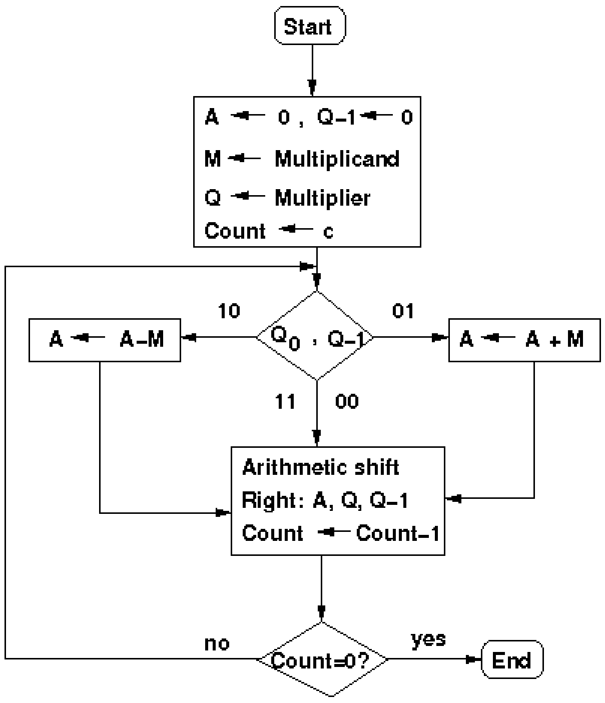
3. Software Required

A) Virtual Lab.

4. Brief Theoretical Description

Booth's multiplication algorithm is an algorithm which multiplies 2 signed integers in 2's complement. The algorithm is depicted in the following figure with a brief description. This approach uses fewer additions and subtractions than more straightforward algorithms.

The multiplicand and multiplier are placed in the m and Q registers respectively. A 1 bit register is placed logically to the right of the LSB (least significant bit) Q0 of Q register. This is denoted by Q-1. A and Q-1 are initially set to 0. Control logic checks the two bits Q0 and Q-1. If the twi bits are same (00 or 11) then all of the bits of A, Q, Q-1 are shifted 1 bit to the right. If they are not the same and if the combination is 10 then the multiplicand is subtracted from A and if the combination is 01 then the multiplicand is added with A. In both the cases results are stored in A, and after the addition or subtraction operation, A, Q, Q-1 are right shifted. The shifting is the arithmetic right shift operation where the left most bit namely, A_{n-1} is not only shifted into A_{n-2} but also remains in A_{n-1}. This is to preserve the sign of the number in A and Q. The result of the multiplication will appear in the A and Q.



5. Simulation

A.*Virtual Lab:*

The virtual laboratory is an interactive environment for creating and conducting simulated experiments: a playground for experimentation. It consists of domain-dependent simulation programs, experimental units called objects that encompass data files, tools that operate on the objects.

The objective is to expose the students to various key aspects of Digital Logic and computer organization by enabling them to perform FPGA based prototyping of experiments with support of a virtual environment

Procedure to perform the experiment:Design of Control Unit

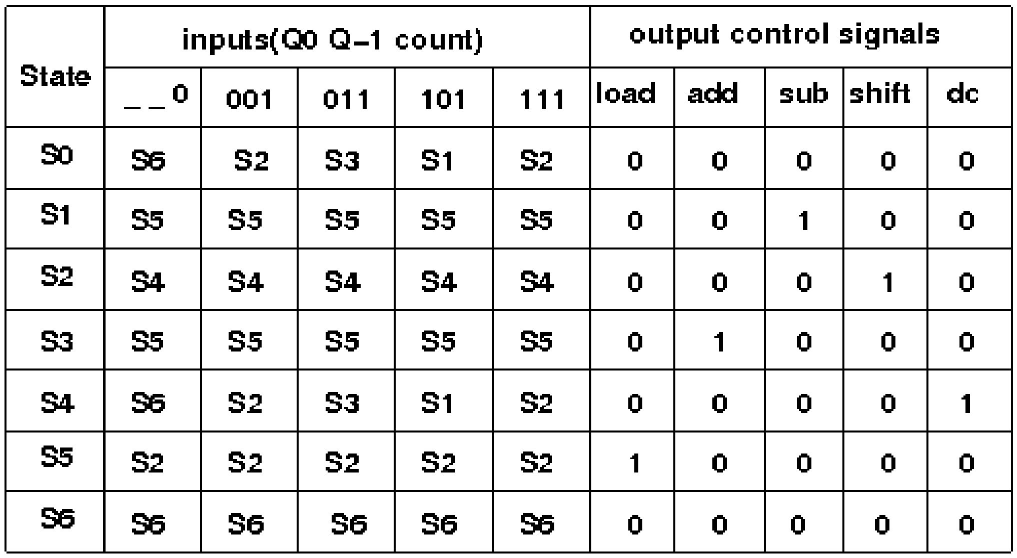
1. Start the simulator as directed.This simulator supports 5-valuedlogic.
2. To perform the experiment on the given modules, we need the datapath

specified for booth's multiplication, a controller with a specified state chart, a clock input, bitswitch (to give input, which will toggle its value with a double click), bit displays (for seeing output),wires.

3. Instantiating the controller: A control unit can be seen as a finite state machine, so its behavior can be represented in a state table. The controller of the simulator accepts the Moore type state chart and must contain an end state. State names will automatically be generated in the form of S. In the left pane of the simulator, click on the ASM chart button in the controller subsection. Give the required informations in the appeared form as follows:

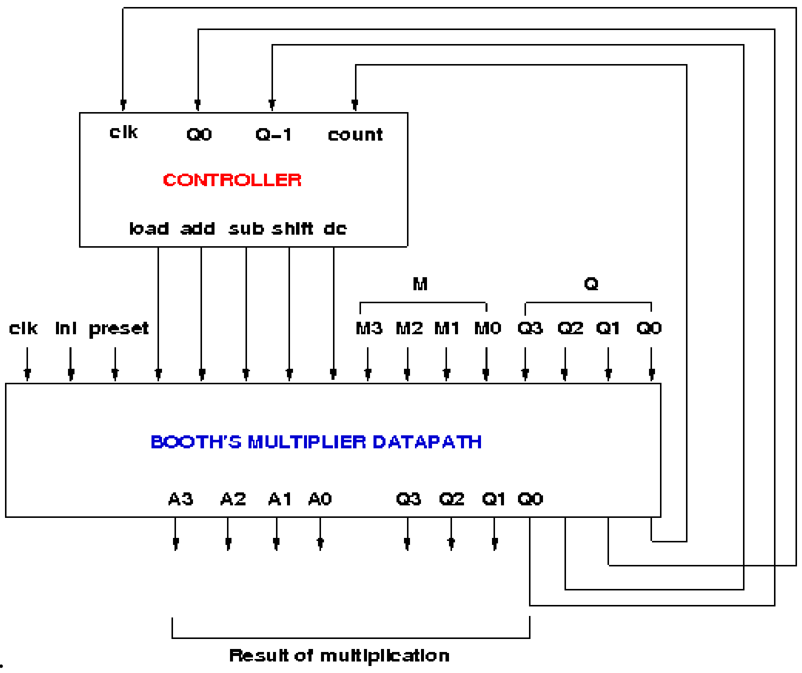
- Number of states: 7
- Number of inputs: 3
- Number of outputs: 5

The controller will generate 5 output control signals. After entering these informations, the second form will appear where you can set the names of the inputs and outputs. Here inputs are the Q0, Q-1, count. Outputs control signals are load, add, sub, shift, dc. The order of given input/outputs are maintained while creating terminals of the controller. for example, the first output signal will appear in the left most output terminal (lower terminals), second output will appear in the second left most bit and so on. In case of input terminals, the left most bit is for clock input, so the first input appears in the second terminal, and then the order is maintained. Then the third form will appear where you actually specify the state chart i.e. state, outputs of that state and transition conditions . The fields of the chart will be generated dynamically according to previously given information on states, inputs and outputs.After entering the following state chart, click on the controller component in the palette of the simulator then click on the position of the design editor where you want to put the component (no drag and drop, simple click will serve the purpose).



State	inputs(Q0 Q-1 count)					output control signals				
	_ _ 0	001	011	101	111	load	add	sub	shift	dc
S0	S6	S2	S3	S1	S2	0	0	0	0	0
S1	S5	S5	S5	S5	S5	0	0	1	0	0
S2	S4	S4	S4	S4	S4	0	0	0	1	0
S3	S5	S5	S5	S5	S5	0	1	0	0	0
S4	S6	S2	S3	S1	S2	0	0	0	0	1
S5	S2	S2	S2	S2	S2	1	0	0	0	0
S6	S6	S6	S6	S6	S6	0	0	0	0	0

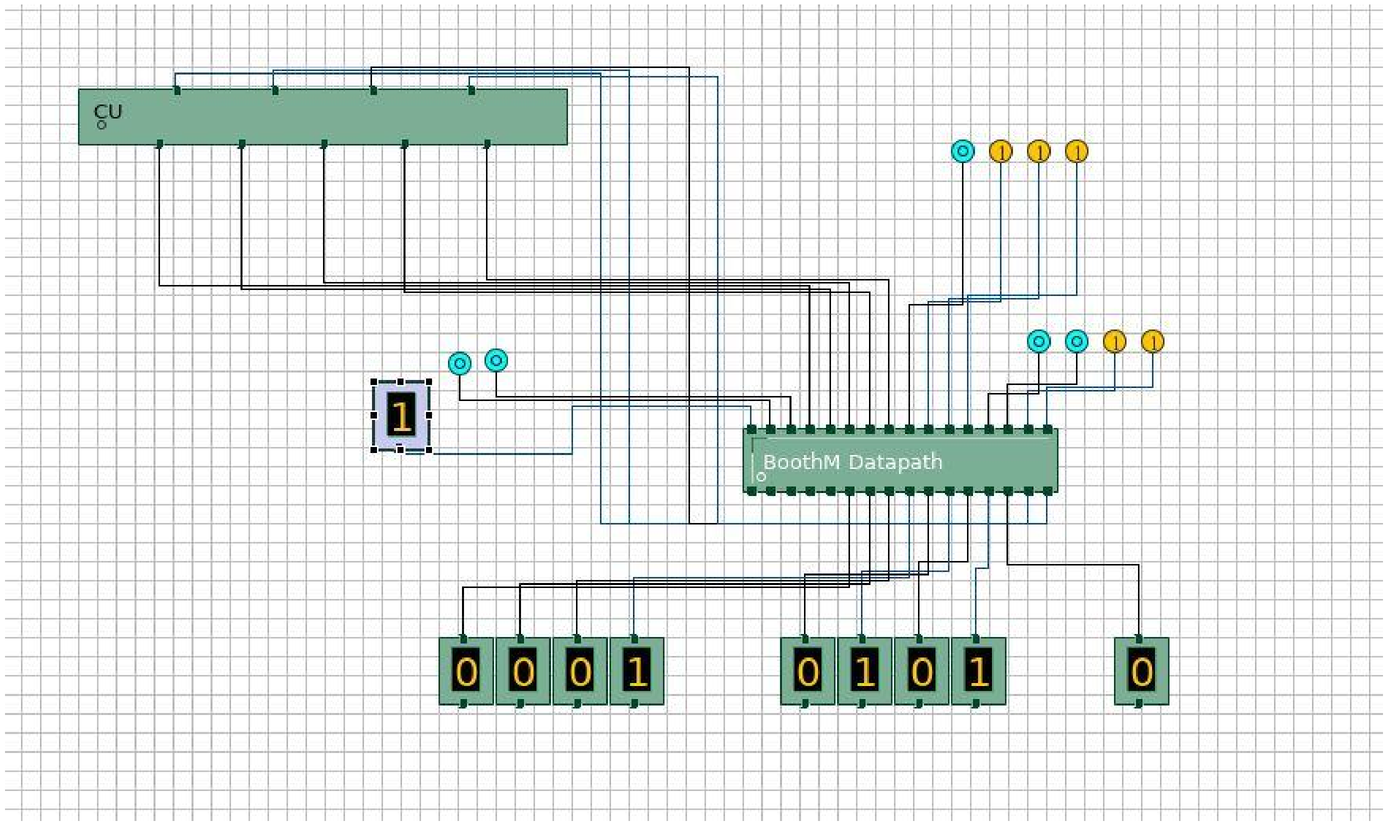
3. Instantiate the Booth's multiplier datapath from the sequential ckt drawer in thepalette (by clicking as mentionedpreviously).
4. The pin configuration of the component is shown whenever the mouse is hovered onany canned component of the palette or pressing the *show pin configuration* button on the toolbar will show it constantly in the left pane. Pin numbering starts from 1 and from the bottom left corner(indicating with the circle) and increasesanticlockwise.
5. Pin configuration of the datapathmodule:
    - M : Multiplicand (4 bit), Q : Multiplier (4bit)
    - Initialization : Inl:1, preset:1, set M, Q, start clock
    - Starting multiplication: Inl:0, preset:0, startclock
    - Result: FQ0 to FA3, at end state (here it is S6). These are the content ofA(4bit) and Q(4bit) register, total 8 bit (FQ0 is LSB, FA3 isMSB)
    - I/P:
        - Clk:32, Inl:31,preset:30
        - Control pins: load:29, add:28, sub:27, shift:26,dc:25
        - Multiplicand: M3 : 24, M2 : 23, M1 : 22, M0 :21
        - Multiplier: Q3 : 20, Q2 : 19, Q1 : 18, Q0 :17
    - O/P: FQ-1 is output of Q-1 bit register, similarly FQ0 to FQ3 are for Qregister and FA0 to FA3 are for A register.
        - Datapath to controller input: Count, clkToController, FQ-1,FQ0
        - Count : 16, clkToController : 15, FQ-1 : 14, FQ0 : 13
        - FQ1 : 12, FQ2 : 11, FQ3 :10
        - FA0 : 9, FA1 : 8, FA2 : 7, FA3 :6
6. To connect any two components select the Connection menu of Palette, and then clickon the Source terminal and click on the target terminal. According to the following diagram connect all the components. Connect the controller outputs to the specified control input terminals of the datapath, specified datapath outputs to the inputs of the controller, the clock input, Bit switches with the inputs and Bit displays component with the outputs (from Display and Input drawer of the pallet,if it is not seen scroll down in the drawer). After the connection is over click the selection tool in



thepallete.

7. At first initialize the multiplier by giving the specified inputs specified earlier, this will load the multiplier and multiplicand, then start the multiplication operation by givingthe specified inputs specified earlier. At the end state (S6), the multiplication result will be seen through ports FQ0 to FA3 (FQ0 is LSB, FA3 is MSB). The current state of the controller is shown in the left pane as it transits from one state to another. The controller can be reset by clicking the *reset controller* button in the top toolbar, to start with a new input.

6. Attach the screen shots of the implemented circuits.



7. Conclusion:

Thus the required control signals are generated using Finite State Machine.