University of Texas at Austin

Digital Logic Design

Lab 7: VHDL for System Design

Joshua Dong
jid295

December 3, 2016

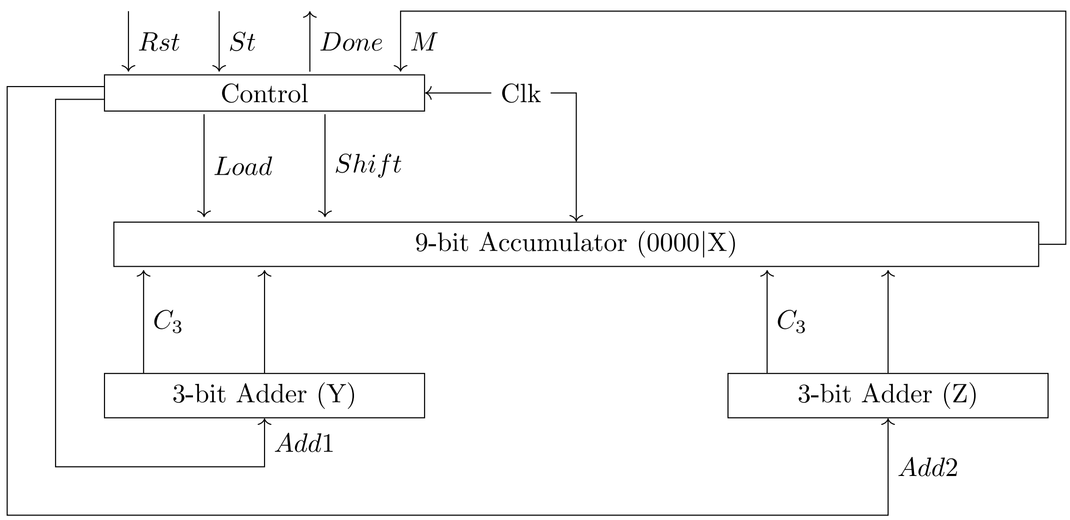
Contents

Problem Statement

Design an arithmetic unit that computes W = X*Y + Z, where **X is 5 bits, Y is 3 bits and Z is 3 bits.** Assume that the start signal (St) is 1 exactly for one clock cycle. When St is '1', in the first clock cycle, the Multiplier (X) should be loaded from the bus. In the second clock cycle, the Multiplicand (Y) should be loaded from the same bus. Finally, in the third clock cycle, Z (the term to be added) should be loaded. Then the state machine should multiply X by Y. Use a 9-bit accumulator, and design the multiplier without using a counter. Use the overloaded addition operator to add. Use a second adder to add Z to X*Y and store the result in the accumulator using a fourth load signal.

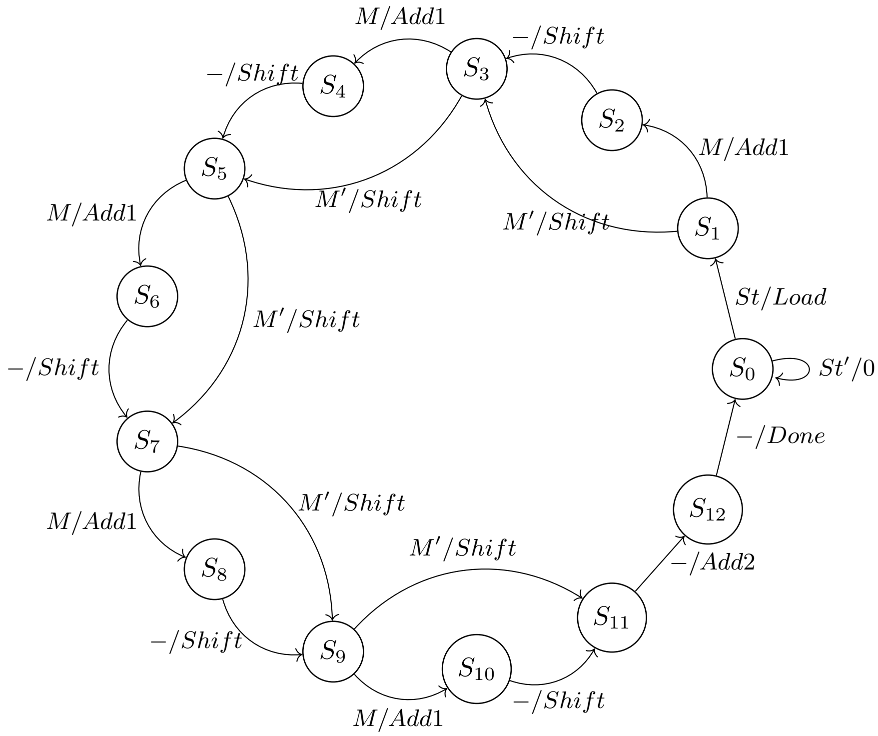
Block Diagram

The following block diagram of the system shows registers, adders, MUXes, and other components. Specified are the used signals, size of registers, adders, etc. The design includes an active-high asynchronous RESET for your design.



State Graph

Below is the state graph for the control circuit.



VHDL Code

Below is simply an implementation of the state graph:

```
library ieee;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;
use IEEE.std_logic_arith.all;
use IEEE.numeric_std.all;


entity SM is
    port(rst, clk, X: in std_logic;
          Q1, Q2, Q3, Z: out std_logic);
end SM;

architecture SM_a of SM is
    signal state, nextstate: integer range 0 to 12 := 0;
begin
    process (state, X)
    begin
        case state is
            -- default to no action
            Load <= '0';
            Shift <= '0';
            Add1 <= '0';
            Add2 <= '0';
            Done <= '0';
```

```
when 0 =>
    if (St = '0') then
        nextstate <= 0;
    else
        Load <= '1';
        nextstate <= 1;
    end if;
when 1 =>
    if (M = '1') then
        Add1 <= '1';
        nextstate <= 2;
    else
        Shift <= '1';
        nextstate <= 3;
    end if;
 when 2 =>
    Shift = '1';
    nextstate <= 3;
when 3 =>
    if (M = '1') then
        Add1 <= '1';
        nextstate <= 4;
    else
        Shift <= '1';
        nextstate <= 5;
    end if;
 when 4 =>
    Shift = '1';
    nextstate <= 5;
when 5 =>
    if (M = '1') then
        Add1 <= '1';
        nextstate <= 6;
    else
        Shift <= '1';
        nextstate <= 7;
    end if;
 when 6 =>
    Shift = '1';
    nextstate <= 7;
when 7 =>
    if (M = '1') then
        Add1 <= '1';
        nextstate <= 8;
    else
        Shift <= '1';
        nextstate <= 9;
    end if;
 when 8 =>
    Shift = '1';
    nextstate <= 9;
when 9 =>
    if (M = '1') then
```

```
                    Add1 <= '1';
                    nextstate <= 10;
                else
                    Shift <= '1';
                    nextstate <= 11;
                end if;
            when 10 =>
                Shift = '1';
                nextstate <= 11;
            when 11 =>
                Done <= '1';
                nextstate <= 11;
            when 12 =>
                Add2 <= '1';
                nextstate <= 0;
        end case;
    end process;

    process (clk, rst)
    begin
        -- active-low asynchronous clear
        if (rst = '0') then
            state <= 0;
        -- rising edge trigger
        elsif rising_edge(clk) then
            state <= nextstate;
        end if;
    end process;
end SM_a;
```

ctrl.vhdl

Waveforms

HDL Synthesis Report

No latch was identified, so below is the HDL Synthesis Report page.

XDC file