CS203 Project Report

Group 16 : Shivam Prasad 2017CSB1108
Jeetu Kumar 2017CSB1083
Aman Kumar 2017CSB1067
Kartikeya Pise 2017CSB1086

Idea : The idea is to make a polar to rectangular coordinates converter in Verilog which can take the values r (magnitude) and θ (angle) as the inputs using the switches and can display the corresponding x & y cordinate values on the seven-segment display.

Interface : The rightmost 2 LED's show the states and Leftmost shows overflow.
There are two modes of operation:

1) Floating point mode :

Input : The inputs are given in the order of r and θ with r being a half precision floating point number and θ being the binary equivalent of the angle
eg) $(11110)_2$ for angle 30°
The machine accepts the input when any of the 4 buttons excpt the central button (reset) is pressed.

Output : The output is displayed on the seven segment display in the floating point format (each digit is 4 bit hexadecimal value).
The calculated values are stored in registers and can be displayed whenever the user presses the buttons : btnR for Y value
btnL for X value

This mode of operation offers better precision and range.

2) **Fixed Point mode**

Input : Here r is given as a fixed point number, for the convenience of the user with first 8 bits as integer parts and next 8 bits for fraction part. +ve input assumed

The angle input is same for both the modes.

Output : The output is shown as the rounded off value of the integer parts of the X and Y coordinates

X is displayed on pressing btnU
Y is displayed on pressing btnD

The central button is the reset button.

LEDs : The LED's are used to show the state of the machine

No led on : Idle state
led[0] on : Ready to accept first input
led[1] on : Ready to accept second input
led[0] & led[1] on : Result state is achieved. Press buttons to view th different results

Modules :

① float_multi → It multiply two 16 bit floating point input one of (r) and another of (θ) and give output result in floating point. (θ: sinθ or cosθ)

② Normalize → it takes 22 bit input (Product of mantissa)
→ and gives 10 bit output mantissa
→ whether (1) or (0) is to be added to result exponent.

③ Lookup lut x → it takes input (90-θ) and gives its corresponding cosθ value in floating point.

④ Lookup Lut y → it takes input as (θ) in binary and gives its corresponding sinθ value in floating point notation.

⑤ Lookup2 LutfX → it takes input (90-θ) and gives its corresponding cosθ in fixed point notation. (first 8 bit denotes integer part and rest 8 bit denotes fraction part).

⑥ Lookup2 LutfY → it takes input θ and gives its corresponding sinθ in fixed point notation.

(7) fixed_multi → it takes two 16 bit input in the form of fixed point notation and return result as 16 bit output in fixed point notation.

(8) ssddecode → This module takes 4 bit input and as 4 bit can vary from (0 to 15), it returns 7 bit output denoting same input in its corresponding hexadecimal notation.

(9) ssd_cntr_ssd controller → This module takes 4 bit input and displays output on 7 segment display the result of hexdecimal we got from ssddecode,
and also controls which of the four will glow.

(10) bin2bcd → This module takes 9 bit input (binary) and convert it into 12 bit BCD, which can be shown on (7 segment display) as a (3 digit integer).

(11) Topmodule → This module is the main module which maintains the states of the machine and also calls all the other modules in the specific order to get the output displayed on the seven segment display.

State diagram

commonBTN = btnL | btnR | btnO | btnU

rst = btnC