Introduction

This implementation uses a DataPath and Controller to build a FSM. The design is a synthesizable Finite State Machine and Datapath which computes the greatest common denominator (GCD) of two 4-bit numbers. After the program is executed, the GCD_Output prints the common denominator and give the user a done signal to let the user know the GCD execution for those two numbers was accomplished. After every run, the 'Reset' button that is connected to the FPGA hard reset is used. The design also uses a debouncer to sync timing on button pushes.

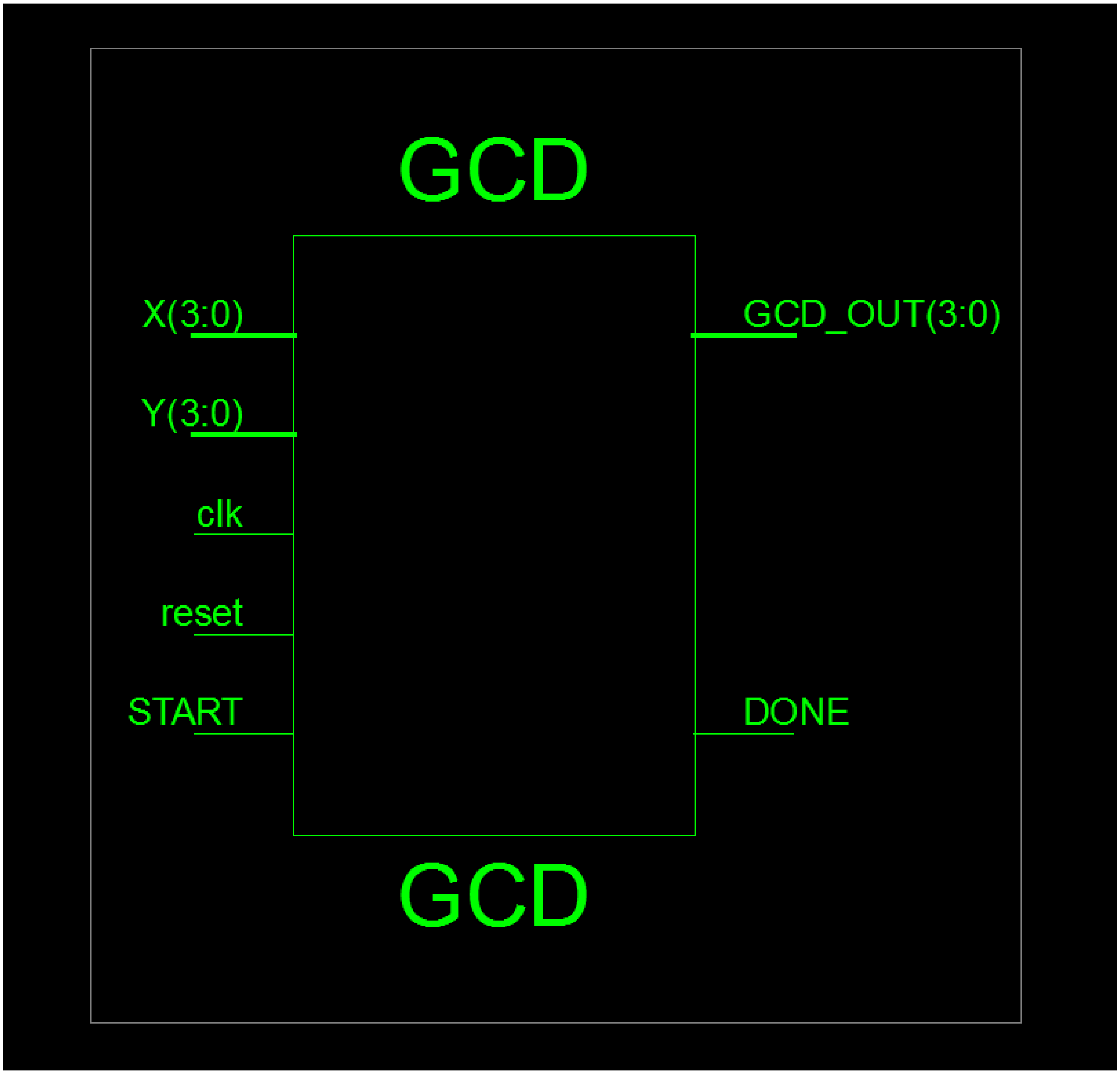
Design

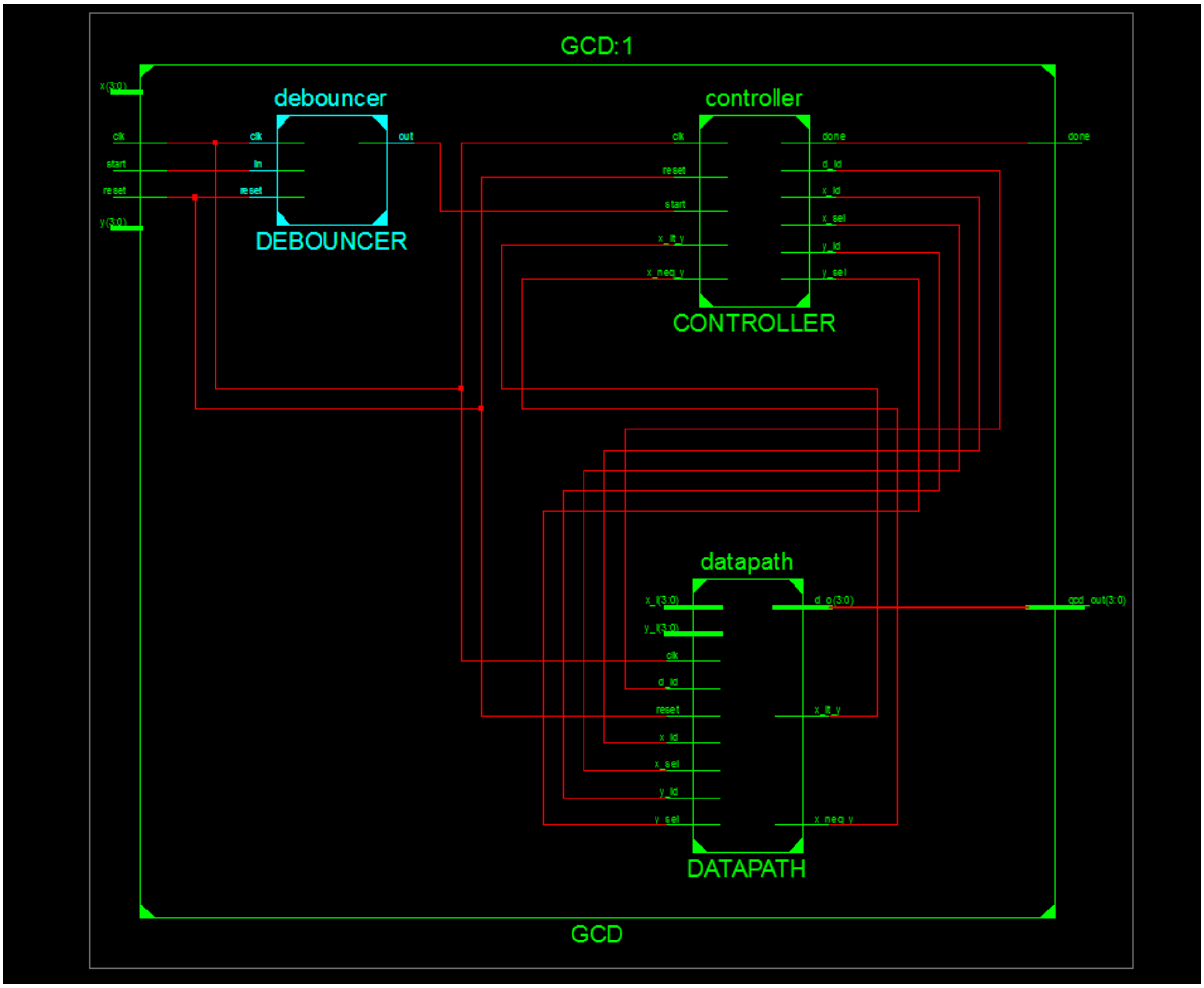
The logic design uses 3 register instances, 2 mux instances, 1 comparator instance, and 2 subtraction instances. The program waits for the user to press the 'Start' button then the process begins. The 'x_in' input goes into the register 'X', and 'y_in' input goes into the register 'Y'. X and Y are then compared, if the numbers are equal then the GCD_OUTPUT would simply be X, but if not, the program goes through the loop which compares X and Y and updates them by subtracting one from the other depending on which is the greater. This loop repeats until the two 4-bit numbers are equal to each other. Once the loop finishes, it then goes to the last state and sets 'DONE' signal to 1 so the user knows that the GCD computation is complete.

The design consists of 13 different states, 'state0', 'state1', 'state2', 'state3'… 'state12'. 'state0' initializes the signal 'DONE' and moves to 'state1', at 'state1' if 'Start = 1', it jumps to 'state2'. From 'state2' it goes to 'state5' where the "Select" and "Load" for X and Y are initialized and the comparison starts. From 'state5', it enters the loop where the program keeps looping and comparing X and Y. If X is greater than Y, X ← X – Y, but if Y is greater than X, Y ← Y – X. this loop continues until the two 4-bit numbers are equal to each other.

For the reset command, using the FPGA hard-reset the reset is an active low button (Reset == 0).

Here are some images of the schematic view of the FSM design:




GCD:1
x(3:0)
clk
start
reset
y(3:0)
debouncer
clk
in
reset
out
DEBOUNCER
controller
clk
reset
start
x_lt_y
x_neq_y
done
d_ld
x_ld
x_sel
y_ld
y_sel
CONTROLLER
done
datapath
x_i(3:0)
y_i(3:0)
clk
d_ld
reset
x_ld
x_sel
y_ld
y_sel
d_o(3:0)
x_lt_y
x_neq_y
DATAPATH
gcd_out(3:0)
GCD

Testing

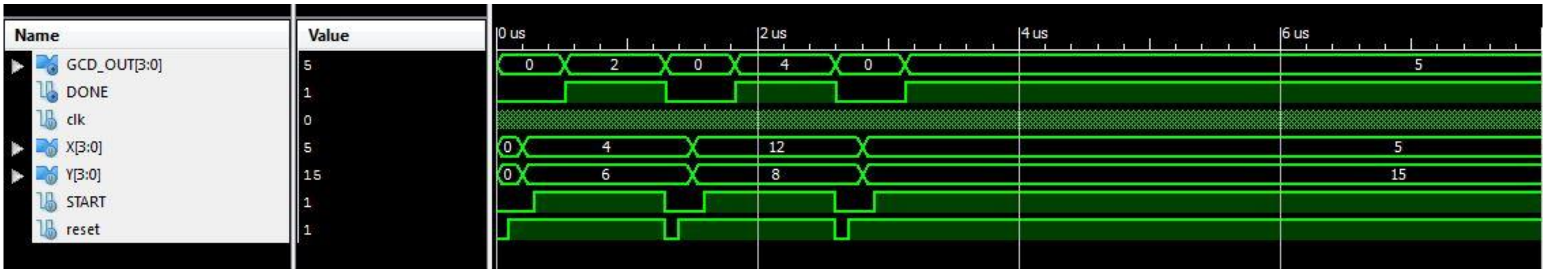
Along with manual testing, a simulation was used by running a testbench with the following test values; (X= 4, Y=8), then (X=12, Y=8), then (X=5, Y=15), then (X=7, Y=2), then (X=15, Y=12) and (X=15, Y=15). Each time new variables were entered and the GCD was computed the state machine needed to be reset to the initial state so the process can continue for new inputs.

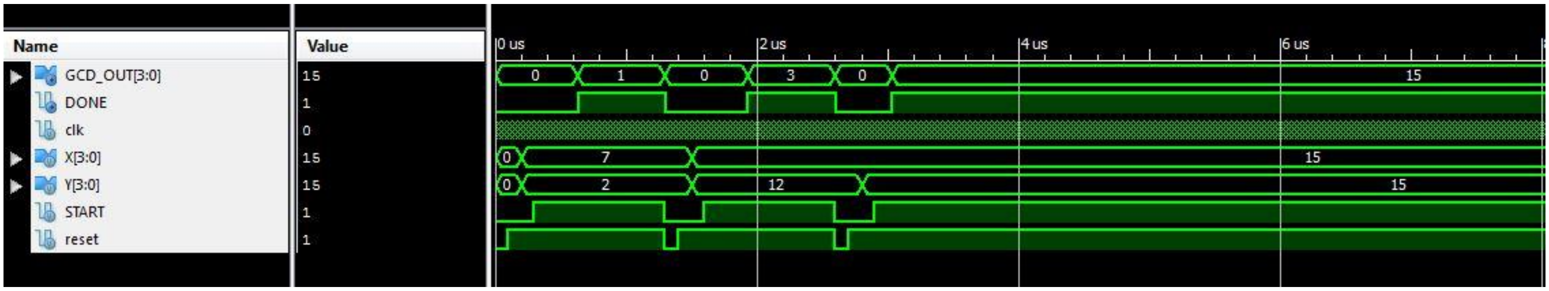
Results

As mentioned in the testing section, several possibilities that the GCD may encounter were tested (X is equal to Y, X is greater than Y, X is less than Y, and X or Y is significantly greater than the other).

The 'GCD_OUT' prints 0 every time a reset is performed between computations.

The waveforms from the testbench are included below;





Conclusion

Good exercise for experimenting with the datapath and controller link. Including the hard reset on the board was also a nice added challenge and a way to use more features of the board.