Gebze Technical University

CSE 331

Computer Organizations

HW3 Report

Yusuf Can Kan
161044007

1 Datapath

In order to implement datapath, first we need to implement 1,6 and 32 bir adders.
We will use 1 bit adder inside the 32 bit adder and and we will use 6 bit adder for counter.

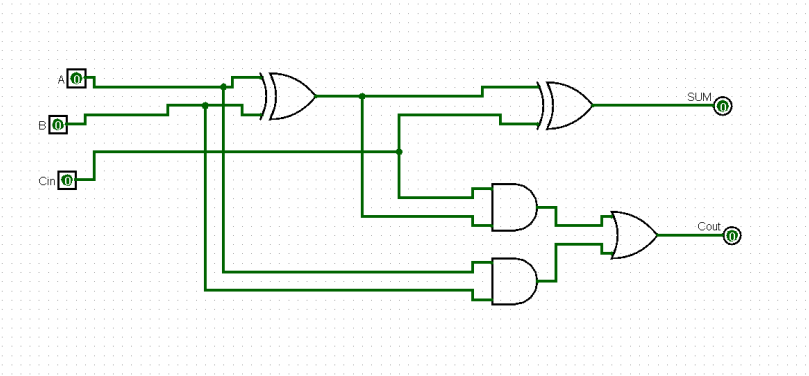

Figure 1: 1-bit adder

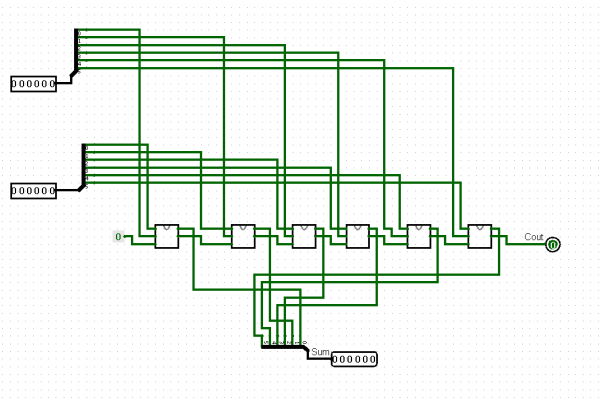

Figure 2: 6-bit adder

After this implementations we need to design shifter for out design. I used 32 bit splitter for the shifter and also I added 1 cout bit input for the case of any carry out comes from multiplication addition.(Ih cout part if

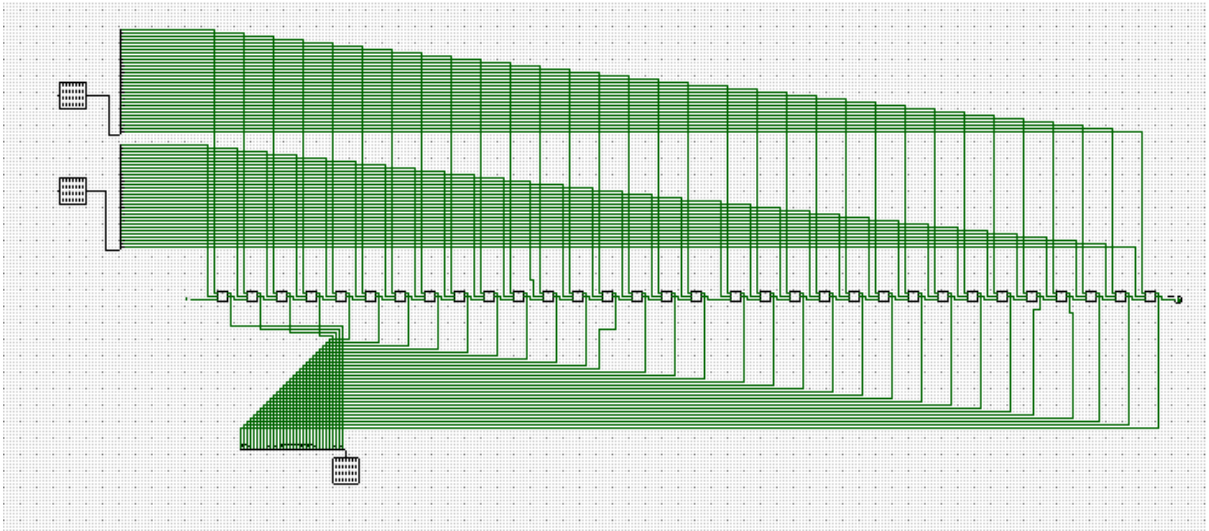

Figure 3: 32-bit adder (The whole design is in datapath.circ file)

multiplication process only calculates shifting' cout will always be 0, but in case of addition and shifting, this cout bit will be cout of the adder.) Also shifter takes 2 32 bit value for calculationg 64 bit shift operation. I t gives 2 32 bit value as a result.

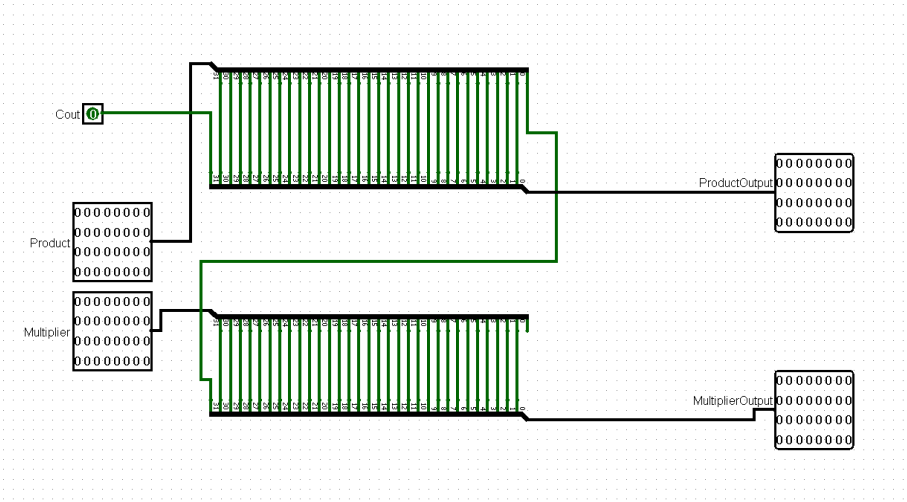

Figure 4: shifter

In addition we need an comparator(just for equal part) for out counter design.

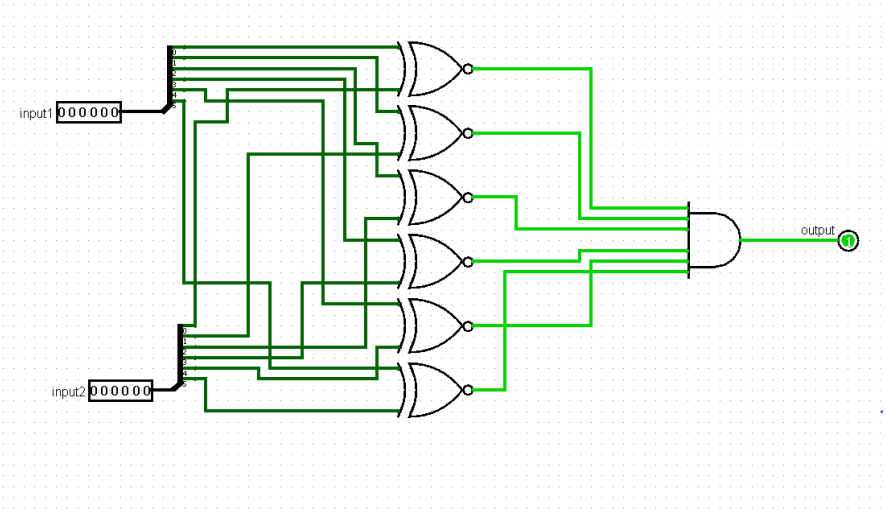

Figure 5: isEqual

The general datapath;

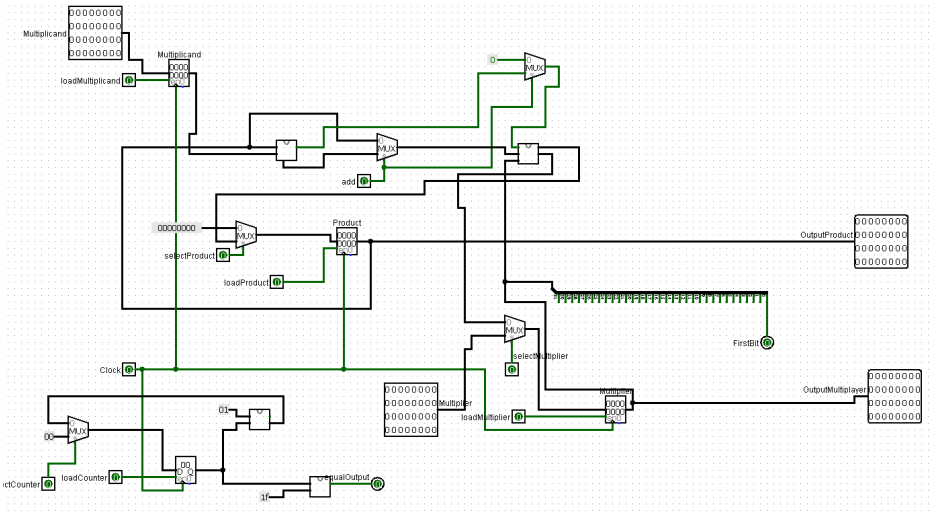

Figure 6: datapath

The datapat supports 32 bit inputs for multiplier and multiplicand. It will gives 64 bit output with using 2 32 bit register.

Datapath includes counter for counting how many times shifting operation is made. If process made 32 times, It finish the program.

2 Control Unit

2.1 State Diagram

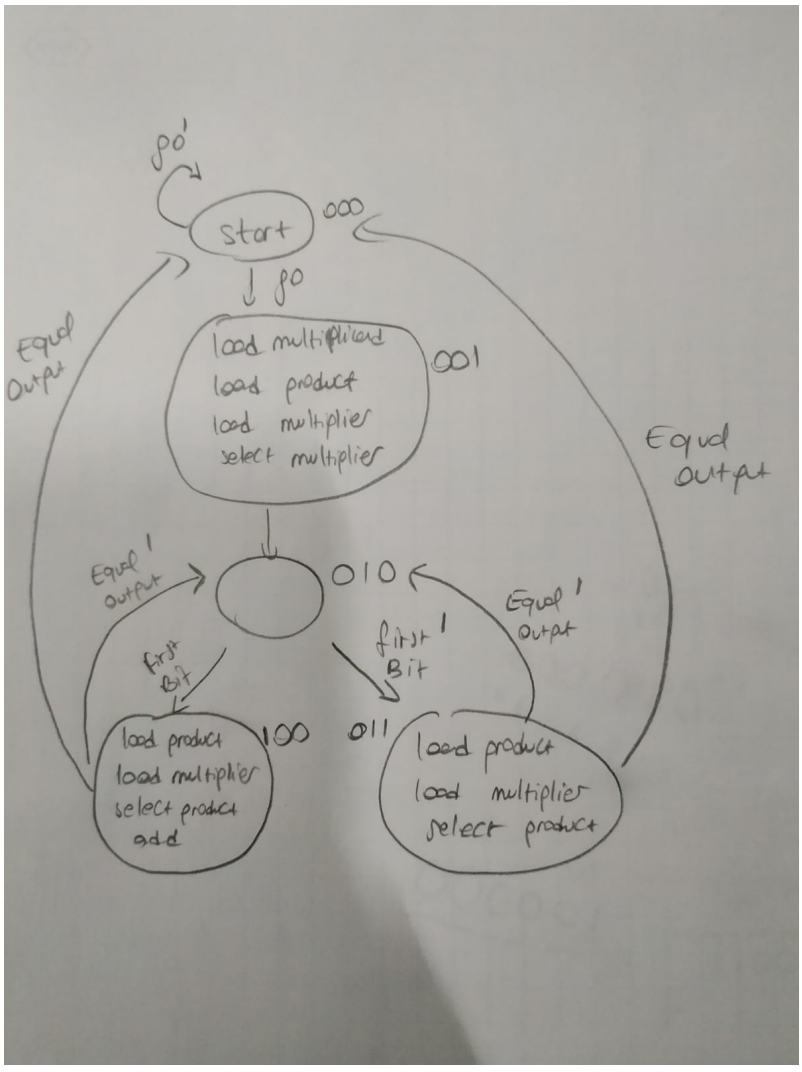

Figure 7: State Diagram

2.2 Tables

	load multiplicand	load product	load multiplier	select multiplier	select product	add	selectCounter	loadCounter
start (000)								
load (001)	1	1	1	1	0	x	1	1
add + shift (100)	0	1	1	0	1	1	0	1
shift (011)	0	1	1	0	1	0	0	1

	INPUT			NEXT STATES		
S2 S1 S0	go	equal Output	firstBit	N2	N1	N0
0 0 0	0	X	X	0	0	0
0 0 0	1	X	X	0	0	1
0 0 1	X	X	X	0	1	0
0 1 0	X	X	1	1	0	0
0 1 0	X	X	0	0	1	1
1 0 0	X	0	X	0	1	0
1 0 0	X	1	X	0	0	0
0 1 1	X	0	X	0	1	0
0 1 1	X	1	X	0	0	0

Figure 8: Boolean Table

N2=S2'S1S0'firstBit
N1=S2'S1'S0 + S2'S1S0'fistBit' + S2S1'S0'equalOutput' + S2'S1S0equalOutput'
N0=S2'S1'S0'go+S2'S1S0'firstbit'

3 Test Cases

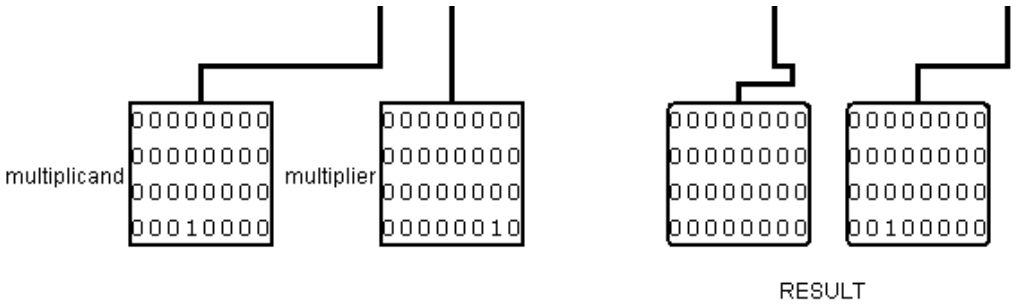

Figure 9: Test 1

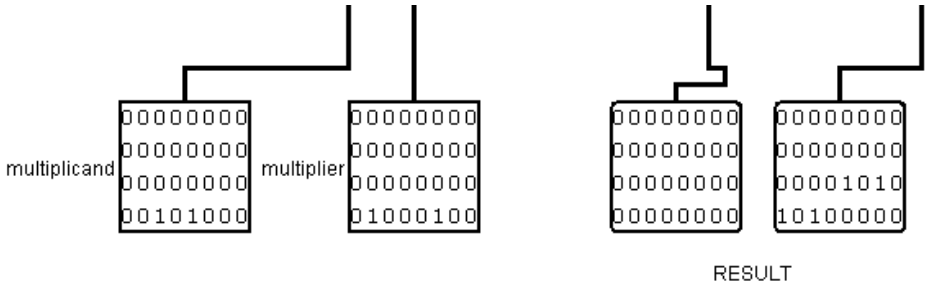

Figure 10: Test 2

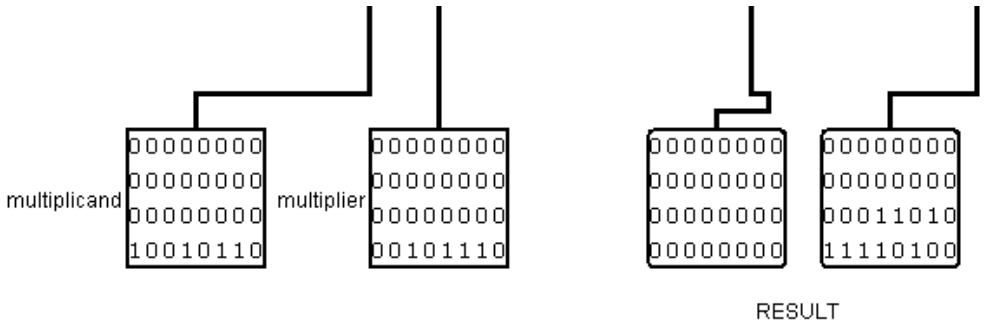

Figure 11: Test 3

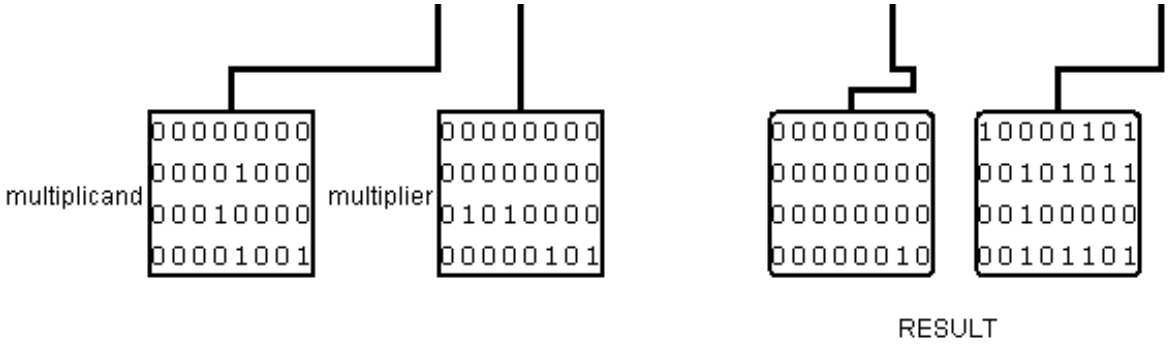

Figure 12: Test 4