CSE 232 Spring 2020

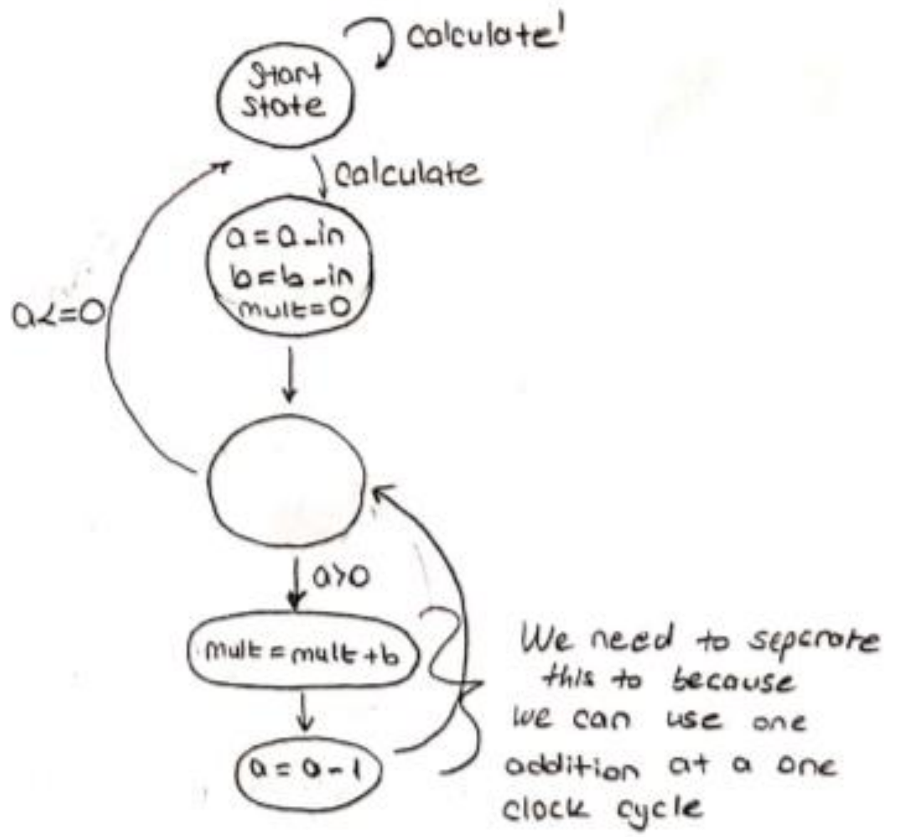
Project 2

171044032
Tanker Tercan

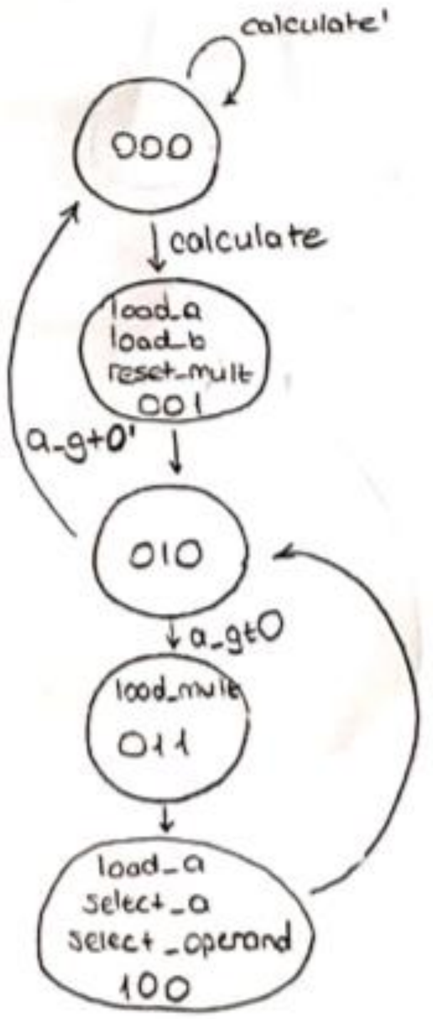
C Code :

```
1- while (!calculate);
2- a=a_in;
3- b=b_in;
4- mult =0;
5- while (a>0) {
6-     mult = mult +b;
7-     a=a-1;
8- }
```

Early FSM :


calculate'
Start State
calculate
a=a_in
b=b_in
mult=0
a<=0
a>0
mult=mult+b
a=a-1
We need to seperate this to because we can use one addition at a one clock cycle

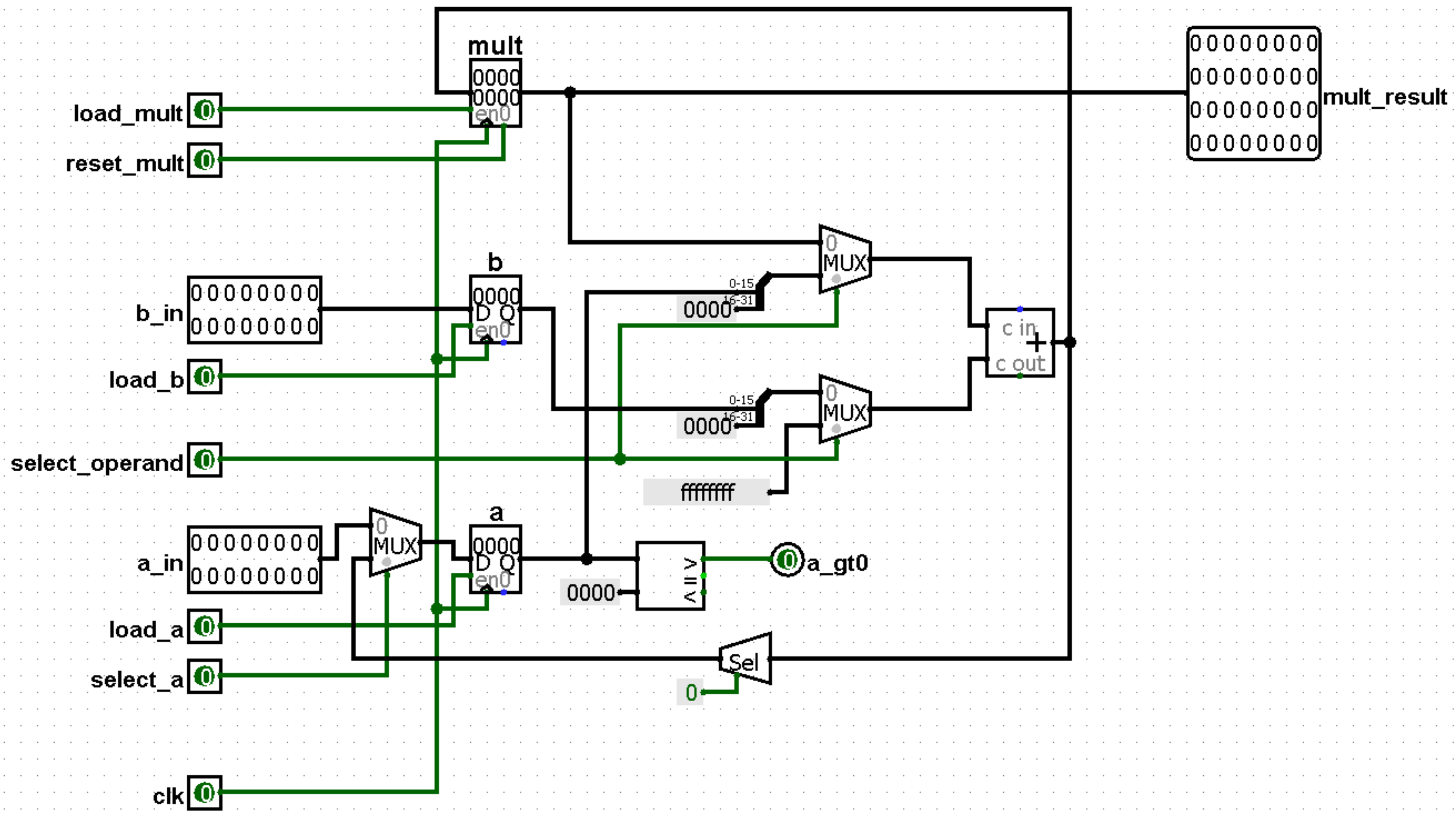
State Diagram :



Datapath :

- We need 3 registers for a, b and mult.
- We can only use one adder unit
- Use three muxes, one for loading a, rest of them will be used for adder
- And one comparator for is a greater than 0

Draw Datapath:



Boolean Expression :

Present State			Inputs		Next State		
P2	P1	P0	a_gt0	calculate	N2	N1	N0
0	0	0	—	0	0	0	0
0	0	0	—	1	0	0	1
0	0	1	—	—	0	1	0
0	1	0	0	—	0	0	0
0	1	0	1	—	0	1	1
0	1	1	—	—	1	0	0
1	0	0	—	—	0	1	0

$N2 = P2'P1P0$

$N1 = P2'P1'P0 + P2'P1P0'\,a\_gt0 + P2P1'P0'$

$N0 = P2'P1'P0'\,calculate + P2'P1P0'\,a\_gt0$

$= P2'P0'(P1'\,calculate + P1\,a\_gt0)$

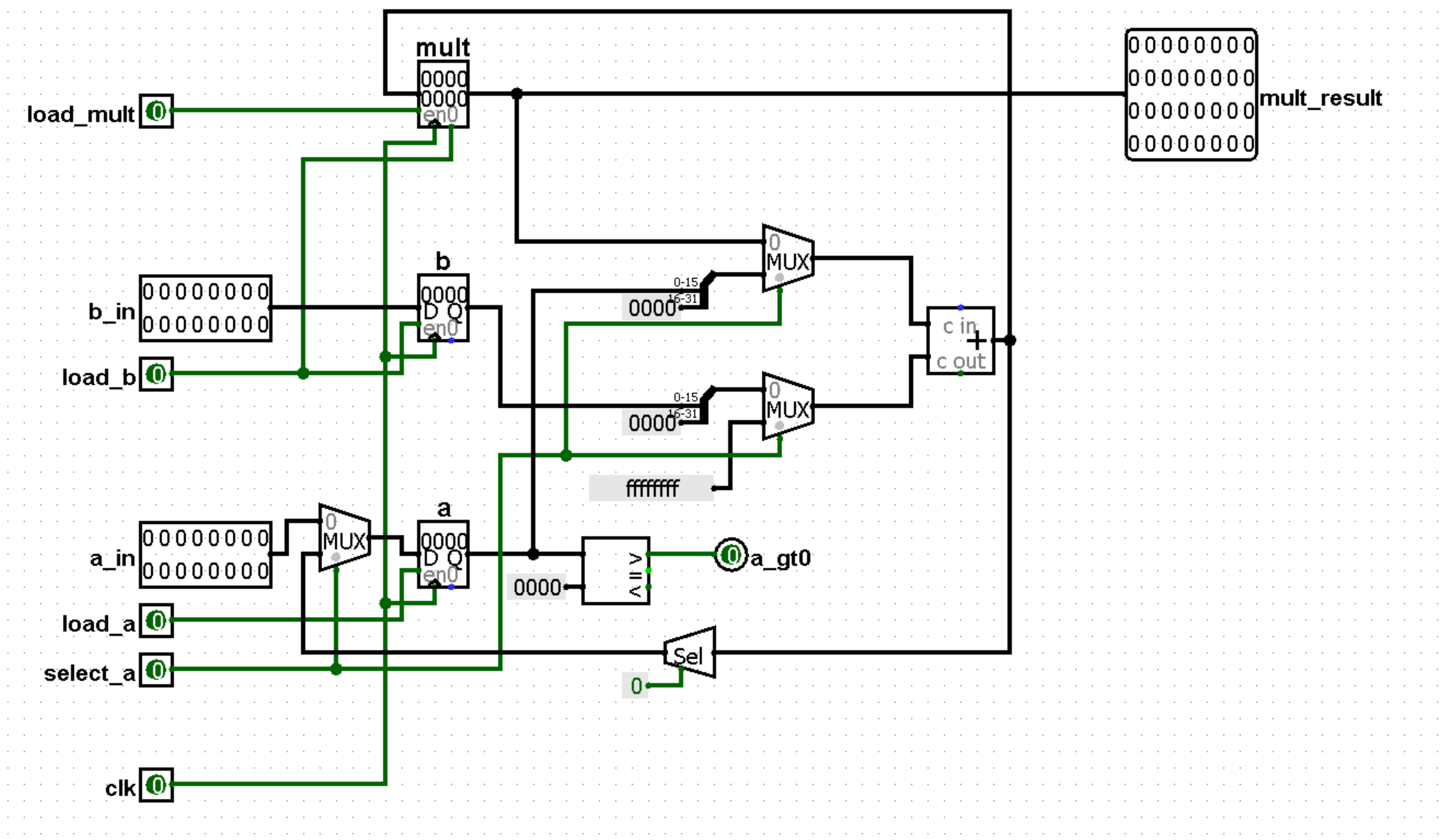
Present State				Outputs					
	P2	P1	P0	load_a	load_b	reset_mult	load_mult	select_a	select_operand
S0	0	0	0	0	0	0	0	0	0
S1	0	0	1	1	1	1	0	0	0
S2	0	1	0	0	0	0	0	0	0
S3	0	1	1	0	0	0	1	0	0
S4	1	0	0	1	0	0	0	1	1

load_a = S1 + S4   reset_mult = S1   select_a = S4

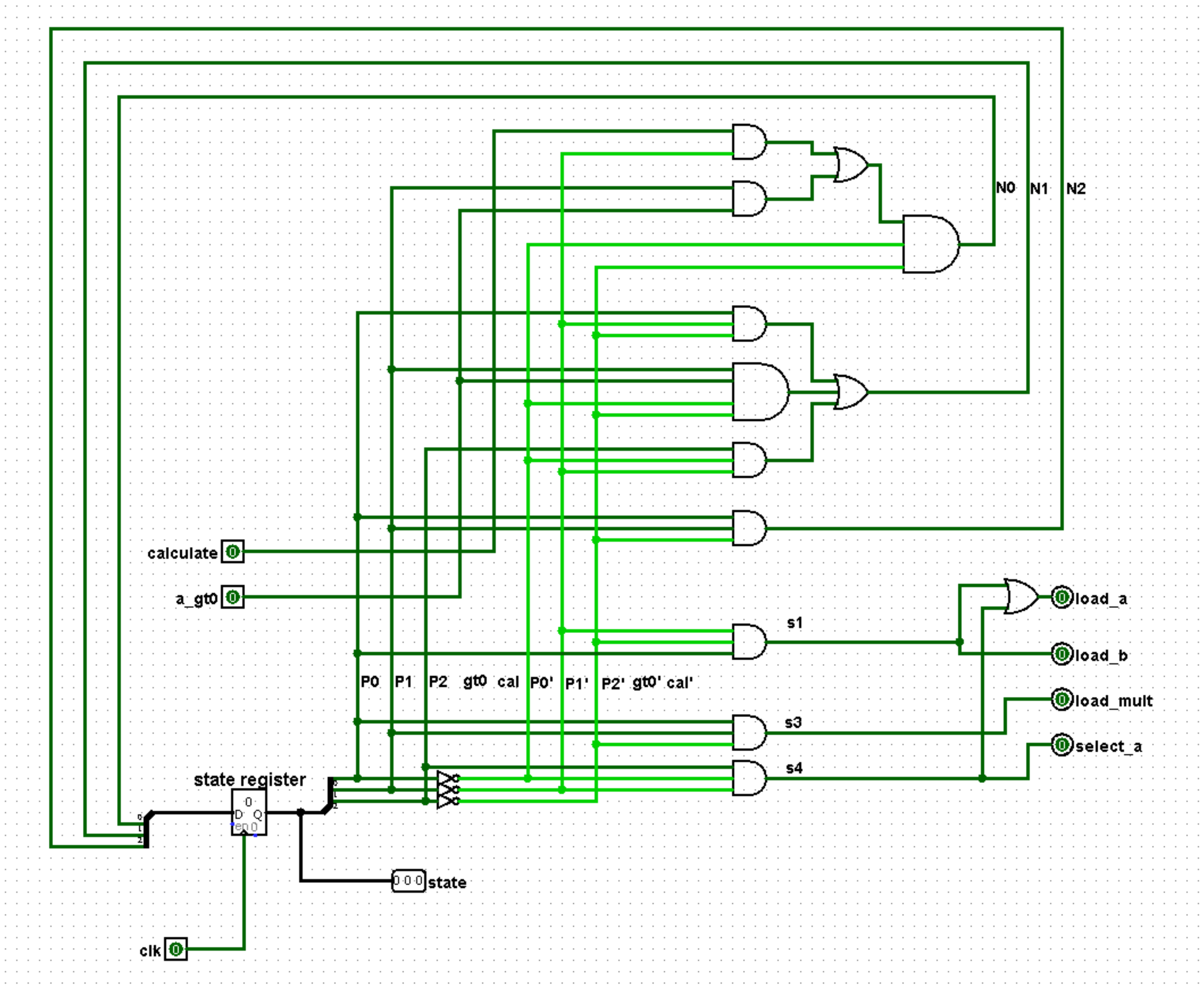
load_b = S1   load_mult = S3   select_operand = S4

* After i designed datapath, i realized there is no need to use reset_mult and select_operand because load_b = reset_mult and select_a = select_operand , so i decided to not to use them.

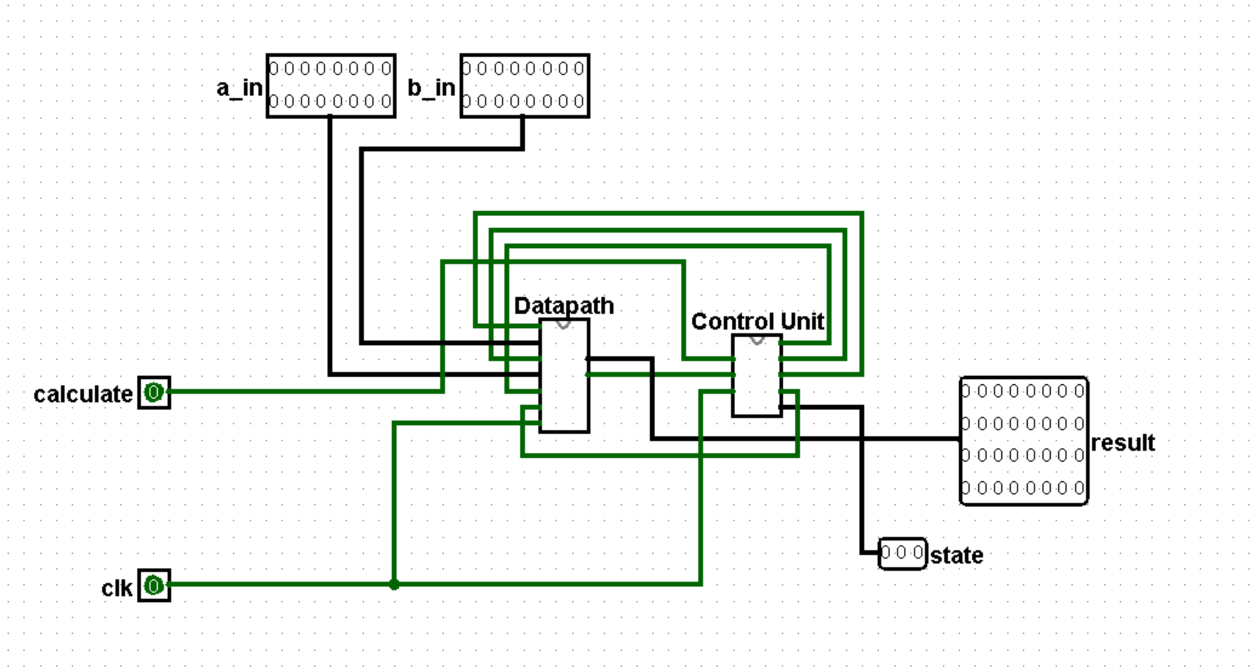
Optimized Datapath:



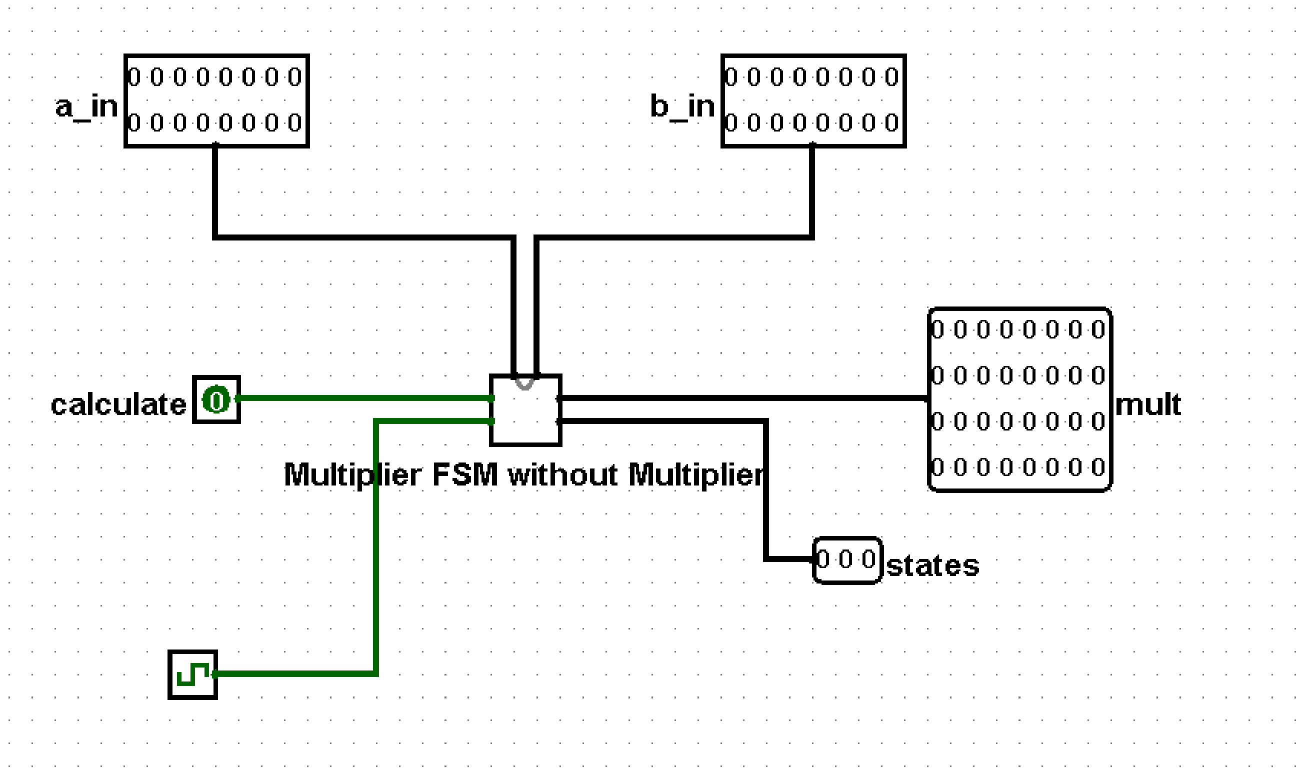
Control Unit:



FSM:


a_in
b_in
Datapath
Control Unit
calculate
clk
result
state

Main:


a_in
b_in
calculate
mult
Multiplier FSM without Multiplier
states

Simulation:

- This FSM does not work non positive integers.

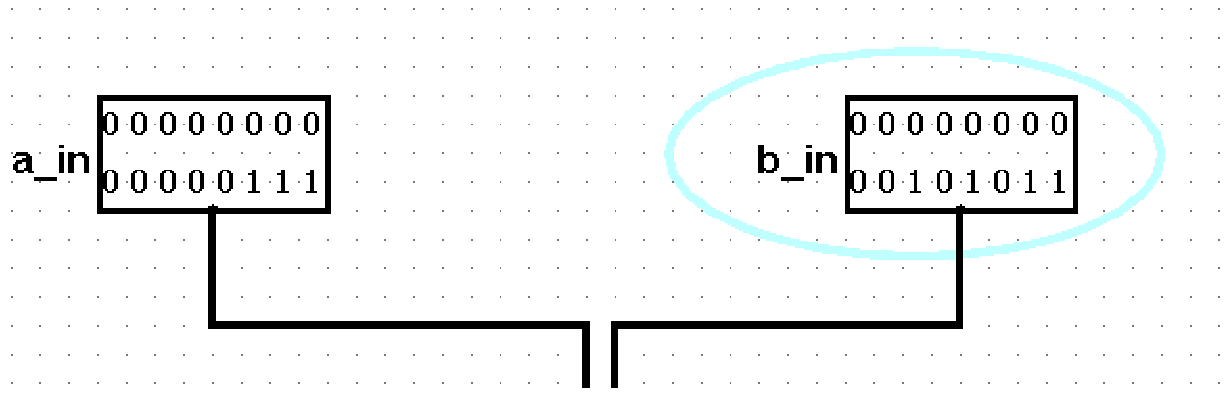
Test Case: Work on positive integers

Test Data: a_in = 7, b_in = 43, mult =?

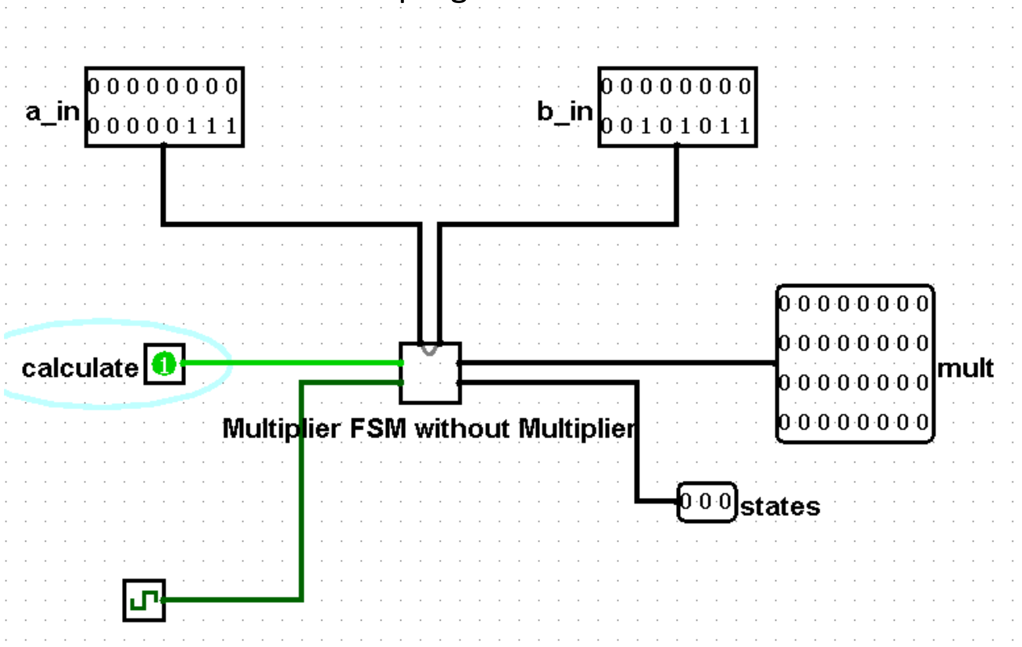
Excepted Result: mult = 301

Simulation:

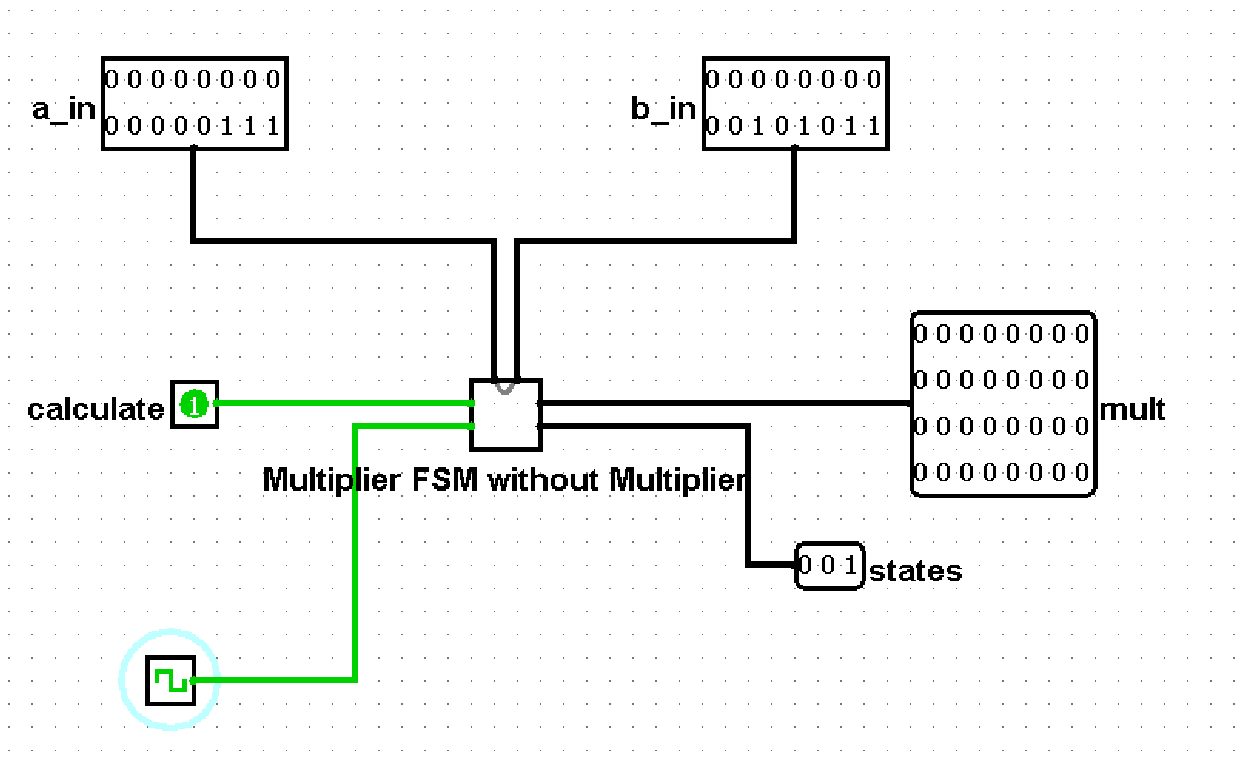
- Set up the inputs



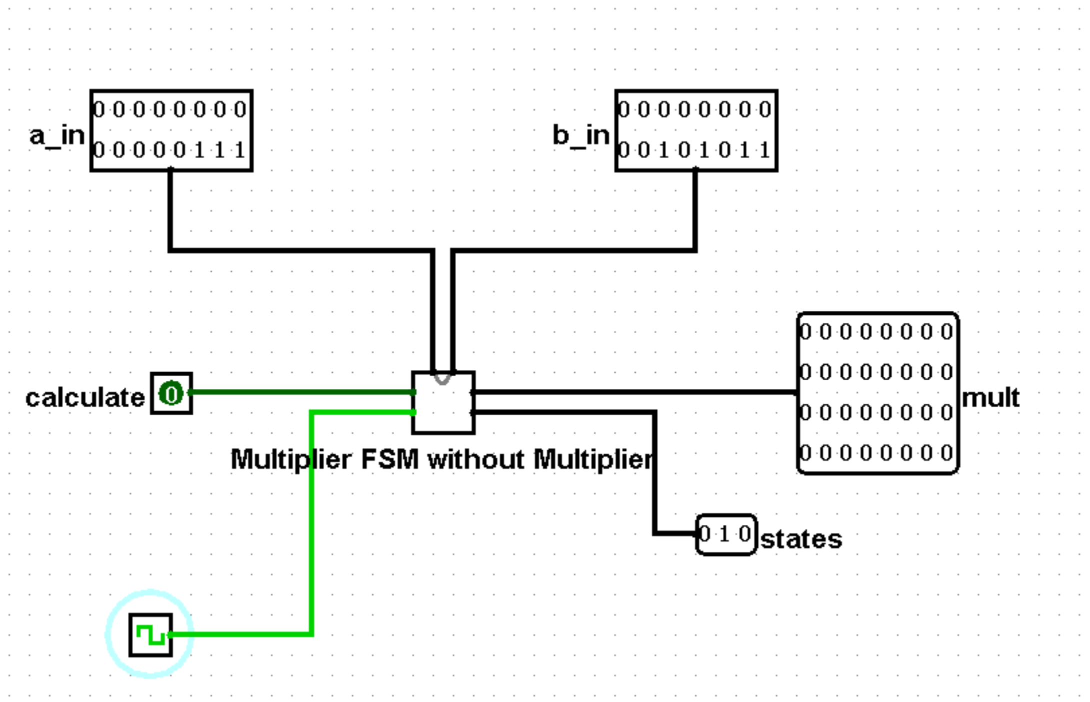
- Press calculate and program will do the rest



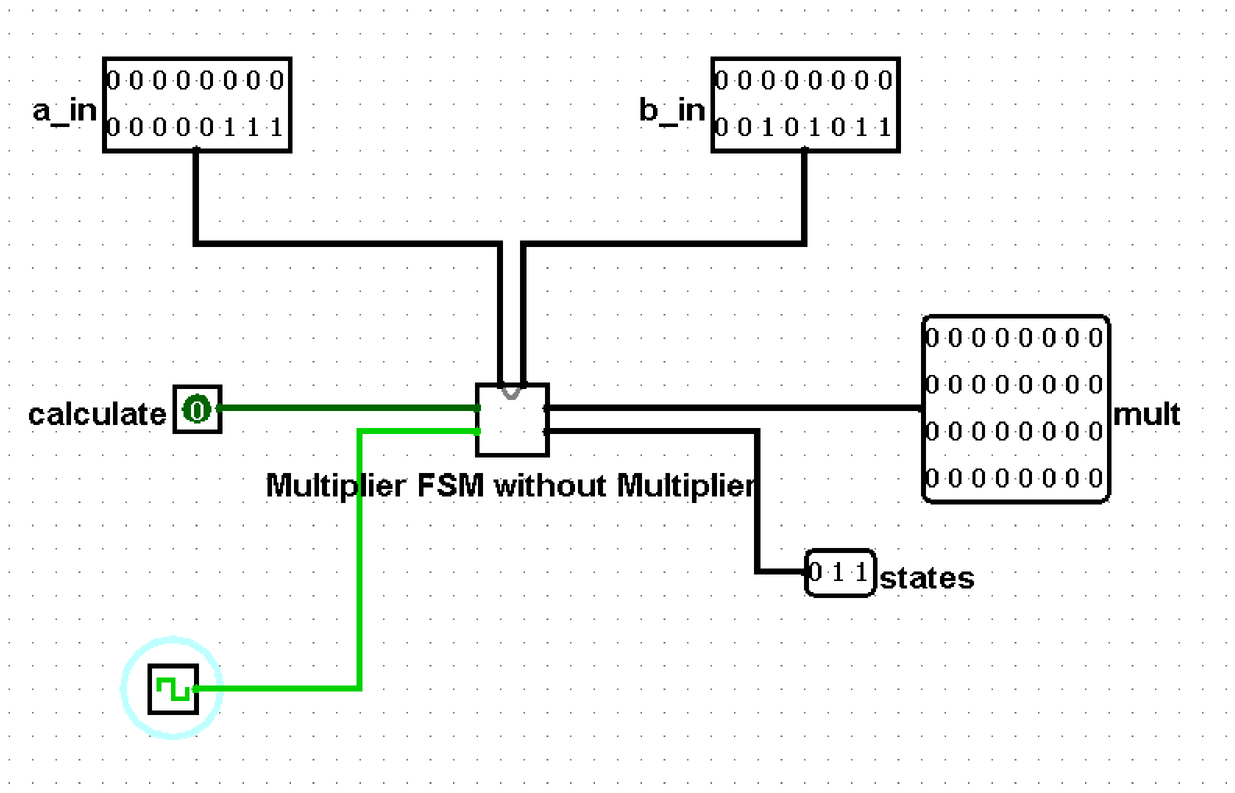
- Enters first state



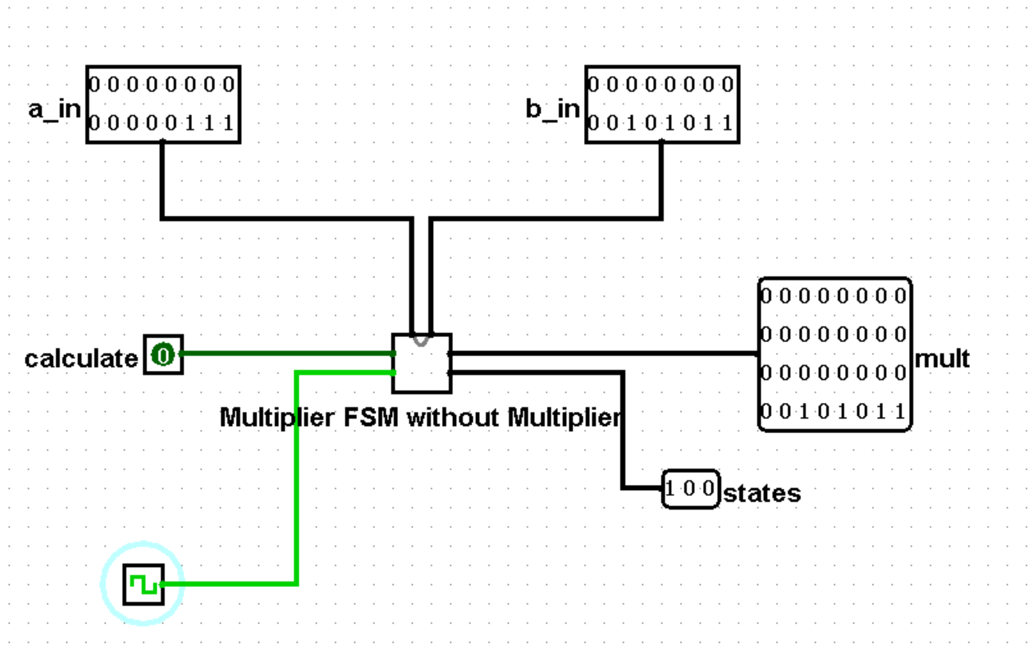
- Second state



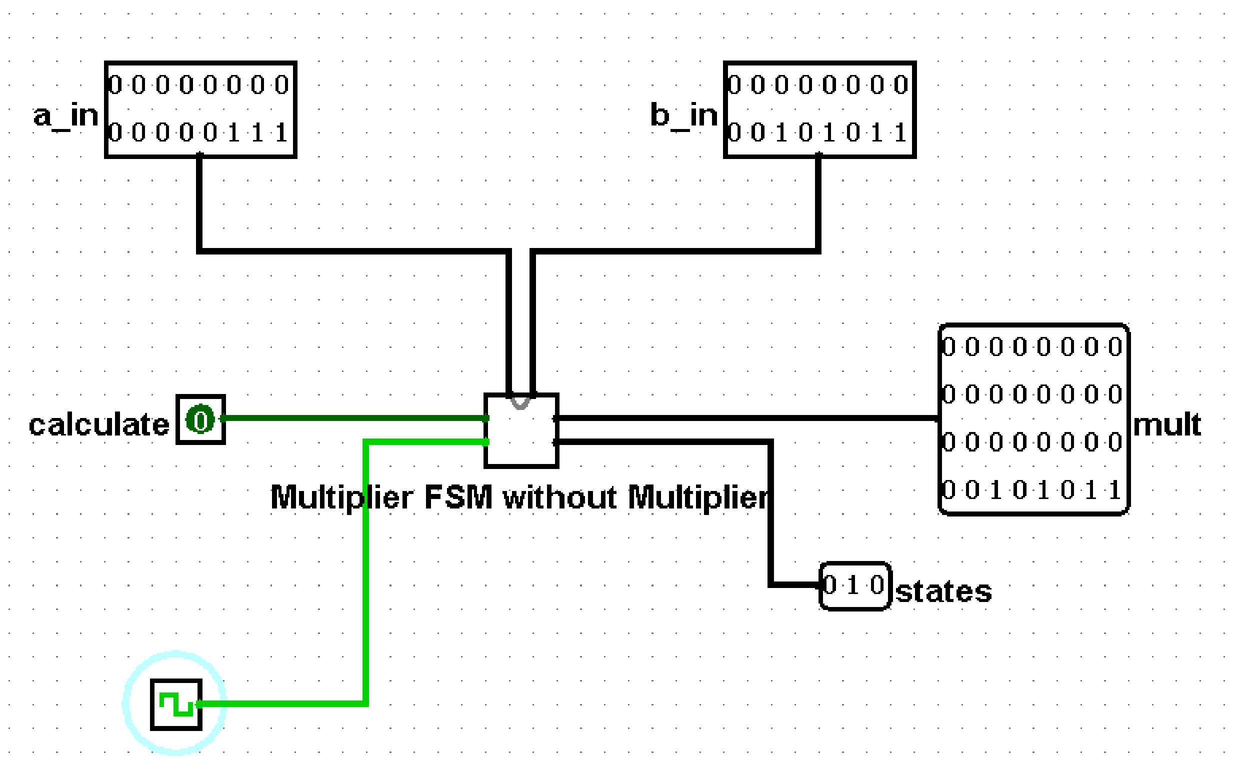
- Third state

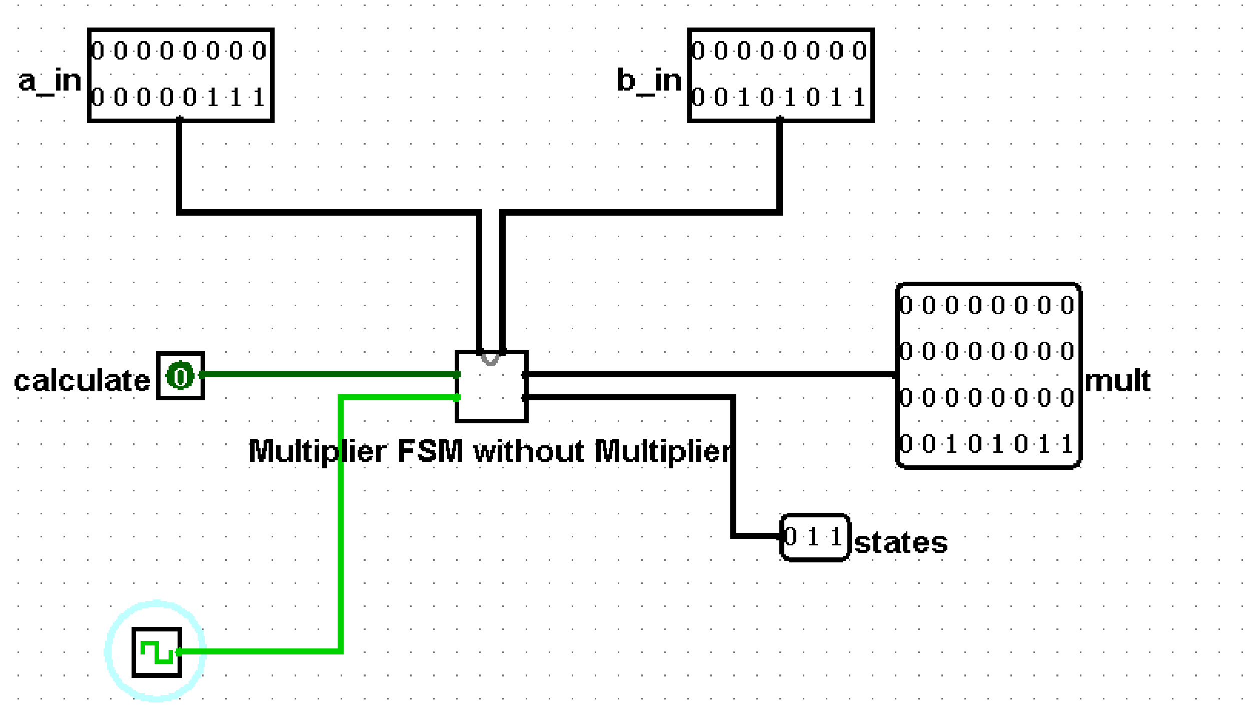


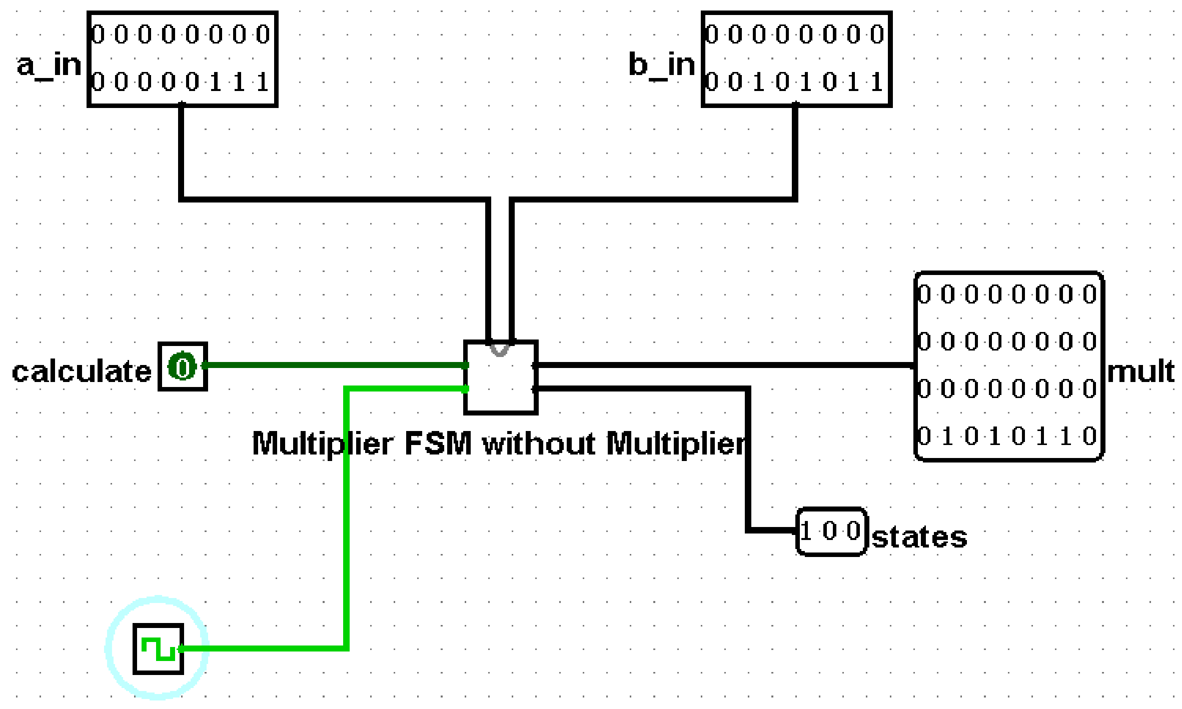
- Fourth state

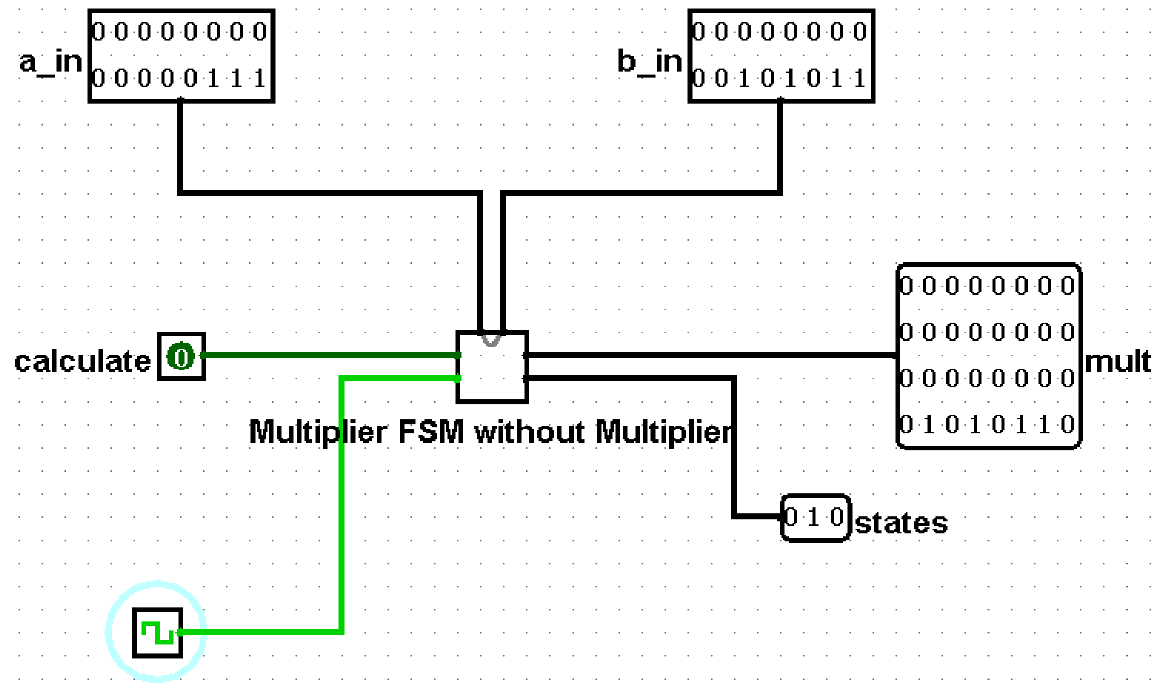


- As you can see, addition added to mult and after this state, it should return to state 2 and it will continue to loop until a_register equals to zero.

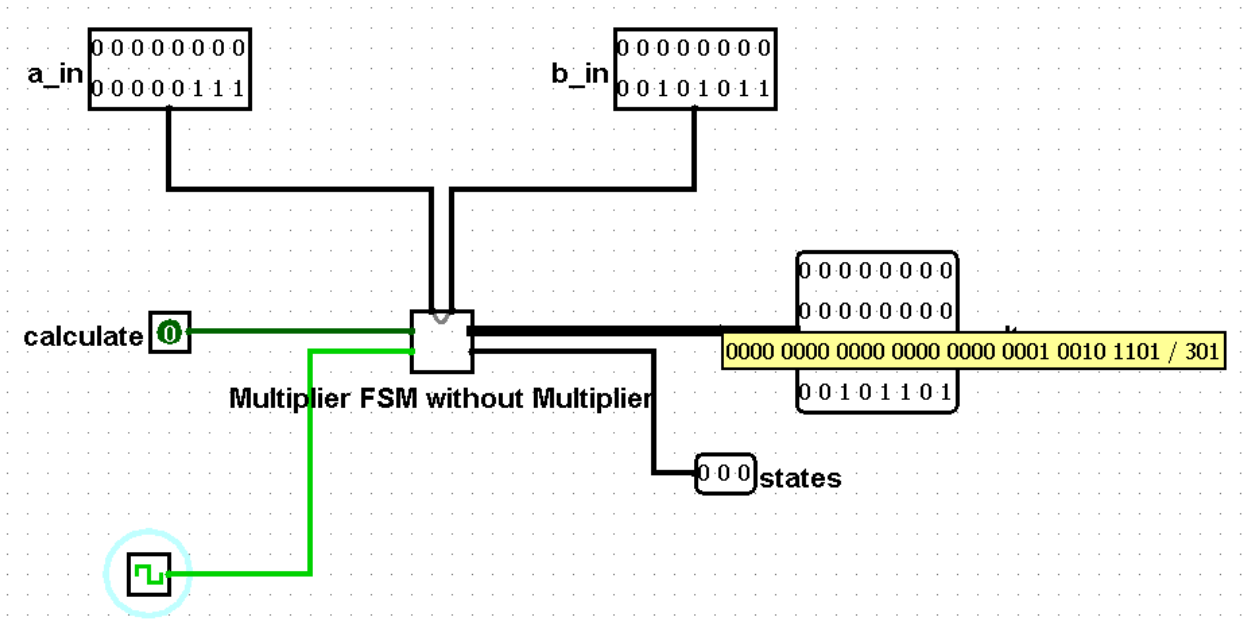





a_in
0 0 0 0 0 0 0 0
0 0 0 0 0 1 1 1
b_in
0 0 0 0 0 0 0 0
0 0 1 0 1 0 1 1
calculate
0
Multiplier FSM without Multiplier
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 1 0 1 0 1 1 0
mult
1 0 0
states


a_in
0 0 0 0 0 0 0 0
0 0 0 0 0 1 1 1
b_in
0 0 0 0 0 0 0 0
0 0 1 0 1 0 1 1
calculate
0
Multiplier FSM without Multiplier
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 1 0 1 0 1 1 0
mult
0 1 0
states

- When a_register becomes zero, multiplication will end, and program will go to the start state



0000 0000 0000 0000 0000 0001 0010 1101 / 301

- Mult equals to 301. Our simulation successfully ended.