# CSE 331 Computer Organizations Homework 3

Fatih Kaan Salgır - 171044009

## 1 Datapath

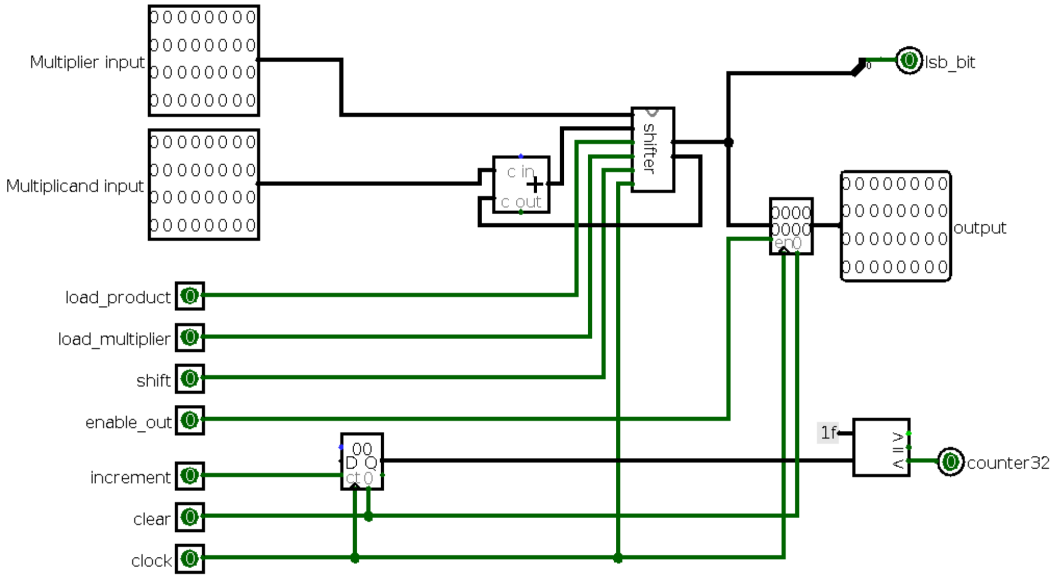
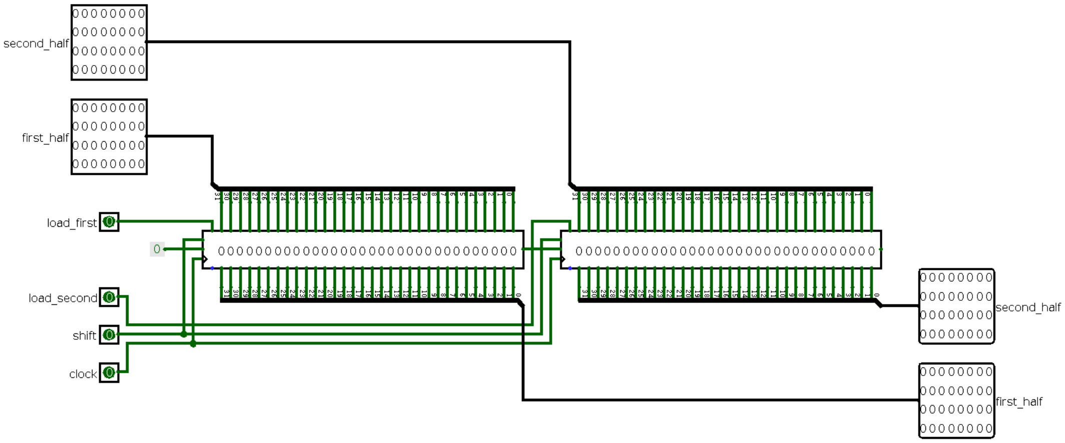


Figure 1: Datapth

- To avoid glitches on the output, I put a register before the output and enabled on the last state. (`enable_out`)
- Counter and comparator are 6 bits (They could be 5 bits, but I prefered this way to simplify state diagram).
- `clear` will clear both counter and contents of the output register after.
- `increment` will increment the counter, so that we can check if we reach 32. In this case machine will go to last state.

## 1.1 Shfiter



- I have used 2 32-bit shift register by connecting first shifter's output bit to second shifter's input bit.
- Since our fsm will be responsible calculating unsigned numbers, first shifter's input bit should be 0.
- `load_first` and `load_second` signals will load the values to registers.
- `shift` signal will shift both registers to right.

## 2 State Diagram

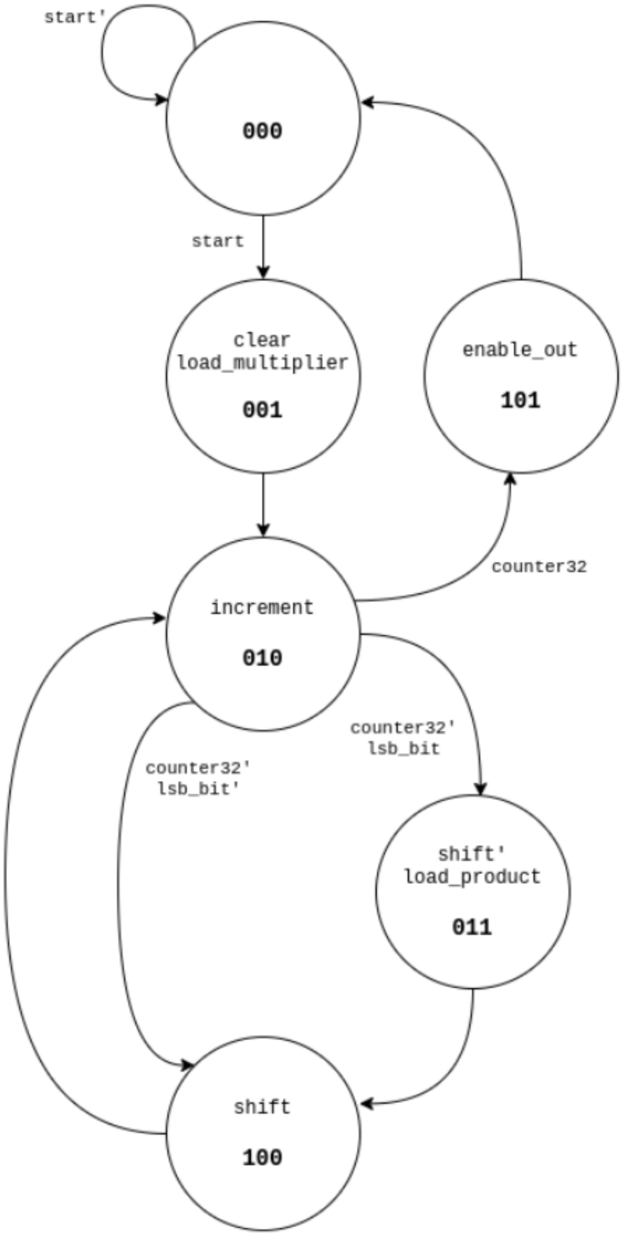


Figure 2: FSM

# 3 Designing Control Unit

## 3.1 Truth Table

Table 1: Next state according to present state and inputs

| s2 | s1 | s0 | start | counter32 | lsb_bit | 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 | 1 | X | 1 | 0 | 1 |
| 0 | 1 | 0 | X | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | X | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | X | X | X | 1 | 0 | 0 |
| 1 | 0 | 0 | X | X | X | 0 | 1 | 0 |
| 1 | 0 | 1 | X | X | X | 0 | 0 | 0 |

```
n2 = s2’ s1 s0’ counter32 + s2’ s1 s0’ counter32’ lsb_bit’ + s2’ s1 s0
n2 = s2’ s1 s0’ ( counter32 + counter32’ lsb_bit’ ) + s2’ s1 s0
n2 = s2’ s1 s0’ ( counter32 + lsb_bit’ ) + s2’ s1 s0
n2 = s2’ s1 ( s0’ ( counter32 + lsb_bit’ ) + s0 )
n2 = s2’ s1 ( counter32 + lsb_bit’ + s0 )

n1 = s2’ s1’ s0 + s2’ s1 s0’ counter32’ lsb_bit + s2 s1’ s0’
n1 = s1’ ( s2’ s0 + s2 s0’) + s2’ s1 s0’ counter32’ lsb_bit
n1 = s1’ ( s2 XOR s0) + s2’ s1 s0’ counter32’ lsb_bit

n0 = s2’ s1’ s0’ start + s2’ s1 s0’ counter32 + s2’ s1 s0’ counter32’ lsb_bit
n0 = s2’ s1’ s0’ start + s2’ s1 s0’ ( counter32 + counter32’ lsb_bit )
n0 = s2’ s1’ s0’ start + s2’ s1 s0’ ( counter32 + lsb_bit )
```

Table 2: Inputs according to states

| s2 | s1 | s0 | load_product | load_multiplier | shift | enable_out | increment | clear |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |

```
load_product = s2’ s1 s0
load_multiplier = s2’ s1’ s0
shift = s2 s1’ s0’
enable_out = s2 s1’ s0
increment = s2’ s1 s0’
clear = s2’ s1’ s0
```

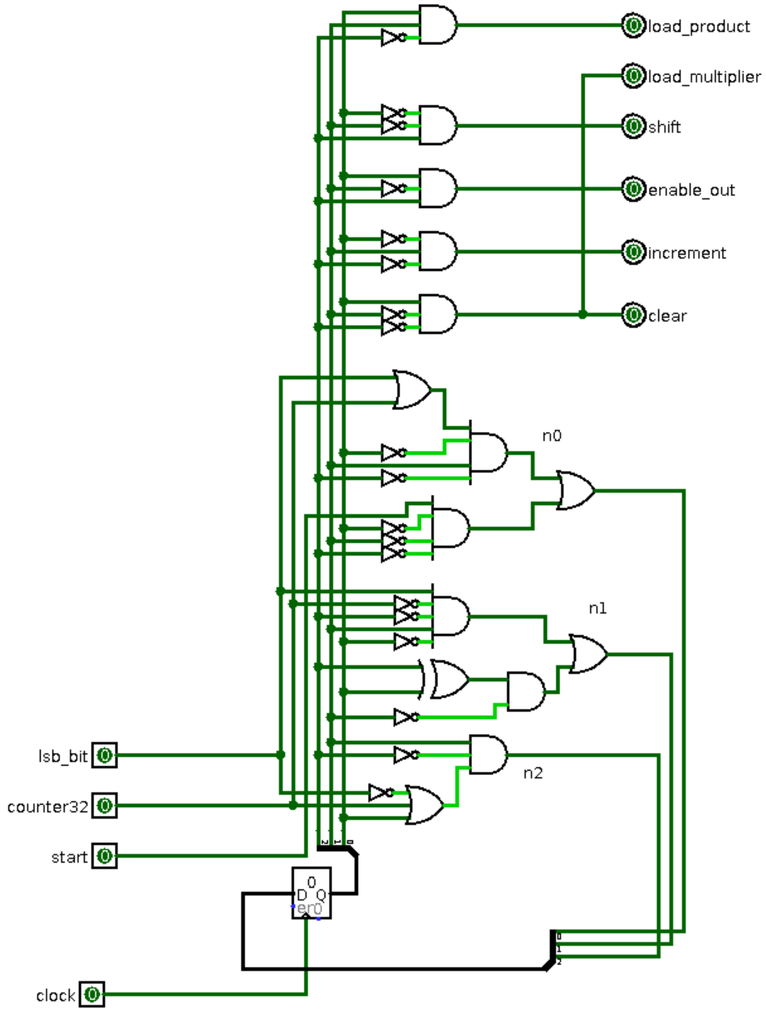


Figure 3: Control Unit

# 4 Connecting Datapath and Control Unit & Tests

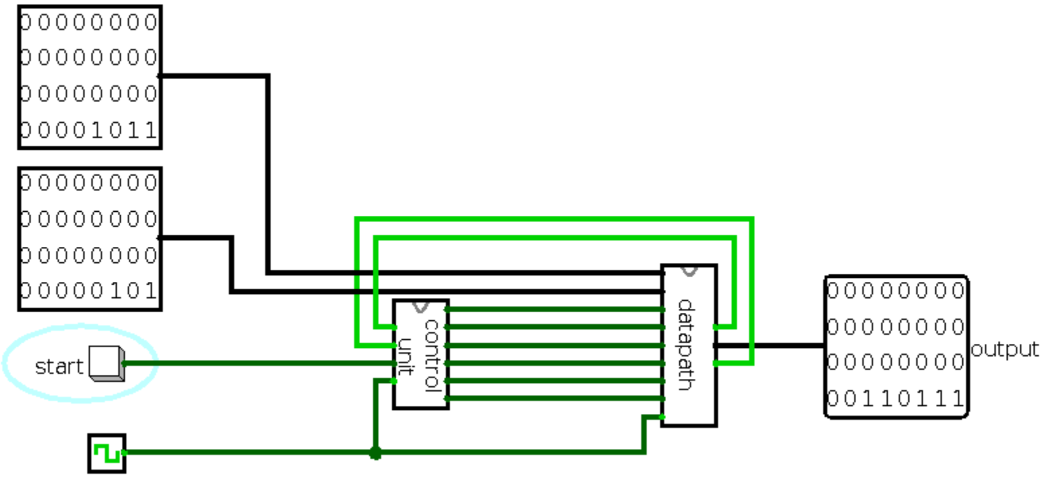


Figure 4: 11 × 5 = 55

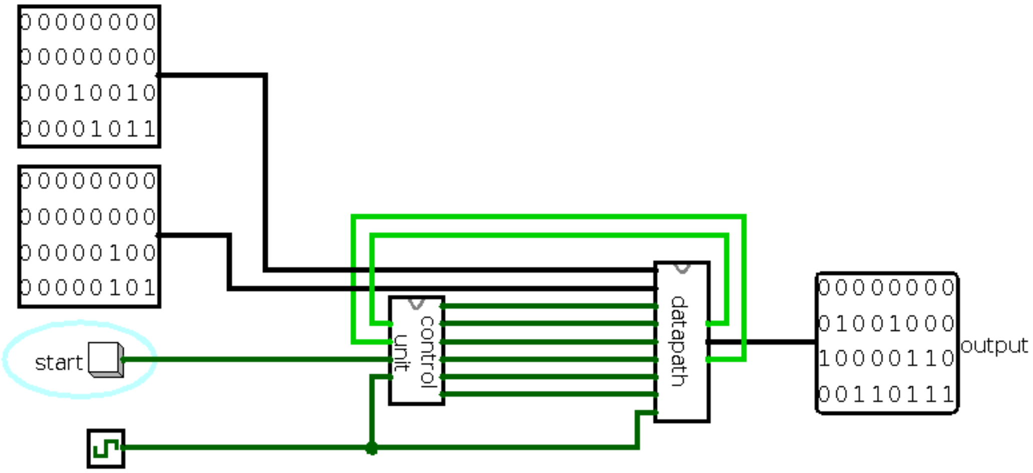


Figure 5: 4619 × 1029 = 4752951

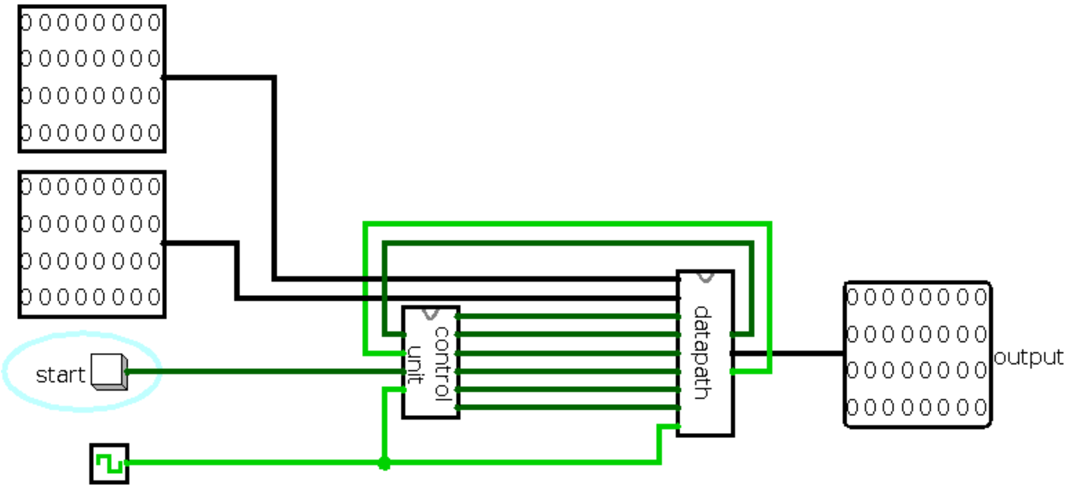


Figure 6: $0 \times 0 = 0$

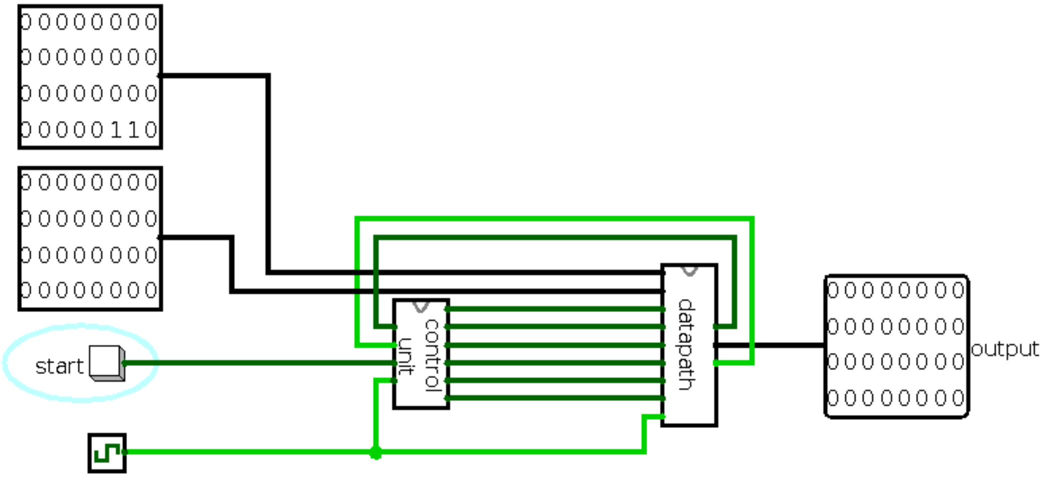


Figure 7: $6 \times 0 = 0$

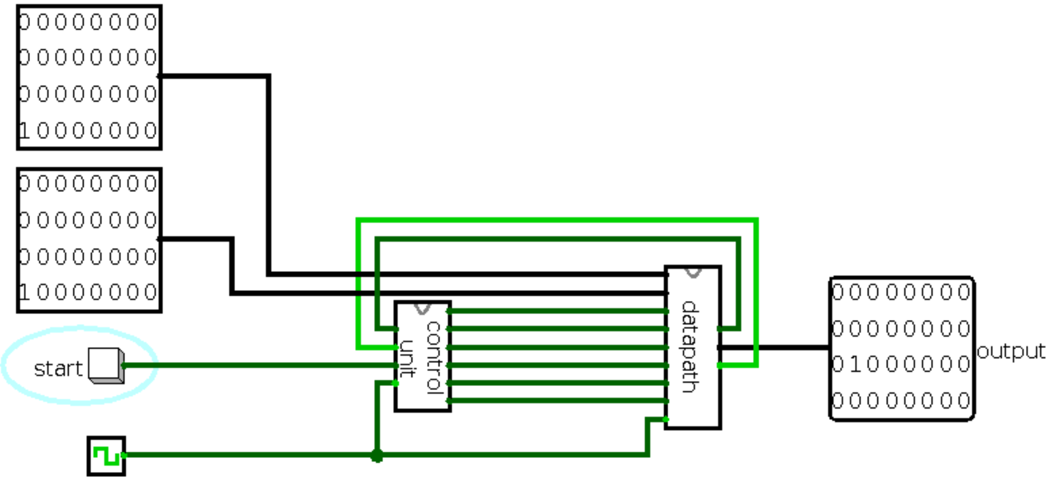


Figure 8: $128 \times 128 = 16384$

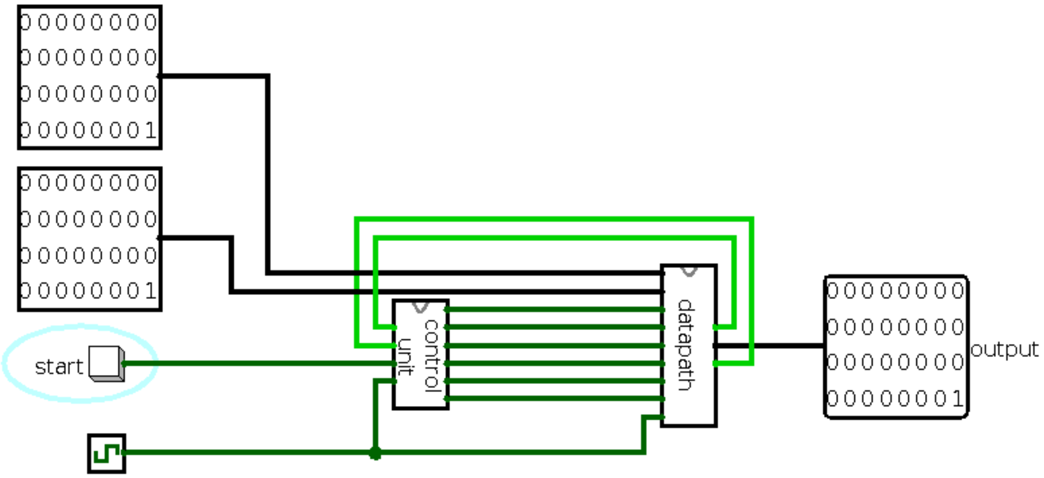


Figure 9: $1 \times 1 = 1$