Multiplier Implementation

Unsigned Multiplication

Iterative Multiplier

- Sequential algorithm:

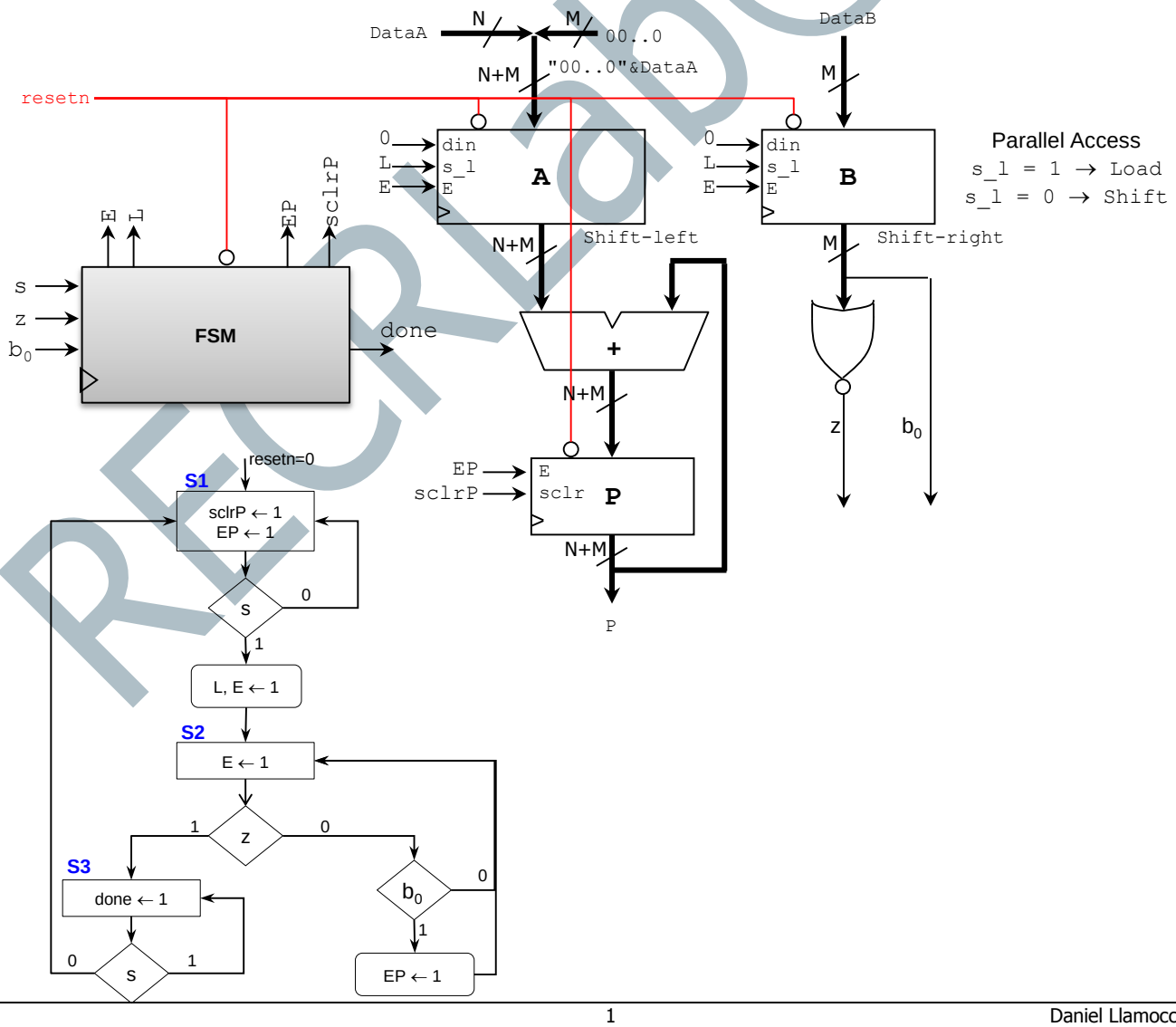
```
P ← 0, Load A,B
while B ≠ 0
   if b0 = 1 then
      P ← P + A
   end if
   left shift A
   right shift B
end while
```

Example:

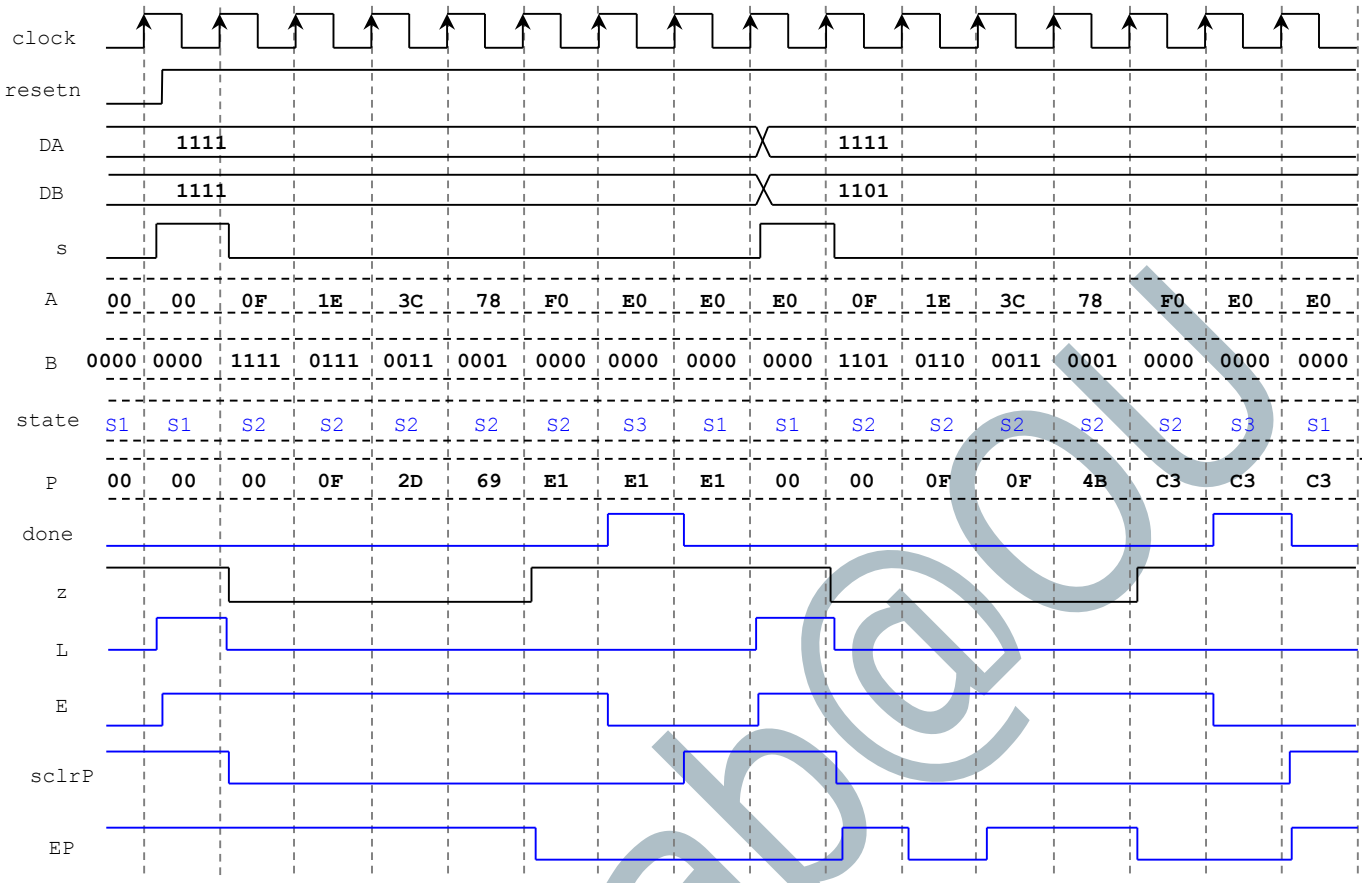
```
          1 1 1 1 x
          1 1 0 1
          ---------
          1 1 1 1   ⟶ P ← 0 + 1111
        0 0 0 0     ⟶ P ← 1111
      1 1 1 1       ⟶ P ← 1111 + 111100 = 1001011
    1 1 1 1         ⟶ P ← 1001011 + 1111000 = 11000011
  -----------------
  1 1 0 0 0 0 1 1
```

```
P ← 0, A ← 1111, B ← 1101
b0=1 ⇒ P ← P + A = 1111.          A ← 11110, B ← 110
b0=0 ⇒ P ← P = 1111.              A ← 111100, B ← 11
b0=1 ⇒ P ← P + A = 1111 + 111100 = 1001011.    A ← 1111000, B ← 1
b0=1 ⇒ P ← P + A = 1001011 + 1111000 = 11000011.  A ← 11110000, B ← 0
```

- Iterative Multiplier Architecture: FSM + Datapath circuit.
  $sclr$: synchronous clear. In this case, if $sclr = 1$ and $E = 1$, the register contents are initialized to 0.
  The solution is computed in $M + 1$ cycles (worst-case scenario)

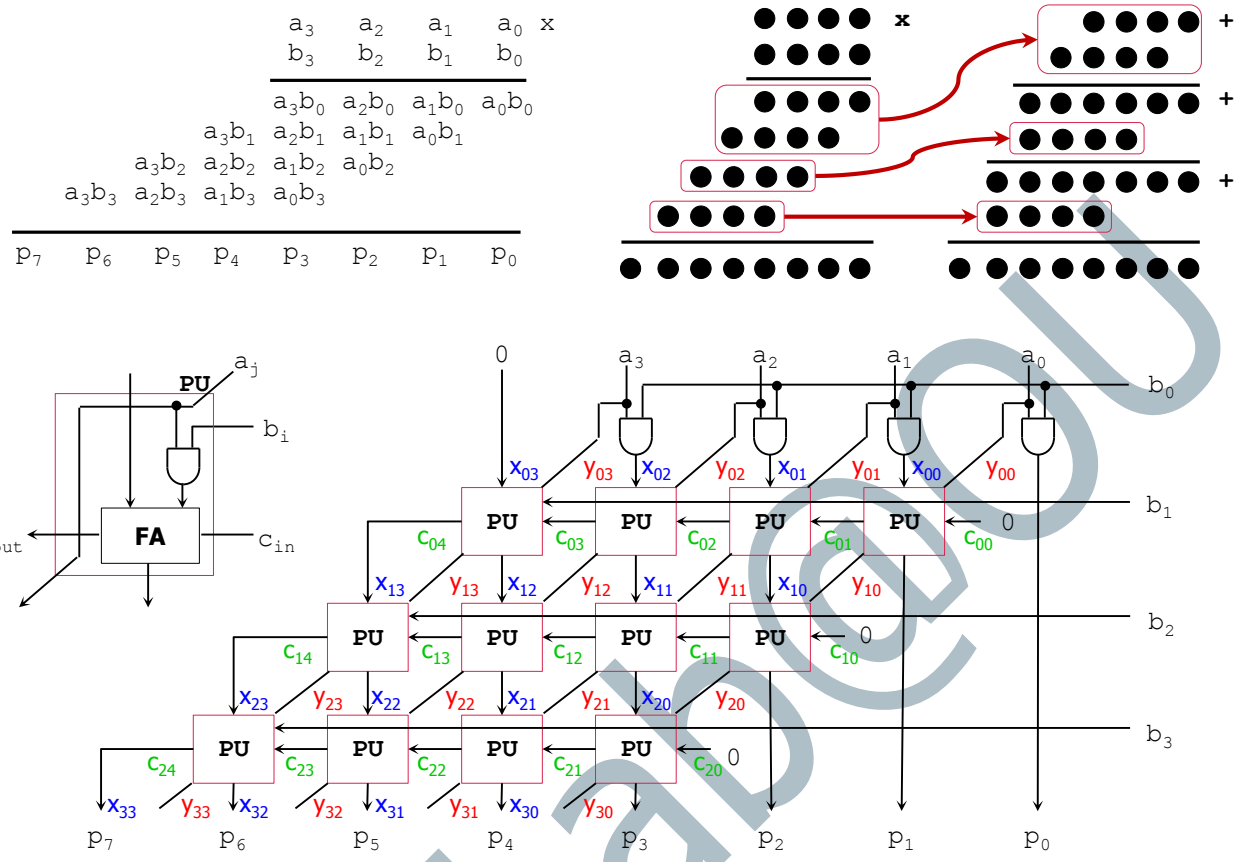


Example (timing diagram):



COMBINATIONAL ARRAY MULTIPLIER

- In this unfolded version (purely combinational), we have a different hardware for every summation of two rows.



PIPELINED ARRAY MULTIPLIER

- To increase the frequency of operation, we place registers at every stage. We also include an enable and a valid output.