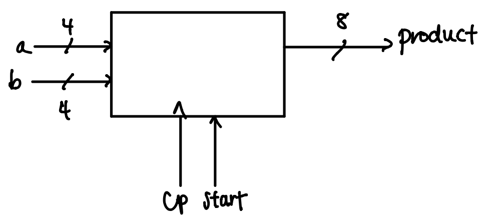
I completed this assignment entirely on my own except for discussion with Janice Tsai.

4-bit sequential multiplier blackbox diagram:

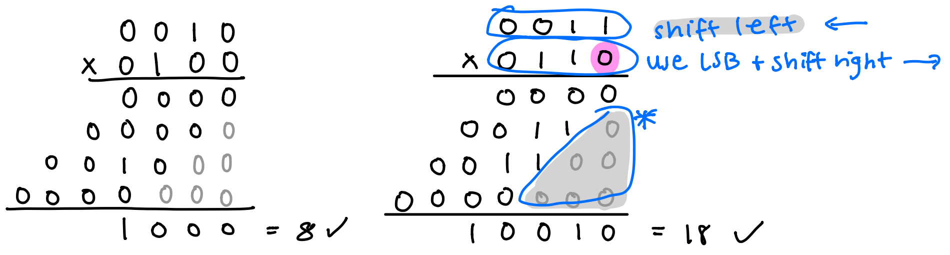


Examples: 2 × 4

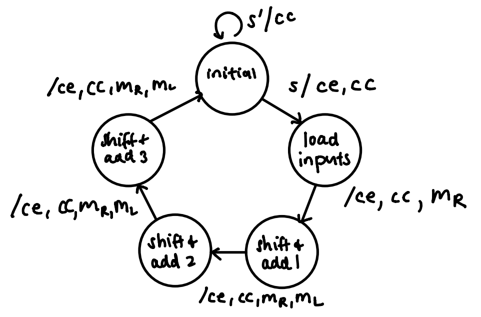
$a = 2 = (0010)_2$
$b = 4 = (0100)_2$

3 × 6

$a = 3 = (0011)_2$
$b = 6 = (0110)_2$



FSM state diagram



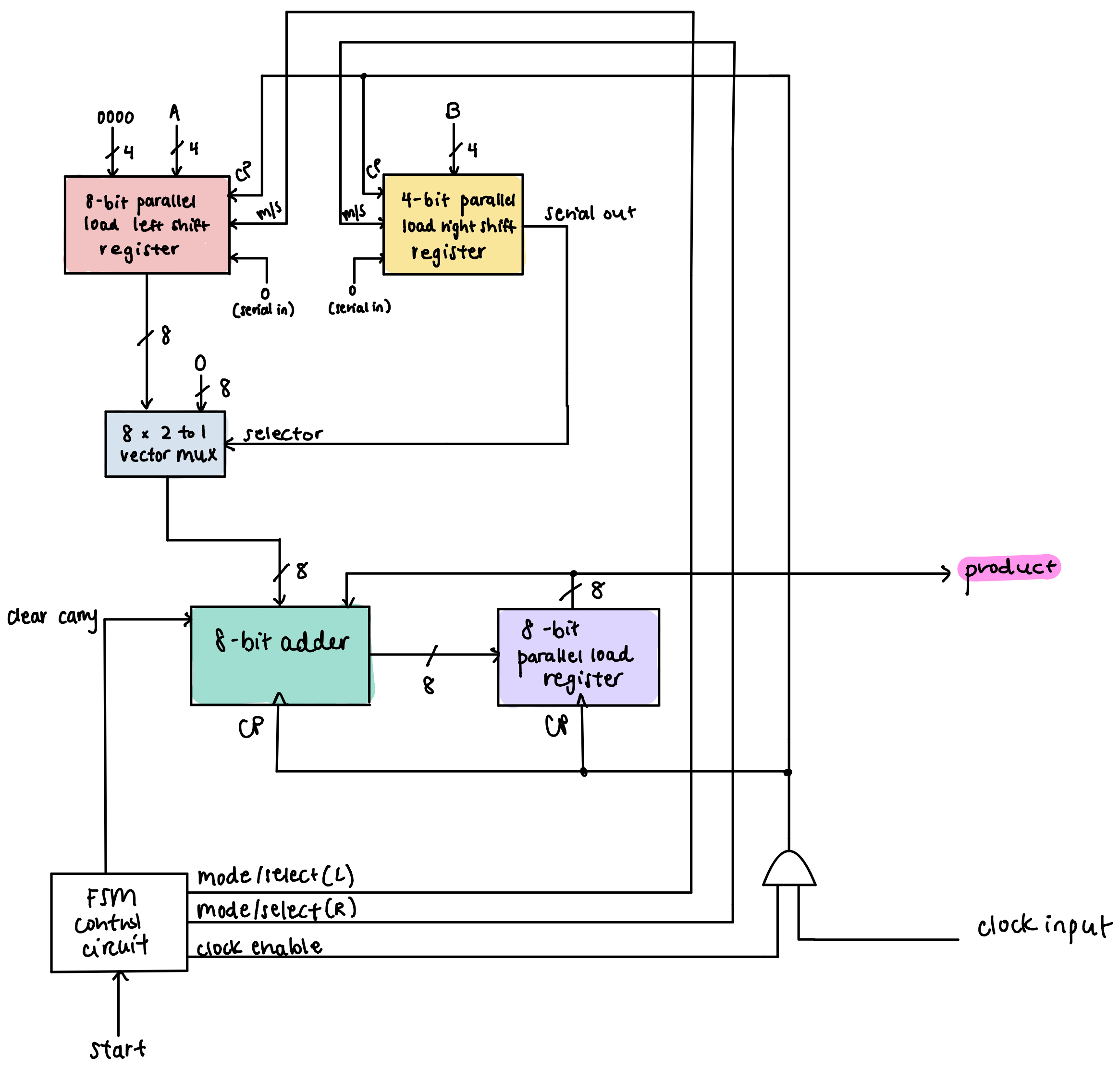
s = start signal
ce = clock enable
m = mode → mode = 0 parallel load
mode = 1 serial load
cc = clear carry

m_R = m/s for 4-bit parallel load right shift register
m_L = m/s for 8-bit parallel load left shift register

FSM I/O:

Inputs: start, clock
Output: clock enable, clear carry, mode/select (R), mode select (L)

Inputs: A, B, start, clock
output: 8-bit product


0000
A
4
4
B
4
CP
CP
8-bit parallel load left shift register
4-bit parallel load right shift register
m/s
m/s
serial out
0 (serial in)
0 (serial in)
8
0
8
8 x 2 to 1 vector mux
selector
8
8
8-bit adder
8
8-bit parallel load register
product
clear carry
CP
CP
FSM control circuit
mode/select(L)
mode/select(R)
clock enable
clock input
start

circuit description

- When the start signal is given, the clear carry clears the carry in the 8-bit adder, the clock is enabled, and both mode/selects are set to 0.

- Our inputs A and B are loaded into the 8-bit parallel load left shift register + 4-bit parallel load right shift register respectfully.

- After the inputs are loaded, mode/select(R)= 1 because we want the LSB of the multiplier. mode/select (L) = 0 still because the first partial multiplication must be w/ the original multiplicand, not shifted left at all.

- The LSB of the 4-bit parallel load right shift register is outputted and used as the selector of the 8 x 2 to 1 mux because we will only add the contents of the multiplicand, which is stored in the 8-bit parallel load left shift register, to the accumulating product if the LSB of the multiplier is 1. Otherwise, we just add 0.

- Then, add the output of the mux to the current product being stored in the 8-bit parallel load register with the use of an 8-bit adder and store it back into the 8-bit register.

- This is repeated for 3 cycles, where mode/select (L) = mode/select (R) = 1, clear carry = 1, and clock enable = 1.
  - mode/selects are both 1 b/c we want to shift both inputs*
  - clear carry remains 1 because we perform independent additions each cycle
  - clock enable remains 1 to continue providing CP to our registers + adder

- our circuit has performed 4 shifts and adds, and our product will be completely computed.