Lecture 8:

Add-Shift Multiplier

Add-Shift Multiplier

- Problem: multiply unsigned binary numbers

```
          1 1 0 1  (13)
          1 0 1 1  (11)
          -------
          1 1 0 1
        1 1 0 1
      -----------
    1 0 0 1 1 1
    0 0 0 0
    -------------
    1 0 0 1 1 1
  1 1 0 1
-----------------
1 0 0 0 1 1 1 1  (143)
```

Add-Shift Multiplier

- Multiplication requires
  - 4-bit Multiplicand and Muliplier registers
  - 8-bit Product register
  - 4-bit adder

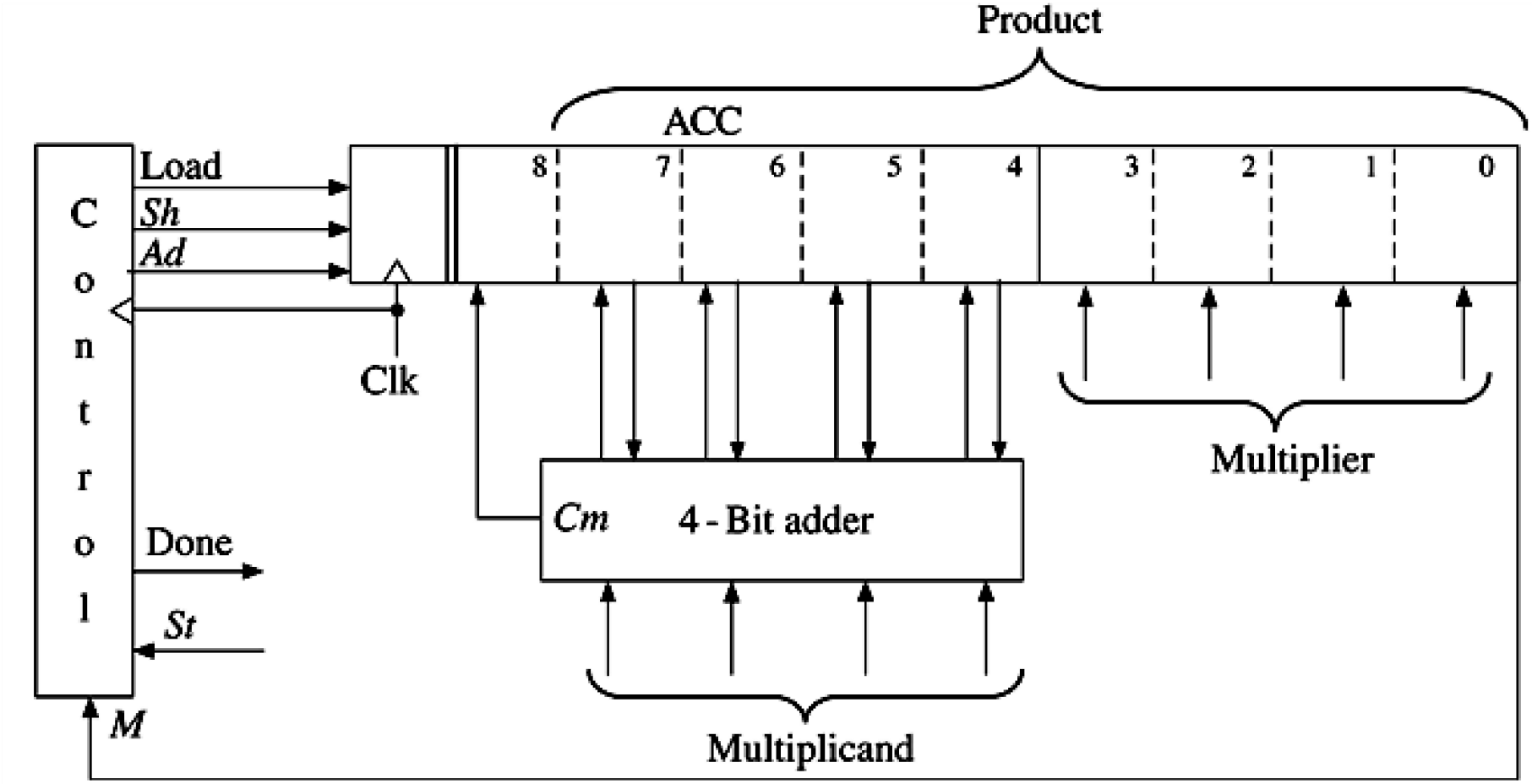
```
          1 1 0 1  (13)
          1 0 1 1  (11)
          -------
          1 1 0 1
        1 1 0 1
      1 0 0 1 1 1
      0 0 0 0
      -----------
      1 0 0 1 1 1
    1 1 0 1
  -------------
1 0 0 0 1 1 1 1  (143)
```

Add-Shift Multiplier

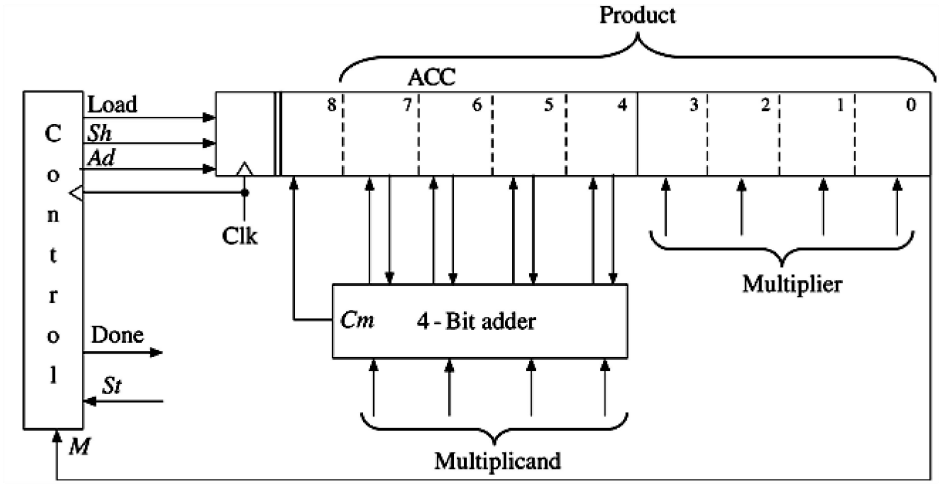
- Product register serves as an accumulator to store sum of partial products
- If you shift Multiplicand to the left each time you would need 8-bit register
- Instead, shift the contents of Product register to the right

```
          1 1 0 1  (13)
          1 0 1 1  (11)
          -------
          1 1 0 1
        1 1 0 1
        -------
      1 0 0 1 1 1
      0 0 0 0
      -------
      1 0 0 1 1 1
    1 1 0 1
    -------
1 0 0 0 1 1 1 1  (143)
```

Add-Shift Multiplier



Add-Shift Multiplier



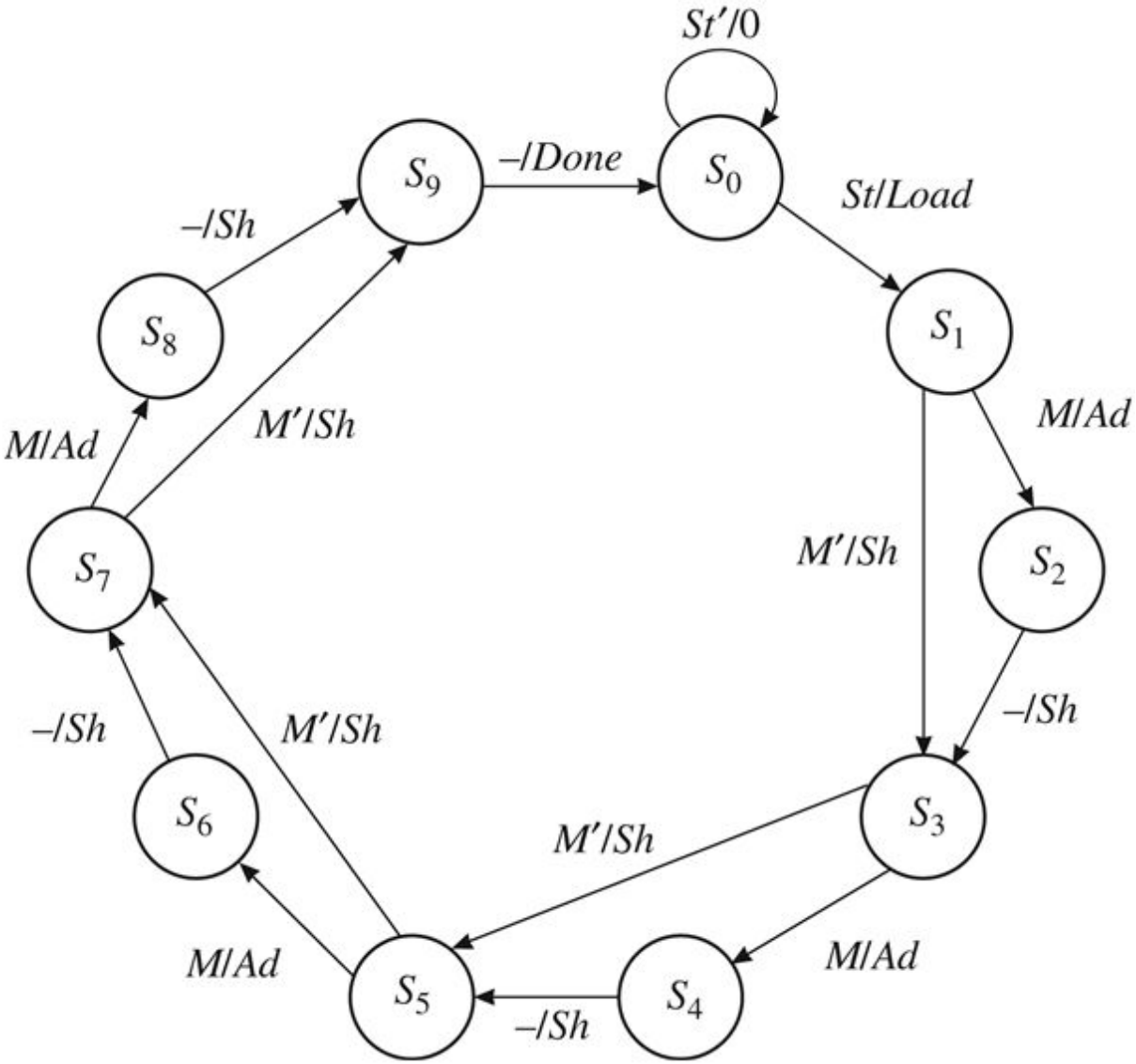
1) Multiplying two 4-bit numbers will generate an 8-bit number

2) The 9th bit is reserved for the internal carry during the addition

initial contents of product register	0 0 0 0 0 | 1 0 1 1 ← M	(11)
(add multiplicand since $M = 1$)	1 1 0 1	(13)
after addition	0 1 1 0 1 | 1 0 1 1	
after shift	0 0 1 1 0 1 | 1 0 1 ← M	
(add multiplicand since $M = 1$)	1 1 0 1	
after addition	1 0 0 1 1 1 | 1 0 1	
after shift	0 1 0 0 1 1 1 | 1 0 ← M	
(skip addition since $M = 0$)		
after shift	0 0 1 0 0 1 1 1 | 1 ← M	
(add multiplicand since $M = 1$)	1 1 0 1	
after addition	1 0 0 0 1 1 1 1 | 1	
after shift (final answer)	0 1 0 0 0 1 1 1 | 1	(143)

dividing line between product and multiplier

Add-Shift Multiplier: Controller



Add-Shift Multiplier: VHDL Code

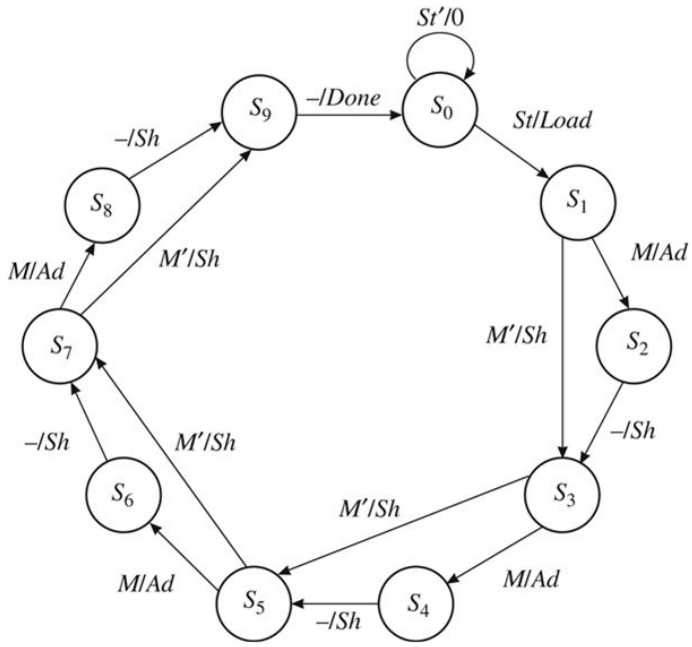

```
-- This is a behavioral model of a multiplier for unsigned
-- binary numbers. It multiplies a 4-bit multiplicand
-- by a 4-bit multiplier to give an 8-bit product.

-- The maximum number of clock cycles needed for a
-- multiply is 10.

library IEEE;
use IEEE.numeric_bit.all;

entity mult4X4 is
  port(Clk, St: in bit;
       Mplier, Mcand: in unsigned(3 downto 0);
       Done: out bit);
end mult4X4;

architecture behave1 of mult4X4 is
signal State: integer range 0 to 9;
signal ACC: unsigned(8 downto 0); -- accumulator
alias M: bit is ACC(0);      -- M is bit 0 of ACC
begin
  process(Clk)
  begin
    if Clk'event and Clk = '1' then  -- executes on rising edge of clock
      case State is
        when 0 =>                -- initial State
          if St = '1' then
            ACC(8 downto 4) <= "00000"; -- begin cycle
            ACC(3 downto 0) <= Mplier;  -- load the multiplier
            State <= 1;
          end if;
```

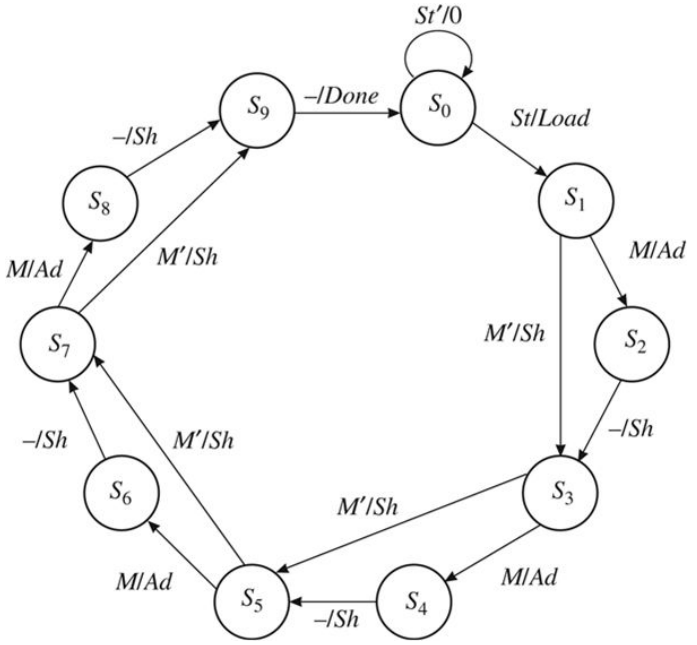


can delcare Alias

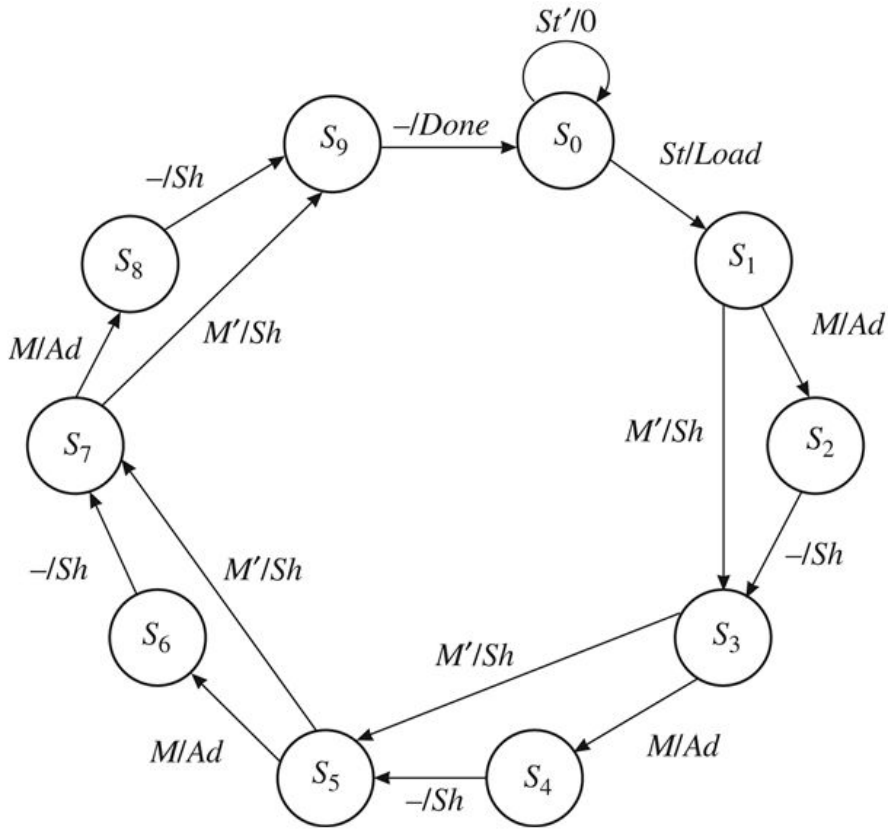
Add-Shift Multiplier: VHDL Code


```
        when 1 | 3 | 5 | 7 =>     -- "add/shift" State
          if M = '1' then         -- add multiplicand
            ACC(8 downto 4) <= '0' & ACC(7 downto 4) + Mcand;
            State <= State + 1;
          else
            ACC <= '0' & ACC(8 downto 1);      -- shift accumulator right
            State <= State + 2;
          end if;
        when 2 | 4 | 6 | 8 =>                  -- "shift" State
          ACC <= '0' & ACC(8 downto 1);        -- right shift
          State <= State + 1;
        when 9 =>                              -- end of cycle
          State <= 0;
      end case;
    end if;
  end process;
  Done <= '1' when State = 9 else '0';
end behave1;
```

Concurrent statement because **Done** should be updated whenever in state 9 and not during transition from 9 to 0 otherwise too late

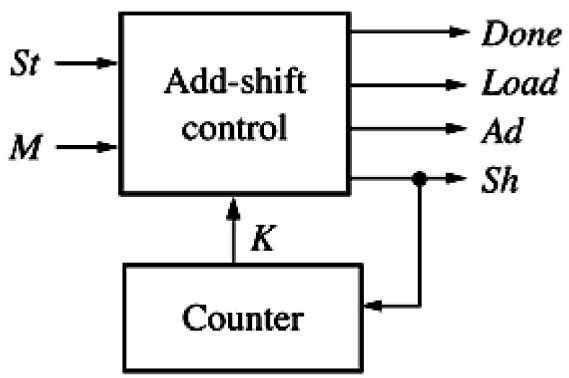


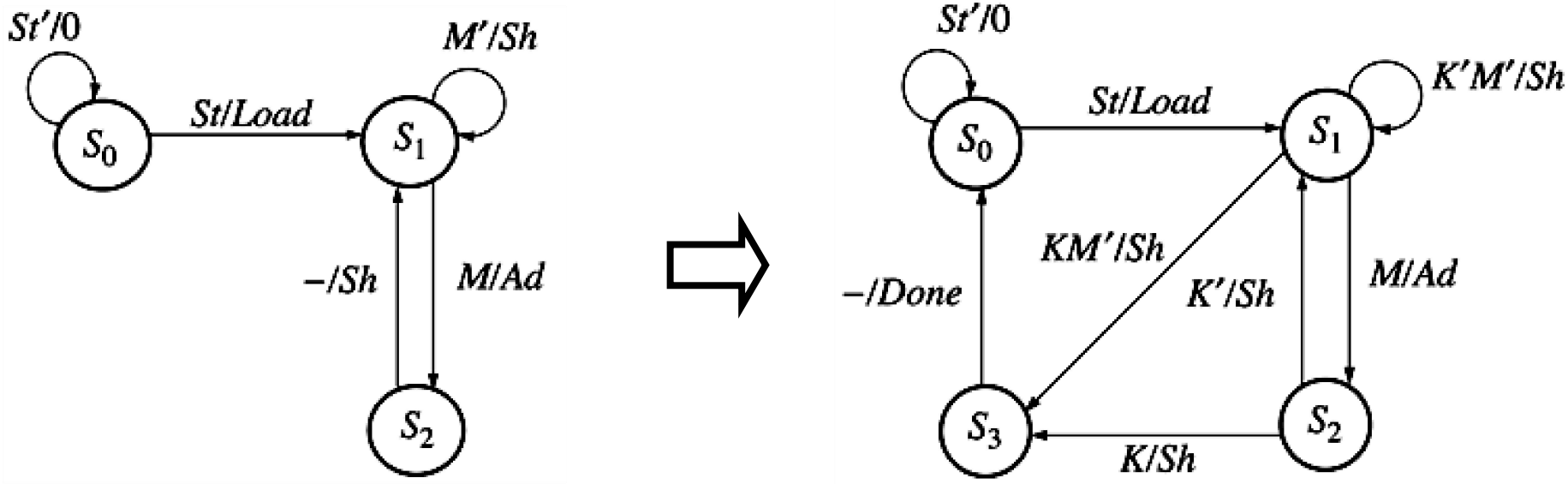
Add-Shift Multiplier: Large Numbers



- In the above state machine, two functions are performed
  - generate add or shift
  - count number of shifts (depends on number of multiplier bits)

Add-Shift Multiplier: Large Numbers





For n-bit multiplier, n number of shifts are required