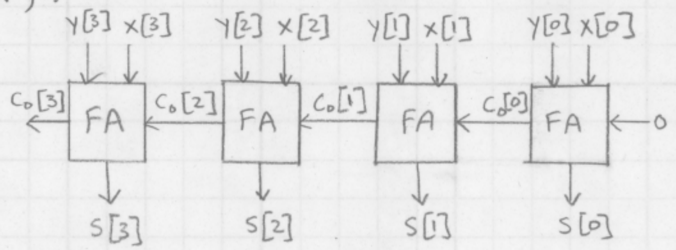
We've looked at fully combinational circuits for addition (the Ripple-Carry Adder):



Let's examine a combinational logic circuit for multiplication:

Array Multiplier

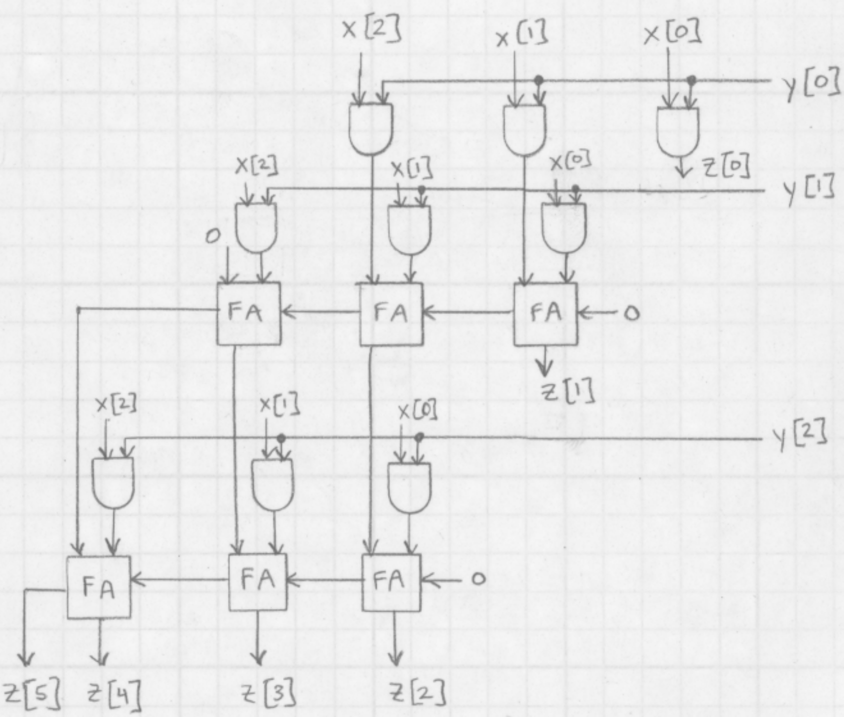
Recall binary (unsigned) multiplication involved creating binary "Partial Products":

Ex: $110_2 = 6_{10}$

$\times\ 101_2 = 5_{10}$

```
   110 }
  000  }  Use AND gates to generate partial products (X·0=0 and X·1=X).
(+)110 }
 11110₂  = 30₁₀ ✓
```

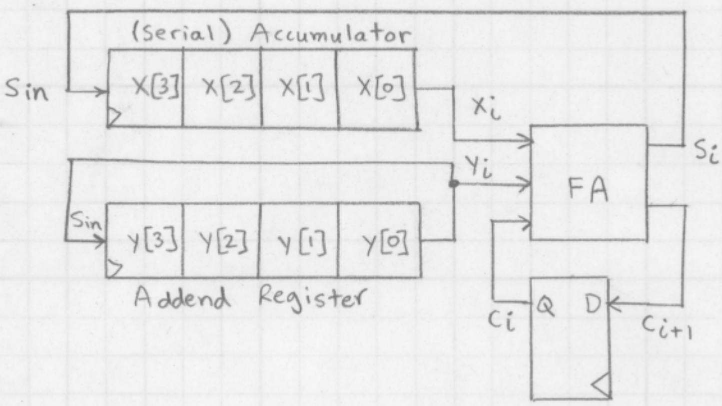
Use ripple carry adders to sum partial products.



Note: Two 3-bit operands yield a 6 bit product.
See Rabaey, Digital Integrated Circuits, for more information.

18.1 Serial Adder With Accumulator

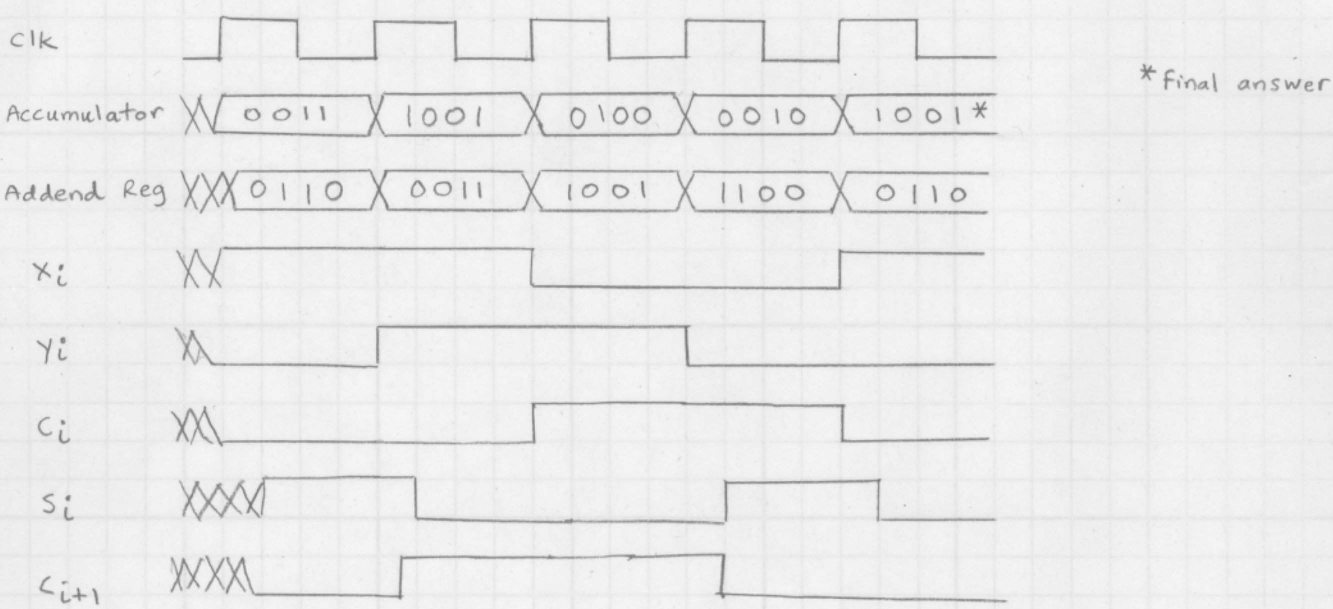
One can also exploit sequential circuits to do arithmetic ⇒ yields less area, less power (under certain circumstances). The serial adder uses a flip-flop and shift registers to time the carry propagation correctly:



On successive cycles, new addend bits and the carry out from the previous clock cycle are present at the Full Adder inputs.

<u>Ex:</u>

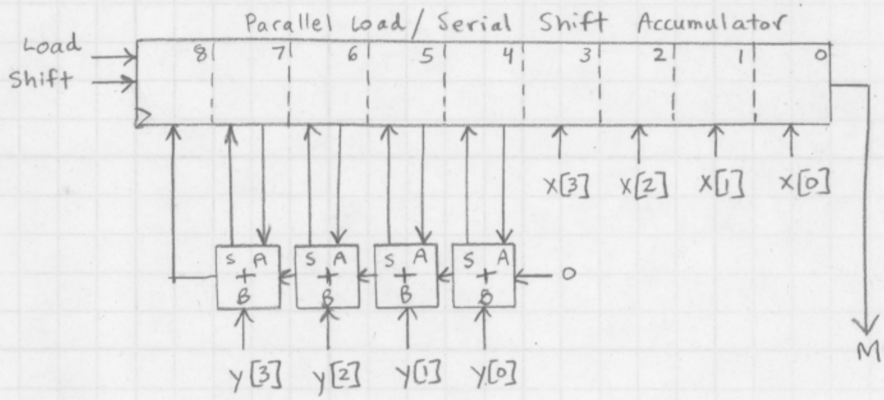
$$\begin{array}{rcr} 3_{10} & = & 0011_2 \\ +\ 6_{10} & = & +0110_2 \\ \hline 9_{10} & & 1001_2 \end{array}$$



To control shifting, clearing of flip-flops, signaling when result is computed, we need a control FSM.

18.2 Serial (Shift and Add) Multiplication

A serial multiplier can be made using a shift register and a ripple-carry adder to sum successive partial products into an accumulator:



Bits 0-7 store product.

M ≡ controls when a new partial product is accumulated.

Ex: $3_{10} \times 4_{10} = 12_{10}$ → $0011_2 \times 0100_2 = 00001100_2$

Divides product from multiplier ↓ (¦)

① ACC 0 0 0 0 ¦ 0 0 1 1
Partial Prod. 0 1 0 0
M = 1
} Add and shift (M=1)

② ACC 0 0 1 0 0 ¦ 0 0 1
Partial Prod. 0 1 0 0
M = 1
} Add and shift (M=1)

③ ACC 0 0 1 1 0 0 ¦ 0 0
Partial Prod. 0 1 0 0
M = 0
} Shift (M=0)

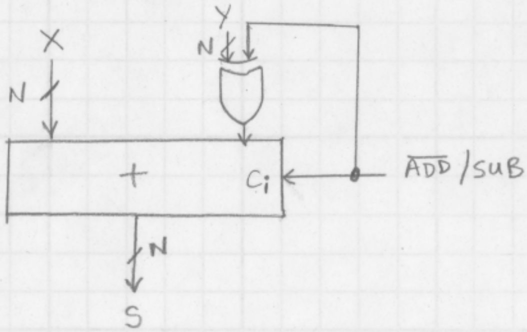
④ ACC 0 0 0 1 1 0 0 ¦ 0
Partial Prod. 0 1 0 0
M = 0
} Shift (M=0)

⑤ ACC 0 0 0 0 1 1 0 0 $= 12_{10}$ ✓ Done
Partial Prod. 0 1 0 0

Extensions to Adder Circuits

Adding basic logic gates to adders yields multiple functions:

Adder / Subtractor ($X, Y \equiv$ 2's complement binary numbers)

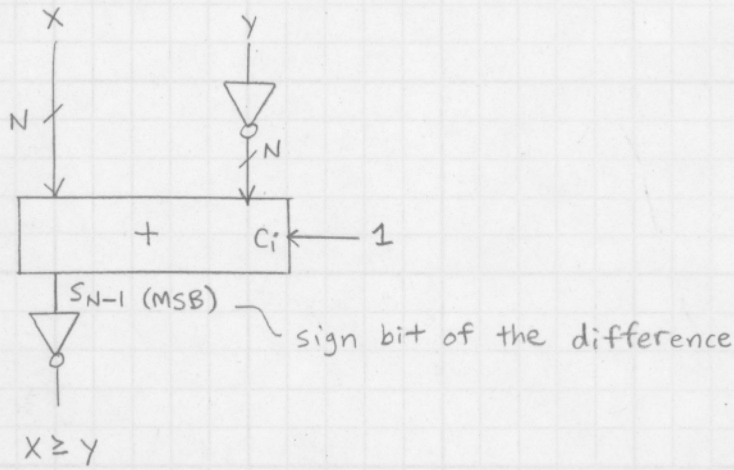


$\overline{ADD}/SUB = 0 \rightarrow S = X + Y$

$\overline{ADD}/SUB = 1 \rightarrow S = X - Y = X + (-Y)$

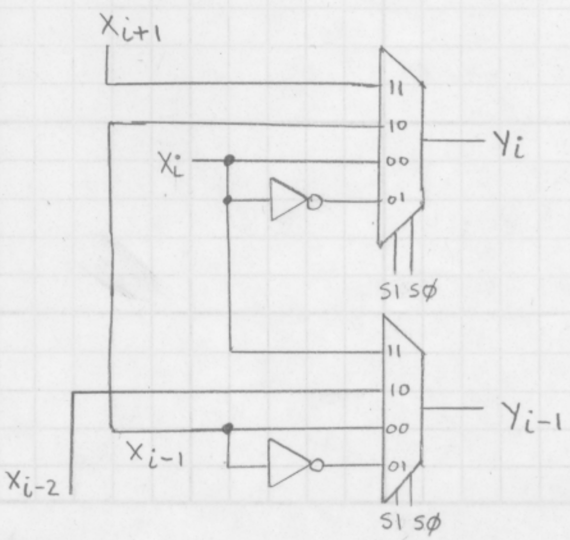
(Note: $Y \oplus 1 = \overline{Y}$)

Comparator ($X, Y \equiv$ 2's complement)



Shifters

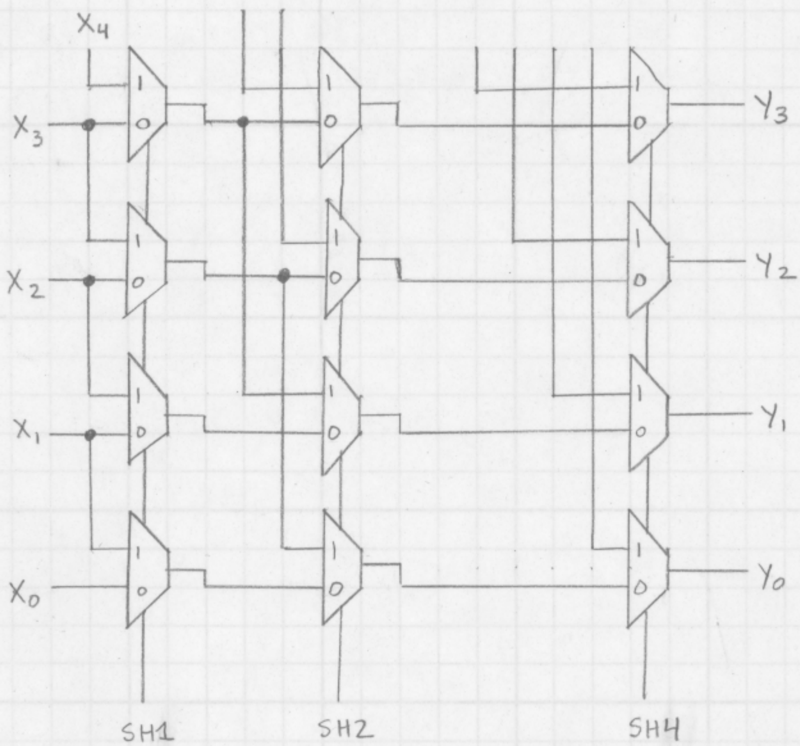
Fixed bit shifts (or rotates) can be done with wiring. Programmable shifting requires active circuits (muxes or tri-state buffers).



S1	S∅	Y_i	
0	0	X_i	buffer
0	1	X_i'	inverter
1	0	X_{i-1}	left shift
1	1	X_{i+1}	right shift

Logarithmic Shifter

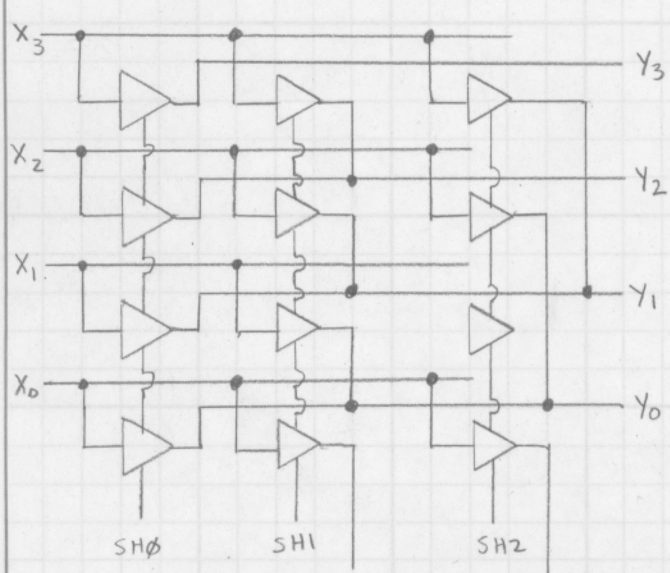
A logarithmic shifter uses a tree of muxes to implement a maximum M bit shift in $\log_2 M$ stages.



SH1	SH2	SH4	Y_0
0	0	0	X_0
1	0	0	X_1
0	1	0	X_2
1	1	0	X_3
	⋮		⋮

Barrel Shifter

A barrel shifter uses an array of switches or tri-state buffers to implement shifting with a constant propagation delay.



SHØ	SH1	SH2	Y_0
1	0	0	X_0
0	1	0	X_1
0	0	1	X_2