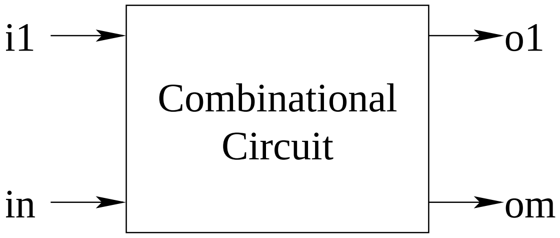
COMBINATIONAL CIRCUITS I

(Adders, Decoders, Multiplexers)

CC are circuits *without* memory where the outputs are obtained from the inputs only. A n-input m-output combinational circuit is of the form



where, $o_i = f(i_1, \ldots, i_n),\ 1 \leq i \leq m.$

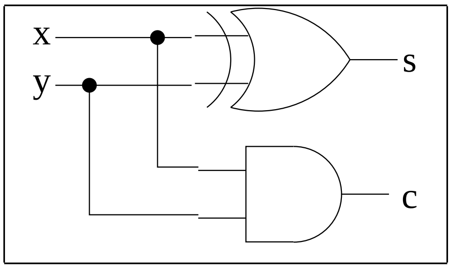
- Half/Full adder/substractor
- Binary ripple adder/substractor
- Look-ahead carry adder
- Code converter, Encoder/Decoder
- Magnitude comparator
- Multiplexer/Demultiplexer
- Read-only memory
- Programmable logic array
- Programmable array logic
- Arithmetic logic unit

ADDERS AND SUBSTRACTORS

Half Adder Adds 2 bits and outputs carry and sum

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

$\mapsto$



$c =$

$s =$

Full Adder Adds previous carry and 2 bits and outputs new carry and sum

x	y	z	c	s
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$\mapsto$

K-map for c

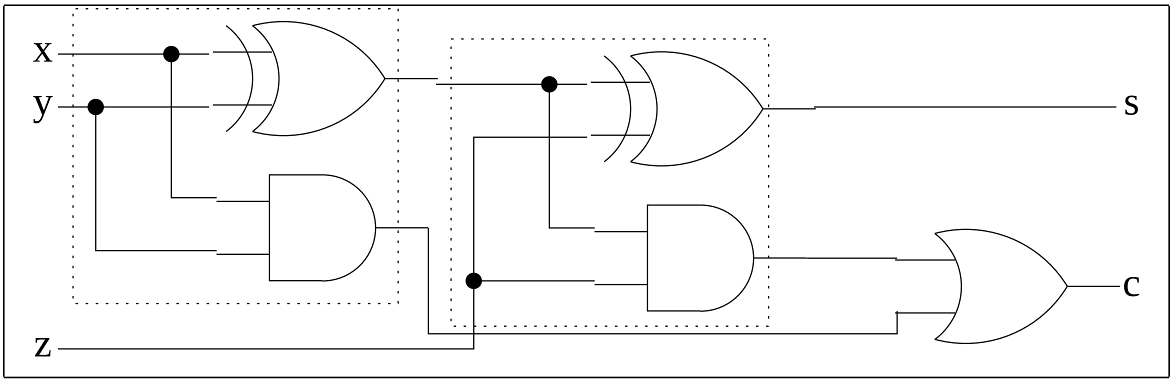
$x\backslash yz$	00	01	11	10
0			1	
1		1	1	1

$c =$

K-map for s

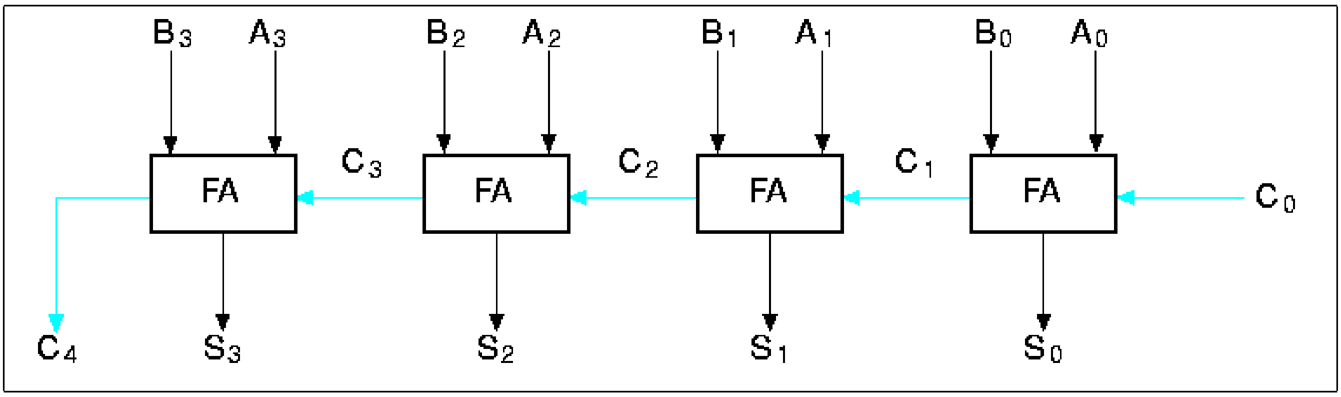
$x\backslash yz$	00	01	11	10
0		1		1
1	1		1	

$s =$



Binary Ripple Carry Adder

Adds 2 n-bit numbers and outputs carry-out and sum



n FAs are connected in cascade

Input carry of FA$_i$ = output carry of FA$_{i-1}$

Simple, easy and *cheap* design

Reusable

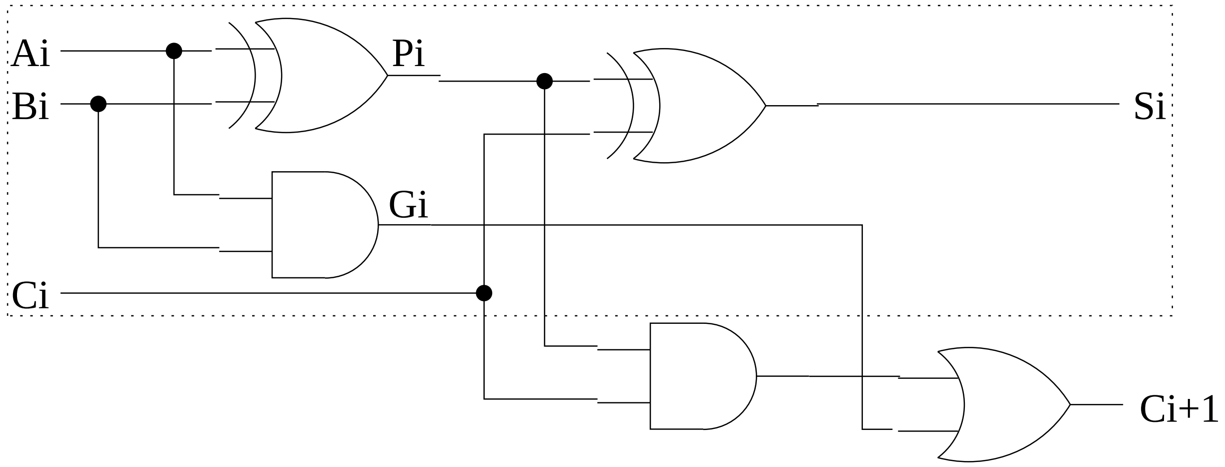
Very *slow*

Can you design a faster n-bit adder circuit?

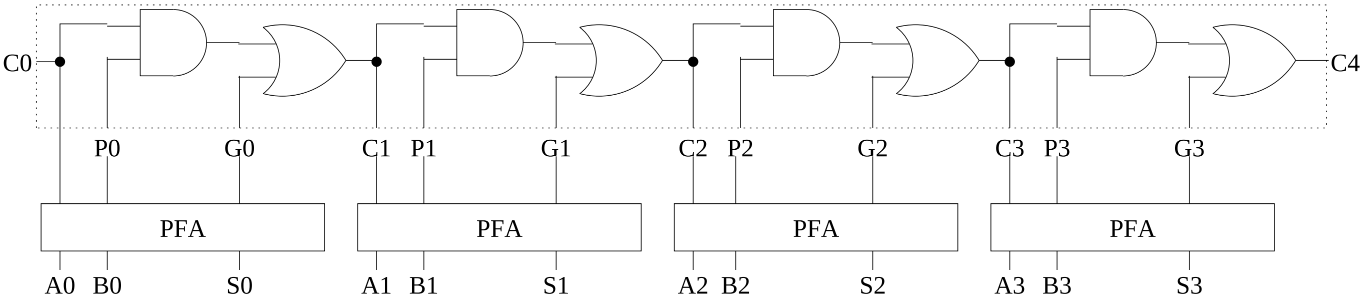
Can you design a smaller n-bit adder circuit?

Carry Lookahead Adder

Faster than BRCA but more complex
FA = Partial Full Adder + Ripple Carry Path



So for a 4-bit BRCA we have 8 gate delays in the RCP



A solution to BRCA problem is to replace the RCP by a two-level circuit (2 gate delays)

How: Each C_i is a function of G_{i-1}, P_{i-1} and C_{i-1}, that is $C_i = f(G_{i-1}, P_{i-1}, C_{i-1}) = G_{i-1} + P_{i-1}C_{i-1}$. Generate a SOP expression for each C_i and obtain its two-level circuit implementation.

Carry Lookahead Adder (continued)

$$C_i = G_{i-1} + P_{i-1}C_{i-1} \text{ for } 1 \le i \le n$$

$$\begin{aligned}
C_1 &= G_0 + P_0C_0 \\
C_2 &= G_1 + P_1C_1 \\
&= G_1 + P_1(G_0 + P_0C_0) \\
&= G_1 + P_1G_0 + P_1P_0C_0 \\
C_3 &= G_2 + P_2C_2 \\
&= G_2 + P_2(G_1 + P_1G_0 + P_1P_0C_0) \\
&= G_2 + P_2G_1 + P_2P_1G_0 + P_2P_1P_0C_0 \\
C_4 &= G_3 + P_3C_3 \\
&= G_3 + P_3(G_2 + P_2G_1 + P_2P_1G_0 + P_2P_1P_0C_0) \\
&= G_3 + P_3G_2 + P_3P_2G_1 + P_3P_2P_1G_0 \\
&\qquad + P_3P_2P_1P_0C_0 \\
&= G_{0\cdots3} + P_{0\cdots3}C_0 \\
G_{0\cdots3} &= G_3 + P_3G_2 + P_3P_2G_1 + P_3P_2P_1G_0 \\
P_{0\cdots3} &= P_3P_2P_1P_0
\end{aligned}$$

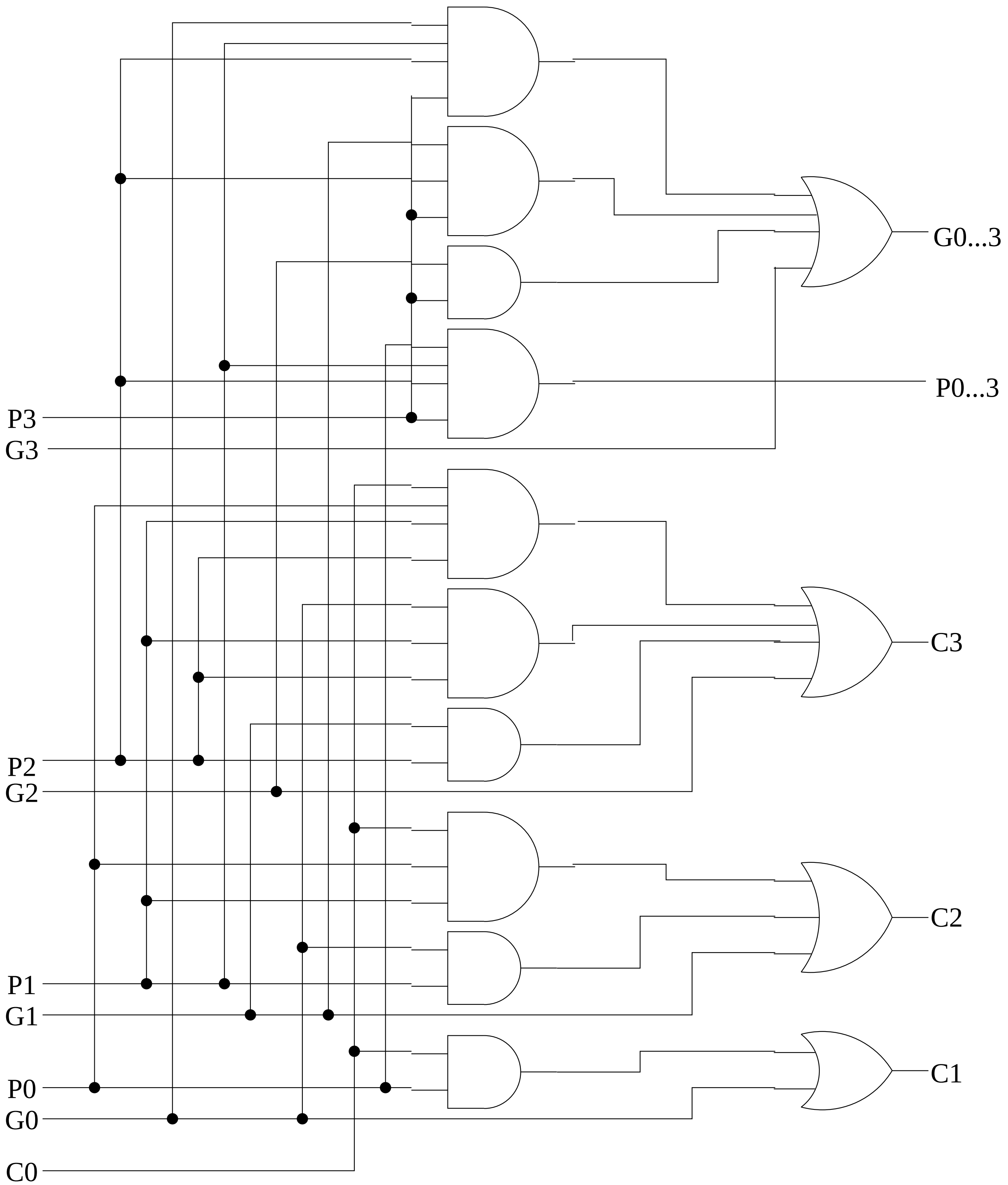
- P_i = Propagate funtion: propagate a carry from position i to $i+1$

- G_i = Generate funtion: generate a carry in position i

- $P_{0\cdots3}$ = Group propagate funtion

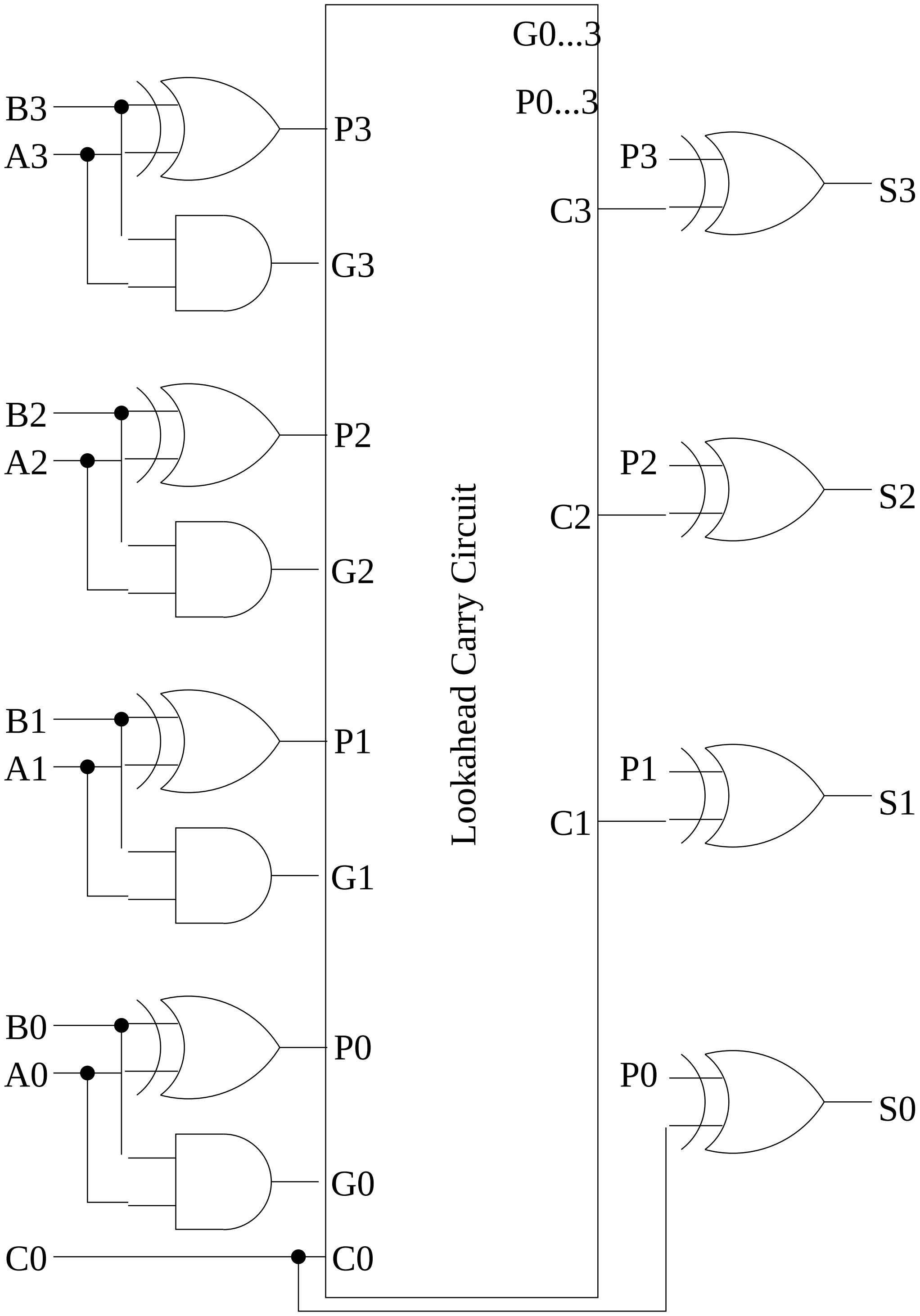
- $G_{0\cdots3}$ = Group generate funtion

Four-Bit Carry Lookahead Circuit



Four-Bit Carry Lookahead Adder (continued)

4-bit FAs with carry lookahead circuit



More expensive than RCP but much more faster

4-bit CLAs can be used to implement circuits for adding $8, 12, 16, \ldots, 4n$-bit numbers

Substractors

Half Substractor, Full Substractor, Binary Ripple Borrow Substractor and Borrow Lookahead Substractor can be defined in the same way as done for the adders. Instead of *carry* and *sum* we have *borrow* and *difference*, and different truth table and circuits

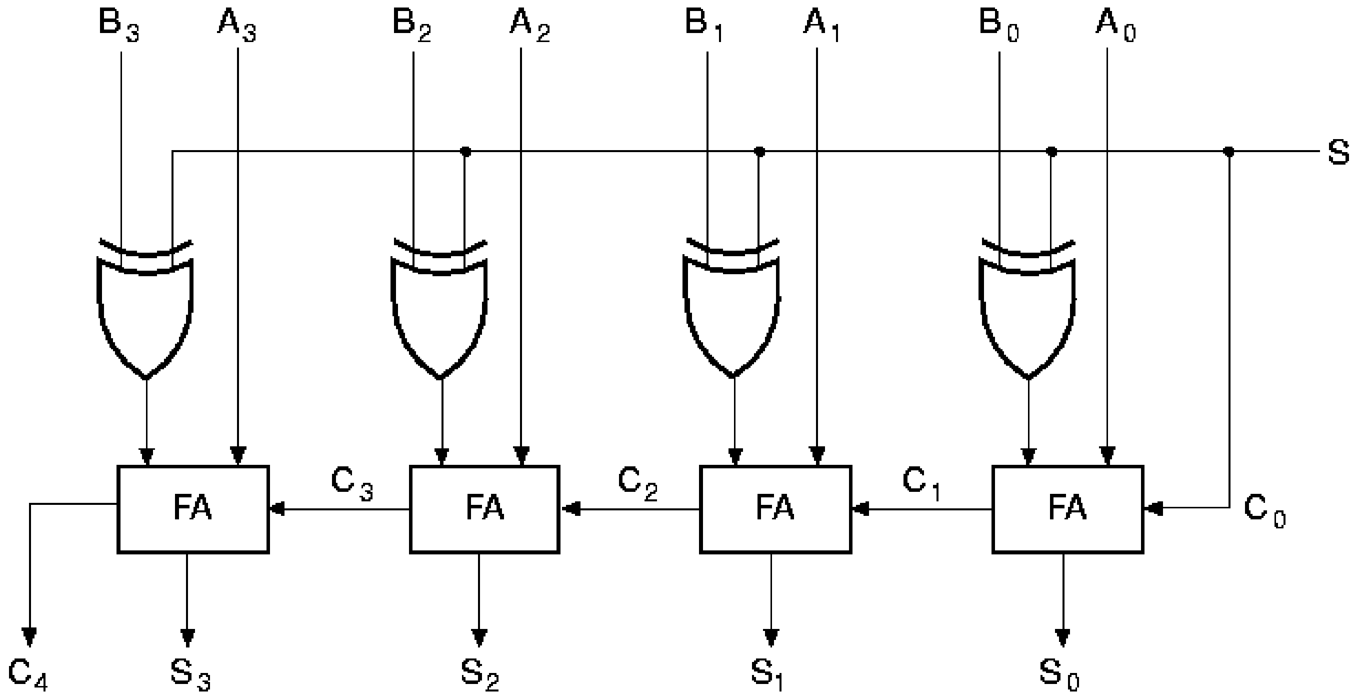
Exercise to submit in next lab Define and implement a Half Substractor, a Full Substractor and a Binary Ripple Borrow Substractor. You'll have a bonus if you implement a Borrow Lookahead Substractor. **Note: it'll be marked**

Do we really need substractor circuits?

If not then how can we implement substraction operation?

Hint: Remember the relationship between binary addition and binary substraction operations

Adder-Substractor Circuit



Circuit is a

Adder When $S = 0$:
$A + B$

Substractor When $S = 1$:
$A + (2\text{CF of } B) = A - B$