# Addition & Subtraction

# Addition and Subtraction

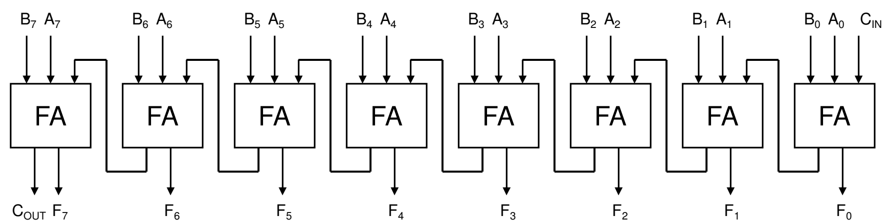
- Motivation: need to add and/or subtract values represented as binary numbers
  - Note need for identification of coding method: unsigned binary, two's complement, fixed point, floating point, excess code, …
  - Scheme should be extensible; that is, once method identified, should be able to apply method with varying numbers of bits

## Adding Two 8-bit Numbers

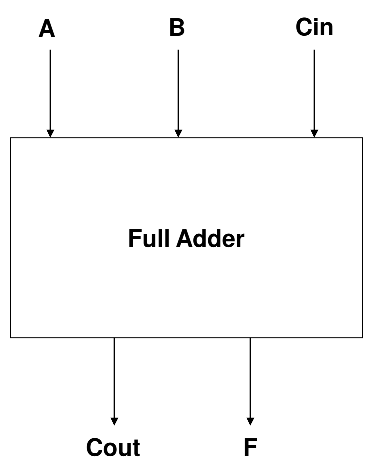
```
  1 1 1 0 1 1 1 0
  0 0 1 0 0 0 1 0
  ---------------
  0 0 0 1 0 0 0 0
```

## Adding Two 8-bit Numbers

```
 1 1 1 0 1 1 1 0 0
  1 1 1 0 1 1 1 0
  0 0 1 0 0 0 1 0
  ---------------
  0 0 0 1 0 0 0 0
```



$B_7$ $A_7$
$B_6$ $A_6$
$B_5$ $A_5$
$B_4$ $A_4$
$B_3$ $A_3$
$B_2$ $A_2$
$B_1$ $A_1$
$B_0$ $A_0$ $C_{IN}$
FA
FA
FA
FA
FA
FA
FA
FA
$C_{OUT}$ $F_7$
$F_6$
$F_5$
$F_4$
$F_3$
$F_2$
$F_1$
$F_0$

A
B
Cin
Full Adder
Cout
F

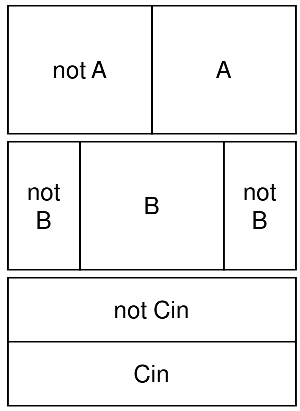
## Full Adder Truth Table

| A | B | Cin | Cout | F |
|---|---|---|---|---|
| 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 |

## K-Map for Three Variables

| | | | A | A |
|---|---|---|---|---|
| | x | x | x | x |
| Cin | x | x | x | x |
| | | B | B | |

## K-Map Construction




## Full Adder – K-Map for F

| | | | A | A |
|---|---|---|---|---|
| | 0 | 1 | 0 | 1 |
| Cin | 1 | 0 | 1 | 0 |
| | | B | B | |

## Full Adder – Kmap for F

| | | | A | A |
|---|---|---|---|---|
| | 0 | 1 | 0 | 1 |
| Cin | 1 | 0 | 1 | 0 |
| | | B | B | |

$$F = \overline{A} \bullet \overline{B} \bullet C + \overline{A} \bullet B \bullet \overline{C} + A \bullet B \bullet C + A \bullet \overline{B} \bullet \overline{C}$$

## Full Adder – K-Map for Cout

| | | | A | A |
|---|---|---|---|---|
| | 0 | 0 | 1 | 0 |
| Cin | 0 | 1 | 1 | 1 |
| | | B | B | |

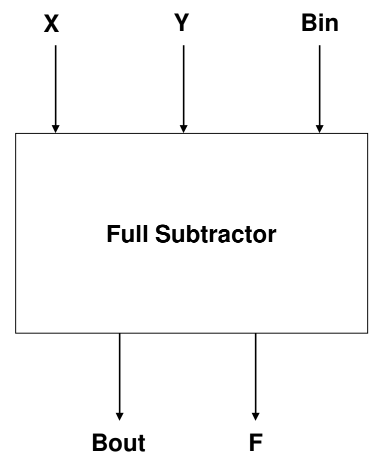
## Full Adder – K-Map for Cout

| | | | A | A |
|---|---|---|---|---|
| | 0 | 0 | 1 | 0 |
| Cin | 0 | 1 | 1 | 1 |
| | | B | B | |

## Full Adder – K-Map for Cout

| | | | A | A |
|---|---|---|---|---|
| | 0 | 0 | 1 | 0 |
| Cin | 0 | 1 | 1 | 1 |
| | | B | B | |

$$\mathrm{Cout} = \mathrm{B} \bullet \mathrm{C} + \mathrm{A} \bullet \mathrm{B} + \mathrm{A} \bullet \mathrm{C}$$




Full Subtractor Truth Table

| X | Y | Bin | Bout | F |
|---|---|---|---|---|
| 0 | 0 | 0 | | |
| 0 | 0 | 1 | | |
| 0 | 1 | 0 | | |
| 0 | 1 | 1 | | |
| 1 | 0 | 0 | | |
| 1 | 0 | 1 | | |
| 1 | 1 | 0 | | |
| 1 | 1 | 1 | | |

## Full Subtractor Truth Table

| X | Y | Bin | Bout | F |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 |

## Full Subtractor – K-Map for F

| | | | X | |
|---|---|---|---|---|
| | 0 | 1 | 0 | 1 |
| Bin | 1 | 0 | 1 | 0 |
| | | Y | | |

$$F = \overline{X} \bullet \overline{Y} \bullet B + \overline{X} \bullet Y \bullet \overline{B} + X \bullet Y \bullet B + X \bullet \overline{Y} \bullet \overline{B}$$

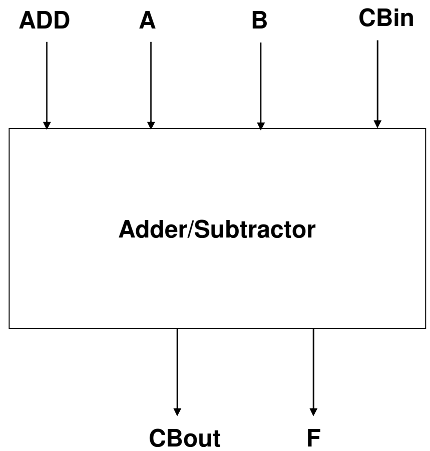
## Full Subtractor – K-Map for Bout

| | | | X | X |
|---|---|---|---|---|
| | 0 | 1 | 0 | 0 |
| Bin | 1 | 1 | 1 | 0 |
| | | Y | Y | |

## Full Subtractor – K-Map for Bout

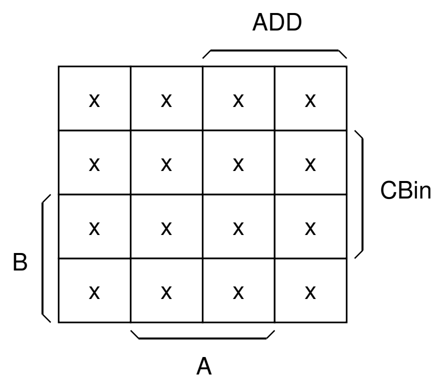
| | | | X | X |
|---|---|---|---|---|
| | 0 | 1 | 0 | 0 |
| Bin | 1 | 1 | 1 | 0 |
| | | Y | Y | |

$$\text{Bout} = \overline{X} \bullet B + \overline{X} \bullet Y + Y \bullet B$$

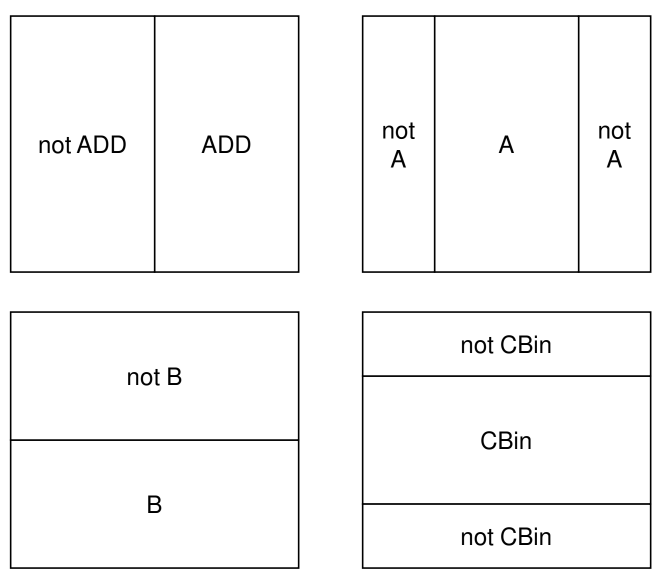



| ADD | A | B | CBin | Cbout | F |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 |

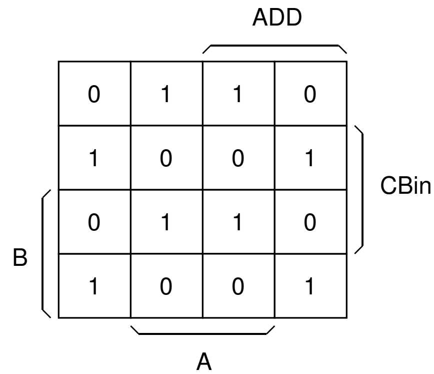
## K-Map For Four Variables



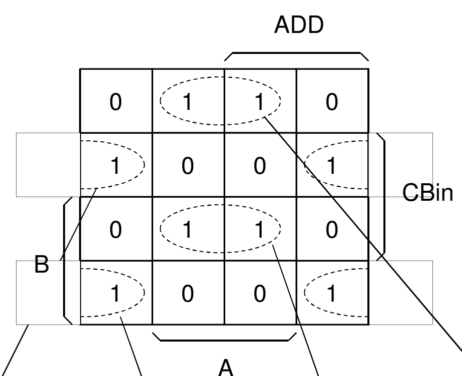

## K-Map Construction (4 bits)



# K-Map For F




# K-Map For F




$$F = \overline{A} \bullet \overline{B} \bullet CBin + \overline{A} \bullet B \bullet \overline{CBin} + A \bullet B \bullet CBin + A \bullet \overline{B} \bullet \overline{CBin}$$

## K-Map For CBout

ADD

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 |

CBin

B

A

## K-Map For CBout

ADD

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 |

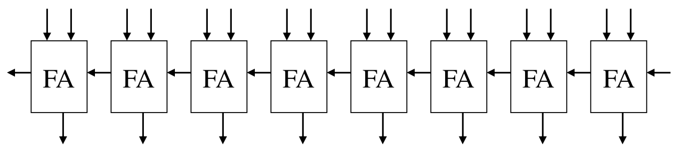
CBin

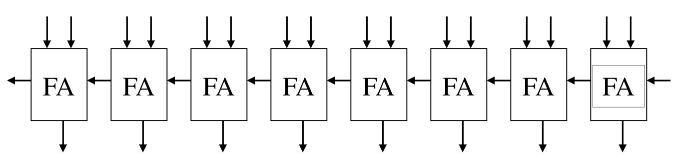
B

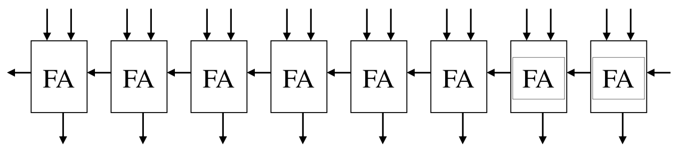
A

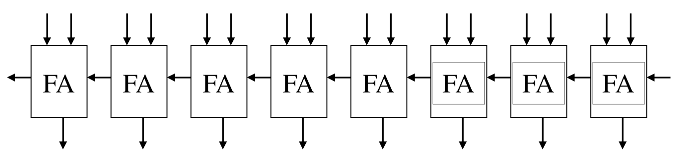
$$\mathrm{CBout} = \overline{\mathrm{ADD}} \bullet \overline{\mathrm{A}} \bullet \mathrm{CBin} + \overline{\mathrm{ADD}} \bullet \overline{\mathrm{A}} \bullet \mathrm{B} + \mathrm{ADD} \bullet \mathrm{A} \bullet \mathrm{CBin} + \mathrm{ADD} \bullet \mathrm{A} \bullet \mathrm{B} + \mathrm{B} \bullet \mathrm{CBin}$$

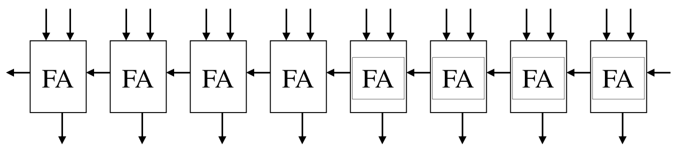
# Timing for Ripple Carry Adder

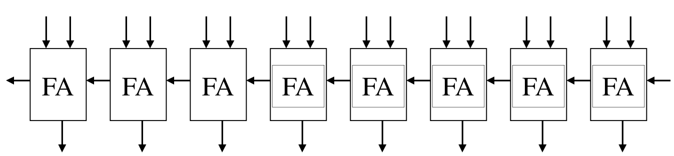


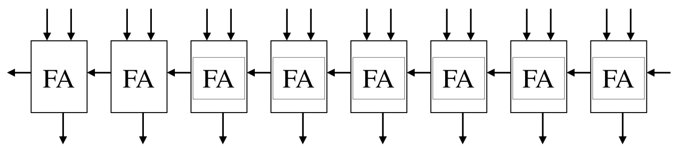



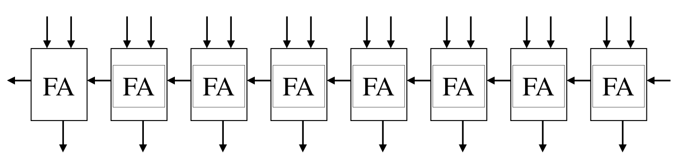

FA FA FA FA FA FA FA FA

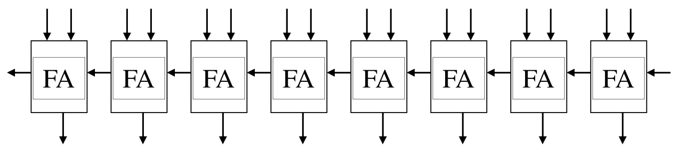

FA FA FA FA FA FA FA FA


FA
FA
FA
FA
FA
FA
FA
FA

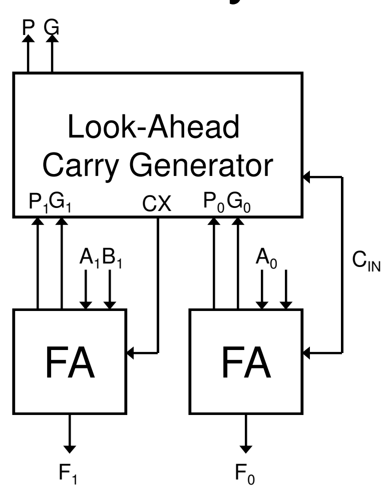

FA
FA
FA
FA
FA
FA
FA
FA


FA
FA
FA
FA
FA
FA
FA
FA


FA
FA
FA
FA
FA
FA
FA
FA




# Look Ahead System (2 Bits)



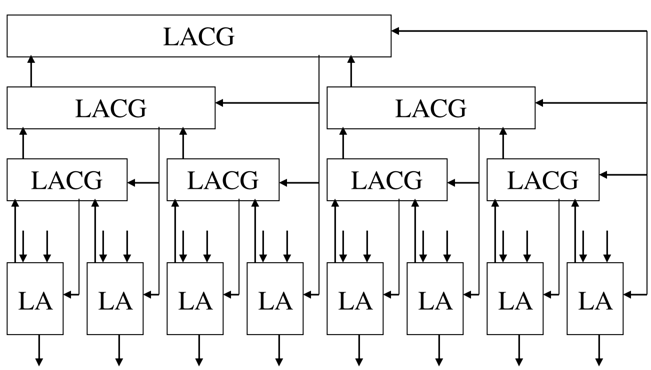
# Look Ahead System (2 Bit)

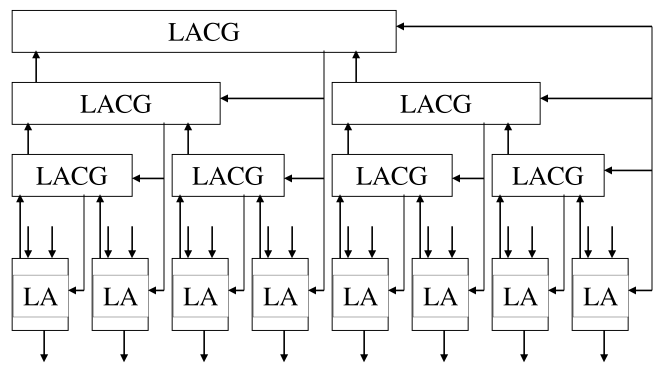
$CX = G_0 + P_0C_{IN}$

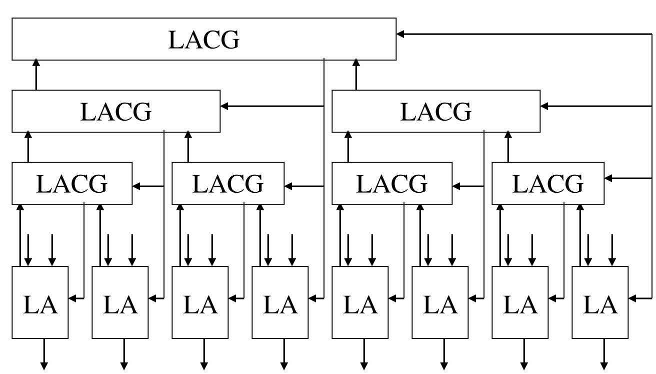
$CY = G_1 + P_1CX$
$\quad = G_1 + P_1G_0 + P_1P_0C_{IN}$

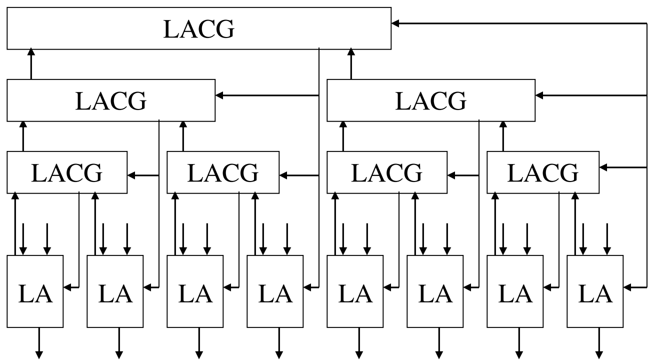
Generate = $G_1 + P_1G_0$
Propagate = $P_1P_0$

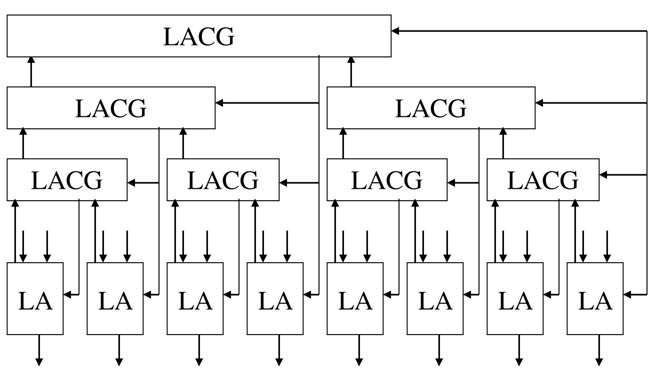
# Look Ahead Method

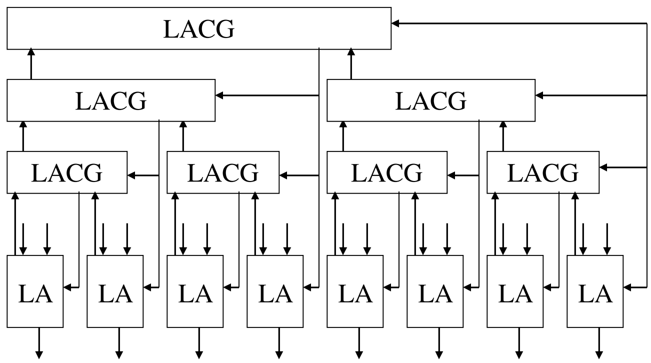


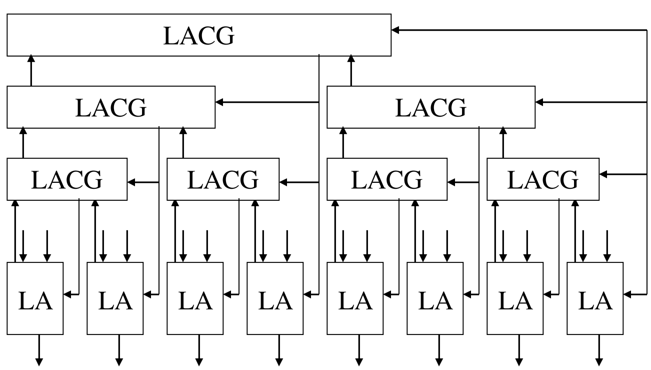

LACG
LACG
LACG
LACG
LACG
LACG
LACG
LA
LA
LA
LA
LA
LA
LA
LA

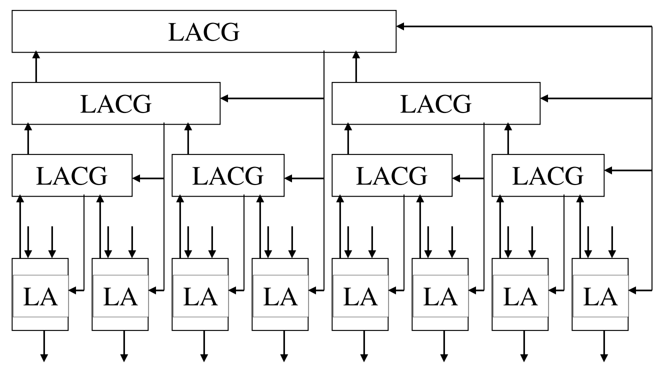

LACG
LACG
LACG
LACG
LACG
LACG
LACG
LA
LA
LA
LA
LA
LA
LA
LA


LACG
LACG
LACG
LACG
LACG
LACG
LACG
LA
LA
LA
LA
LA
LA
LA
LA

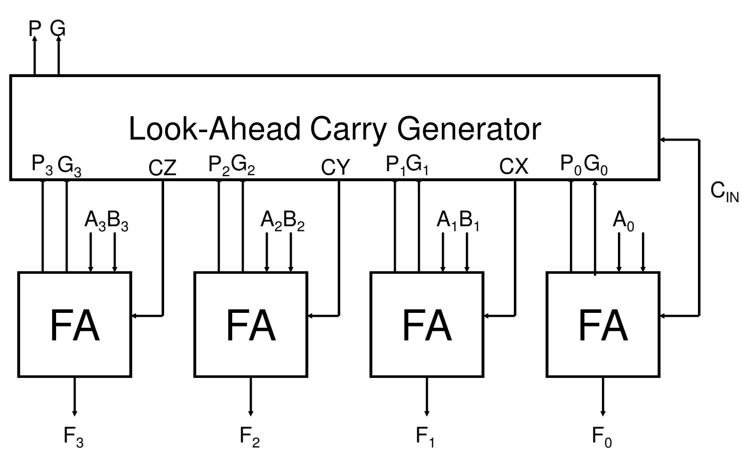

LACG
LACG
LACG
LACG
LACG
LACG
LACG
LA
LA
LA
LA
LA
LA
LA
LA


LACG
LACG
LACG
LACG
LACG
LACG
LACG
LA
LA
LA
LA
LA
LA
LA
LA


LACG
LACG
LACG
LACG
LACG
LACG
LACG
LA
LA
LA
LA
LA
LA
LA
LA




# Look Ahead System



## Look Ahead System

$CX = G_0 + P_0C_{IN}$

$CY = G_1 + P_1CX$
$= G_1 + P_1G_0 + P_1P_0C_{IN}$

$CZ = G_2 + P_2CY$
$= G_2 + P_2G_1 + P_2P_1G_0 + P_2P_1P_0C_{IN}$

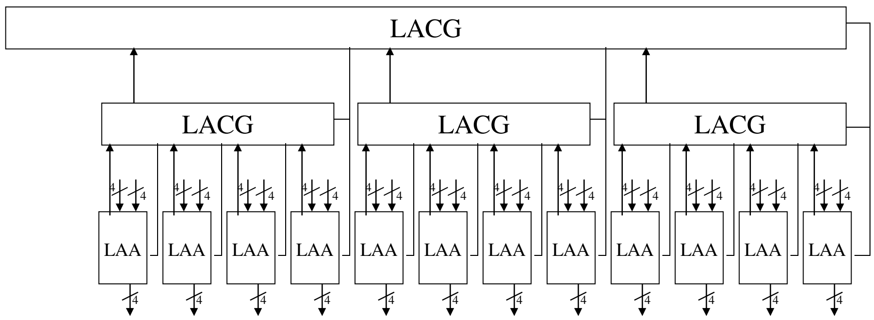
## Look Ahead System

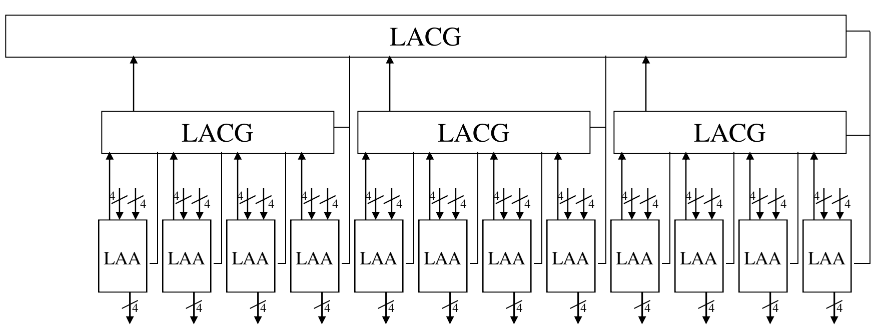
$C_{NEXT} = G_3 + P_3CZ$
$= G_3 + P_3G_2 + P_3P_2G_1 +$
$P_3P_2P_1G_0 + P_3P_2P_1P_0C_{IN}$

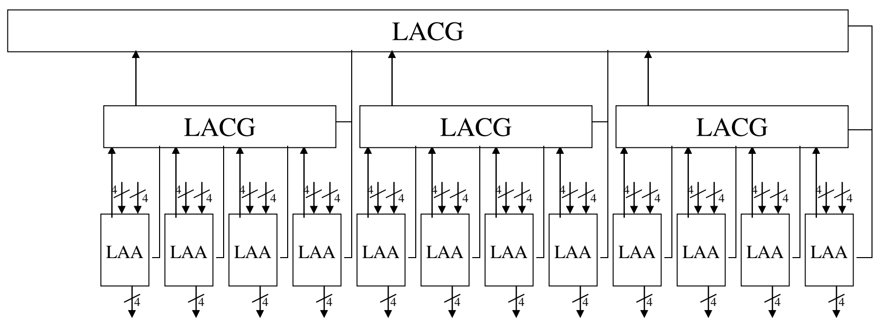
$G_{OUT} = G_3 + P_3G_2 + P_3P_2G_1 +$
$P_3P_2P_1G_0$

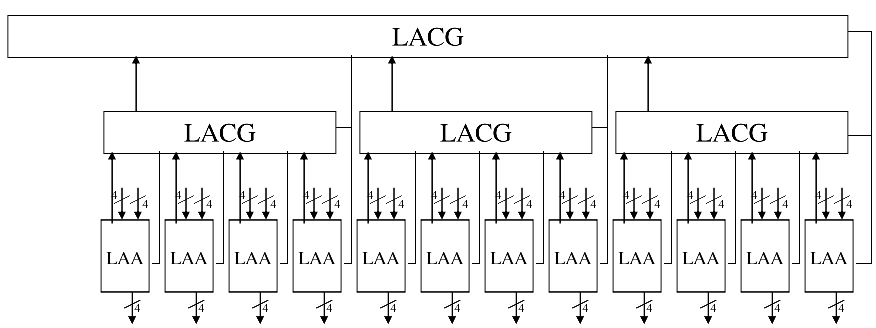
$P_{OUT} = P_3P_2P_1P_0$

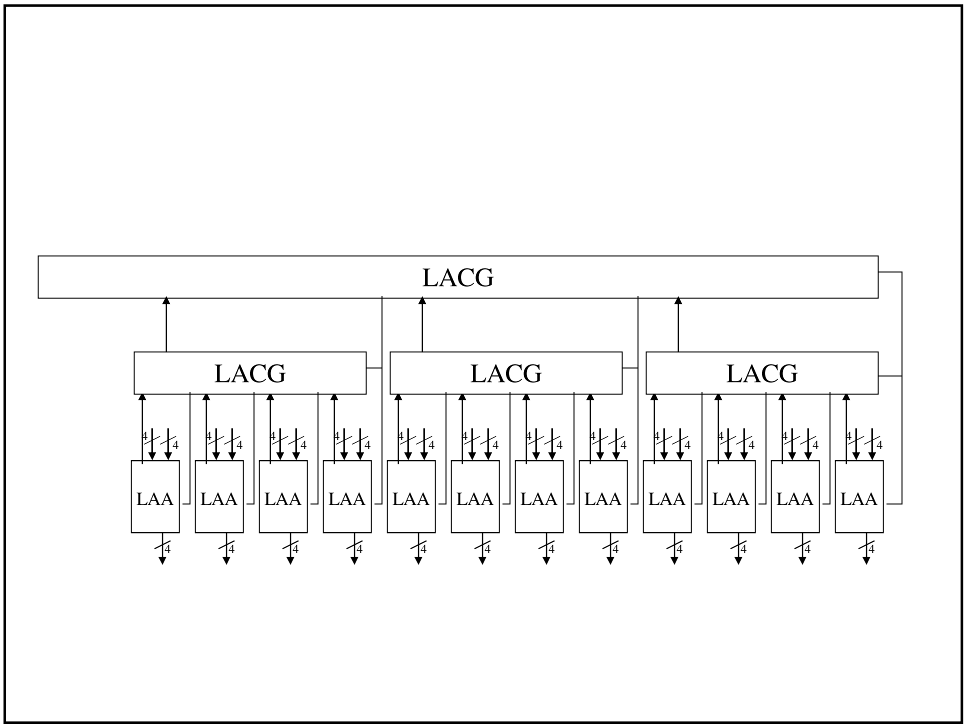
## Look-Ahead, 4 bits wide…

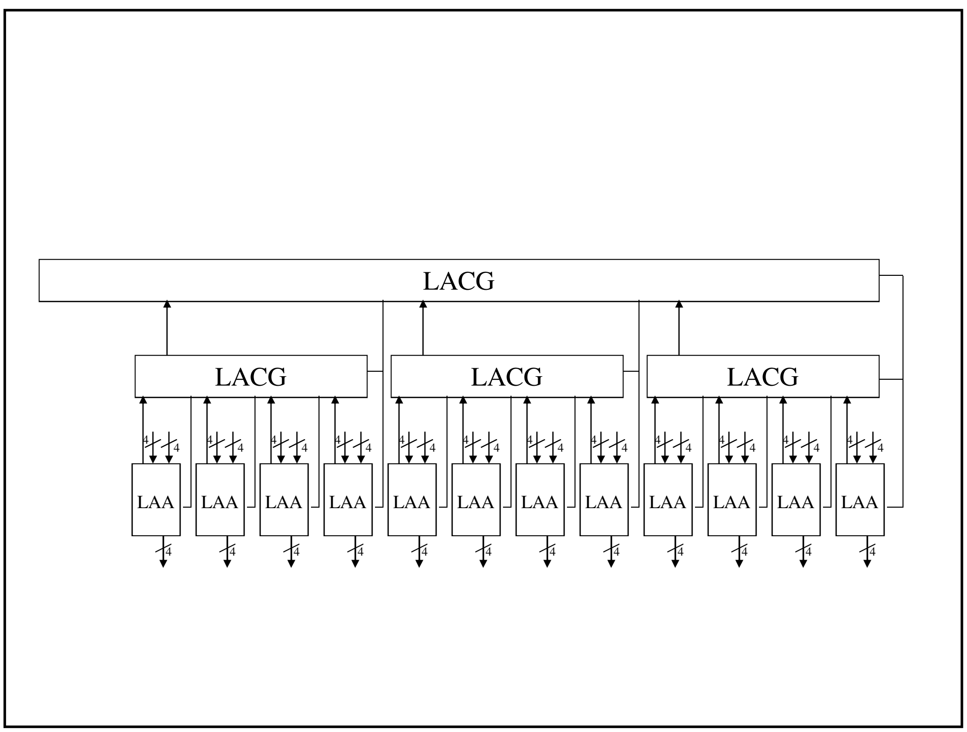



LACG
LACG
LACG
LACG
LAA
4



LACG
LACG
LACG
LACG
LAA
4


LACG
LACG
LACG
LACG
LAA LAA LAA LAA LAA LAA LAA LAA LAA LAA LAA LAA
4


LACG
LACG
LACG
LACG
LAA LAA LAA LAA LAA LAA LAA LAA LAA LAA LAA LAA
4