

## ECE 645: Lecture 3

### Basic Adders and Counters

---

---

---

---

---

---

---

---

### Required Reading

*Behrooz Parhami,*  
*Computer Arithmetic: Algorithms and Hardware Design*

*Chapter 5, Basic Addition and Counting*  
*Sections 5.1-5.5, pp. 75-85.*

---

---

---

---

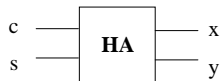
---

---

---

---

### Half-adder


$$x + y = (c \ s)_2$$

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

---

---

---

---

---

---

---

---

### Half-adder

#### Alternative implementations (1)

a)  $c = xy$   
 $s = x \oplus y$

(a) AND/XOR half-adder.

b)  $c = \overline{\overline{x} + \overline{y}}$   
 $s = \overline{xy + \overline{xy}}$

(b) NOR-gate half-adder.

---

---

---

---

---

---

---

---

### Half-adder

#### Alternative implementations (2)

c)  $\overline{c} = \overline{xy}$   
 $s = x\overline{c} + y\overline{c} = \overline{\overline{x\overline{c}} \cdot \overline{y\overline{c}}}$

(c) NAND-gate half-adder with complemented carry.

---

---

---

---

---

---

---

---

### Full-adder

$x + y + c_{in} = (c_{out} \ s)_2$

x	y	c <sub>in</sub>	c <sub>out</sub>	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

---

---

---

---

---

---

---

---

**Full-adder**  
**Alternative implementations (1)**

a)  $s = (x \oplus y) \oplus c_{in}$   
 $c_{out} = xy + c_{in}(x \oplus y)$

(a) Built of half-adders.

---

---

---

---

---

---

---

---

**Full-adder**  
**Alternative implementations (2)**

b)  $c_{out} = xy + xc_{in} + yc_{in}$   
 $s = x \oplus y \oplus c_{in} = xyc_{in} + \bar{x}\bar{y}c_{in} + \bar{x}y\bar{c}_{in} + x\bar{y}\bar{c}_{in}$

(b) Built as an AND-OR circuit.

---

---

---

---

---

---

---

---

**Full-adder**  
**Alternative implementations (3)**

c)

x	y	$c_{out}$	s
0	0	0	$c_{in}$
0	1	$c_{in}$	$\bar{c}_{in}$
1	0	$c_{in}$	$\bar{c}_{in}$
1	1	1	$c_{in}$

(c) Suitable for CMOS realization.

---

---

---

---

---

---

---

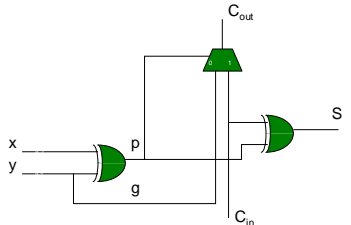
---

### Full-adder

#### Alternative implementations (4)

Implementation used to generate fast carry logic  
in Xilinx FPGAs

x	y	$C_{out}$
0	0	y
0	1	$C_{in}$
1	0	$C_{in}$
1	1	y



$p = x \oplus y$   
 $g = y$   
 $s = p \oplus C_{in} = x \oplus y \oplus C_{in}$

---

---

---

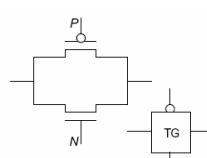
---

---

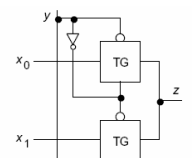
---

---

---



(a) CMOS transmission gate:  
circuit and symbol



(b) Two-input mux built of two  
transmission gates

**CMOS transmission gate and its use in a 2-to-1 mux.**

---

---

---

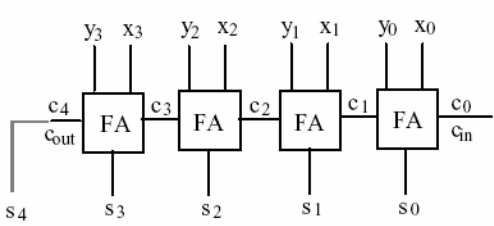
---

---

---

---

---



**(b) Four-bit ripple-carry adder.**

---

---

---

---

---

---

---

---

$$T_{\text{ripple-add}} = T_{\text{FA}}(x, y \rightarrow c_{\text{out}}) + (k - 2) \times T_{\text{FA}}(c_{\text{in}} \rightarrow c_{\text{out}}) + T_{\text{FA}}(c_{\text{in}} \rightarrow s)$$

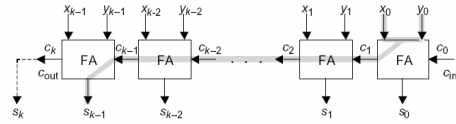


Fig. 5.5 Critical path in a  $k$ -bit ripple-carry adder.

---

---

---

---

---

---

---

---

---

---

### Latency of a $k$ -bit ripple-carry adder

$$T_{\text{ripple-add}} = T_{\text{FA}}(x, y \rightarrow c_{\text{out}}) + (k - 2) \cdot T_{\text{FA}}(c_{\text{in}} \rightarrow c_{\text{out}}) + T_{\text{FA}}(c_{\text{in}} \rightarrow s)$$

$$\text{Latency} \approx k \cdot T_{\text{FA}}$$

$$\text{Latency} \propto k$$

---

---

---

---

---

---

---

---

---

---

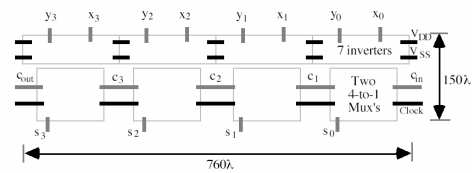


Fig. 5.4 The layout of a 4-bit ripple-carry adder in CMOS implementation [Puck94].

---

---

---

---

---

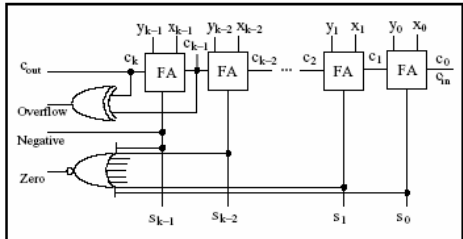
---

---

---

---

---



Two's-complement adder with provisions for detecting conditions and exceptions.

---

---

---

---

---

---

---

---

### Overflow for signed numbers (1)

#### Indication of overflow

$$\begin{array}{r} \text{Positive} \\ + \text{Positive} \\ \hline = \text{Negative} \end{array} \qquad \begin{array}{r} \text{Negative} \\ + \text{Negative} \\ \hline = \text{Positive} \end{array}$$

#### Formulas

$$\begin{aligned} \text{Overflow}_{2's \text{ complement}} &= \overline{x_{k-1}} \overline{y_{k-1}} s_{k-1} + x_{k-1} y_{k-1} \overline{s_{k-1}} = \\ &= c_k \oplus c_{k-1} \end{aligned}$$

---

---

---

---

---

---

---

---

### Overflow for signed numbers (2)

$x_{k-1}$	$y_{k-1}$	$c_{k-1}$	$c_k$	$s_{k-1}$	overflow	$c_k \oplus c_{k-1}$
0	0	0	0	0	0	0
<b>0</b>	<b>0</b>	1	0	<b>1</b>	1	1
0	1	0	0	1	0	0
0	1	1	1	0	0	0
1	0	0	0	1	0	0
1	0	1	1	0	0	0
<b>1</b>	<b>1</b>	0	1	<b>0</b>	1	1
1	1	1	1	1	0	0

---

---

---

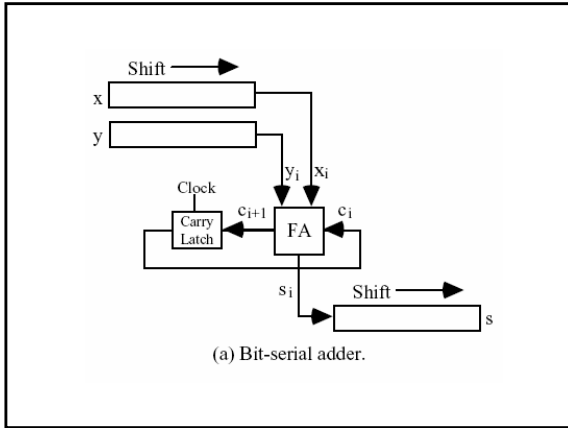
---

---

---

---

---




---

---

---

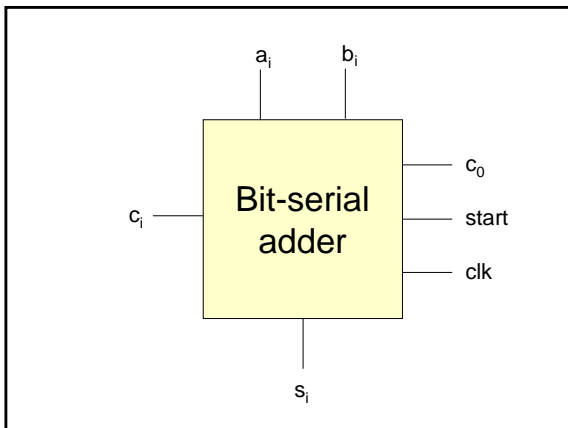
---

---

---

---

---




---

---

---

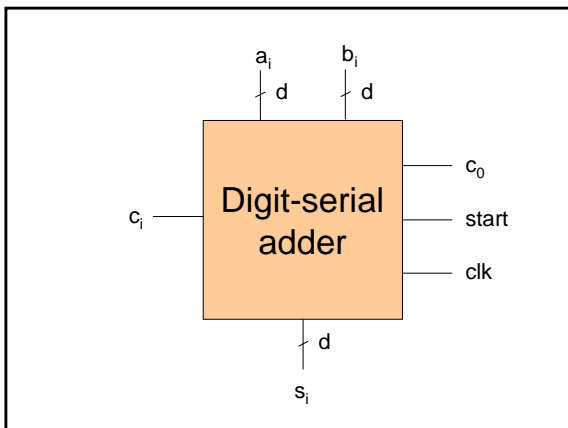
---

---

---

---

---




---

---

---

---

---

---

---

---

### Addition of a constant (1)

$$\begin{array}{r}
 \begin{array}{cccc}
 X_{k-1} & X_{k-2} & \dots & X_1 & X_0 & \text{variable} \\
 + & Y_{k-1} & Y_{k-2} & \dots & Y_1 & Y_0 & \text{constant} \\
 \hline
 S_{k-1} & S_{k-2} & \dots & S_1 & S_0 & 
 \end{array} \\
 \\
 \begin{array}{r}
 X_{k-1} & X_{k-2} & \dots & X_{h+1} & X_h & X_{h-1} & \dots & X_0 & \text{variable} \\
 + & Y_{k-1} & Y_{k-2} & \dots & Y_{h+1} & 1 & 0 & \dots & 0 & \text{constant} \\
 \hline
 S_{k-1} & S_{k-2} & \dots & S_{h+1} & \overline{X_h} & X_{h-1} & \dots & X_0 & 
 \end{array}
 \end{array}$$

---

---

---

---

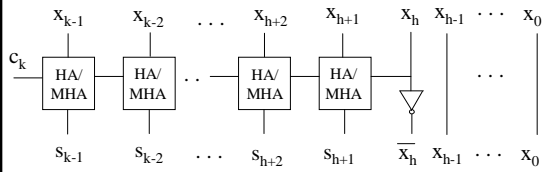
---

---

---

---

### Addition of a constant (2)



If  
 $y_i = 0$  Half-adder (HA)  
 $y_i = 1$  Modified half-adder (MHA)

---

---

---

---

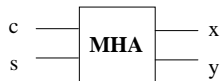
---

---

---

---

### Modified half-adder



$$x + y + 1 = (c \ s)_2$$

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

---

---

---

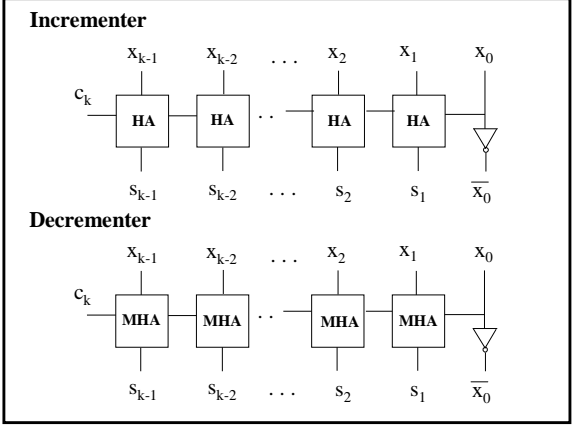
---

---

---

---

---




---

---

---

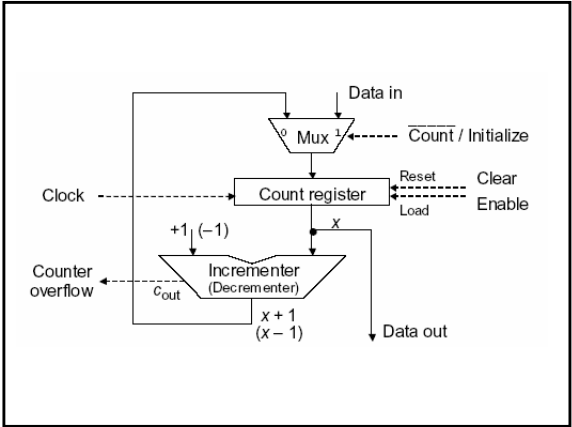
---

---

---

---

---




---

---

---

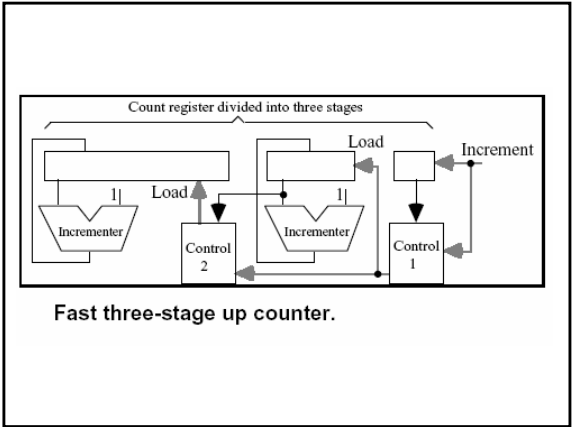
---

---

---

---

---




---

---

---

---

---

---

---

---

### Possible solutions to the carry propagate problem

1. Detect the end of propagation rather than wait for the worst-case time
2. Speed-up propagation via
  - look-ahead
  - carry skip
  - carry select, etc
3. Limit carry propagation to within a small number of bits
4. Eliminate carry propagation through the redundant number representation

---

---

---

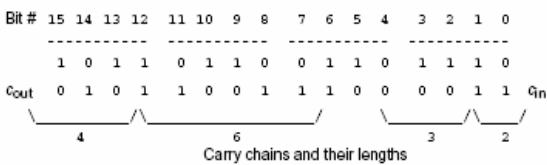
---

---

---

---

---




---

---

---

---

---

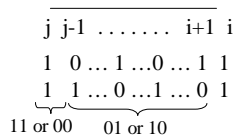
---

---

---

### Analysis of carry propagation

- Probability of carry generation =  $\frac{1}{4}$  ( $x_i y_i = 11$ )
- Probability of carry propagation =  $\frac{1}{2}$  ( $x_i y_i = 01$  or  $10$ )
- Probability of carry annihilation =  $\frac{1}{2}$  ( $x_i y_i = 00$  or  $11$ )



Probability of carry propagating from position i to position j =  $\frac{1^{j-i-1}}{2} \cdot \frac{1}{2} = \frac{1^{j-i}}{2}$

probability of propagation
probability of annihilation

---

---

---

---

---

---

---

---

**Expected length of the carry chain that starts at position  $i$  (1)**

Expected length( $i, k$ ) =

$$\sum_{j=i+1}^{k-1} (j-i) \left(\frac{1}{2}\right)^{j-i} + (k-i) \left(\frac{1}{2}\right)^{k-1-i}$$

Length of the carry chain      Probability of the given length      Distance till the end of adder      Probability of propagation till the end of adder

---

---

---

---

---

---

---

---

**Expected length of the carry chain that starts at position  $i$  (2)**

Expected length( $i, k$ ) =

$$2 - 2^{-(k-i-1)}$$

For  $i \ll k$

Expected length of the carry propagation is  $\approx 2$

---

---

---

---

---

---

---

---