01.03.21 12:50

CSCI 150
Introduction to Digital and Computer System Design
Lecture 3: Combinational Logic Design VII

Jetic Gū

Overview

- Focus: Arithmetic Functional Blocks
- Architecture: Combinatory Logical Circuits
- Textbook v4: Ch4 4.3, 4.4, 4.7; v5: Ch2 2.9, Ch3 3.10, 3.11
- Core Ideas:
    1. Subtraction II
    2. Subtraction III
    3. VHDL

P0
Review

Unsigned Binary Subtraction I

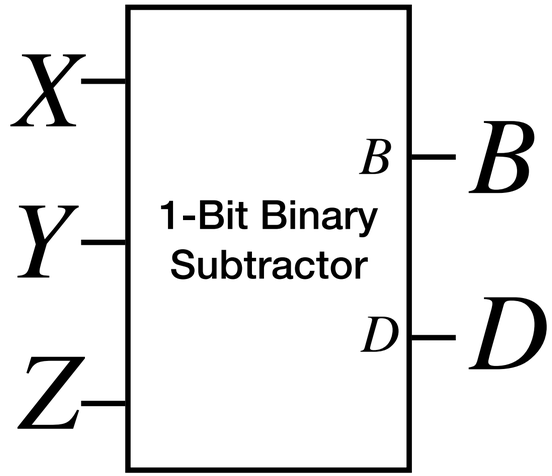
Review

Summary

P1
Subtraction

Unsigned 1-bit Binary Subtraction

- Implementation using 3-to-8 Decoder
  - $B = \Sigma m(1,2,3,7)$
  - $D = \Sigma m(1,2,4,7)$

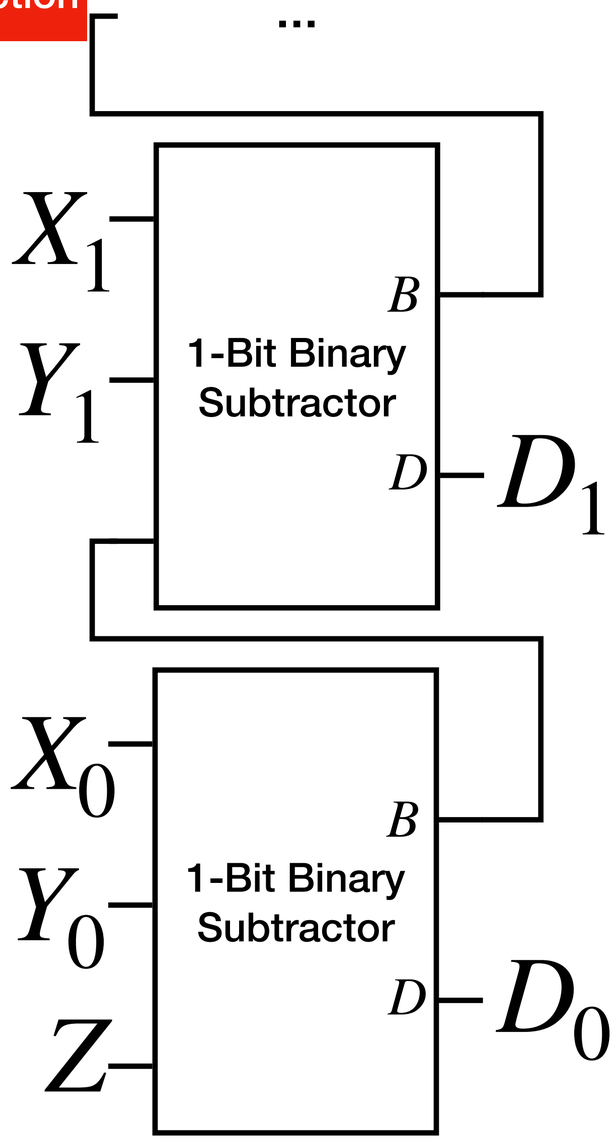


Review

P1
Subtraction

Unsigned Binary Subtraction

Technology
- 1 bit Unsigned Subtractor



		B				Z
Borrows		0	0	0	1	1 0
Minuend $X_{0:n-1}$		1	0	1	1	0
Subtrahend $Y_{0:n-1}$	−	1	0	0	1	1
Difference $D_{0:n-1}$		0	0	0	1	1

Input
Output

Review

Unsigned Binary Subtraction II

$X - Y$ when $Y > X$

Summary

What we have so far

- Binary Adder
- Binary Subtractor ($X - Y, X \geq Y$)

Review

P1
Subtraction

Unsigned Binary Subtraction

$$X > Y, F = X - Y$$

- We learned to perform subtraction, by subtracting the smaller number from the greater number

- What if it's the opposite? i.e. $X < Y, F = X - Y$?

Concept

P1
Subtraction

Unsigned Binary Subtraction

Borrows		
Minuend	10011	
Subtrahend	−00110	
Difference		

Borrows		
Minuend	00110	
Subtrahend	−10011	
Difference		

Concept

P1
Subtraction

Unsigned Binary Subtraction

Borrows	011000
Minuend	10011
Subtrahend	−00110
Difference	01101

This is correct

Borrows	100110
Minuend	00110
Subtrahend	−10011
Difference	10011

This is incorrect

Concept

P1
Subtraction

Unsigned Binary Subtraction

- Standard subtraction module works if the Minuend is bigger than the Subtrahend

Borrows	011000
Minuend	10011
Subtrahend	−00110
Difference	01101

This is correct

Borrows	100110
Minuend	00110
Subtrahend	−10011
Difference	10011

This is incorrect

Concept

P1
Subtraction

Unsigned Binary Subtraction

- Standard subtraction module works if the Minuend is bigger than the Subtrahend

- Incorrect output D
  $= 2^n + X - Y$

- Correct output D'
  $= -(Y - X)$
  $= -(2^n - D)$
  $= -(\overline{D} + 1)$ **(2's compliment)**

Borrows	$1\,00110$
Minuend X	$\,00110$
Subtrahend Y	$-\,10011$
Difference D	$\,10011$
Corrected D'	$-\,01101$

This is incorrect

Concept

2s compliment

- Given binary unsigned integer of n bits D, its 2s compliment
  $2^n - D = \overline{D} + 1$
- Proof
  - Biggest number represented in n bit: $(11...1)_2 = 2^n - 1$
  - $2^n - D = [(11...1)_2 + 1] - D = (11...1)_2 - D + 1 = \overline{D} + 1$
- Implementation
  - Inversion and plus 1, easily doable as complementer

Concept

Subtraction

1. Compute 20-15=5 using 6-bit binary
   - $20 = (010100)_2, 15 = (001111)_2$
2. Compute 15-20=-5 using 6-bit binary and 2s complement

Borrows 0011110

$$\begin{array}{r} 010100 \\ -001111 \\ \hline 000101 \end{array}$$

Example

Subtraction

1. Compute 20-15=5 using 6-bit binary
   - $20 = (010100)_2, 15 = (001111)_2$
2. Compute 15-20=-5 using 6-bit binary and 2s complement
   - Correction: $(111011)_2$ 2s complement: $(000101)_2$

Borrows $1\ 100000$

$$\begin{array}{r} 001111 \\ -010100 \\ \hline 111011 \\ 000101 \end{array}$$

Example

Subtraction

1. Compute 20-15=5 using 6-bit binary
   - $20 = (010100)_2, 15 = (001111)_2$
2. Compute 15-20=-5 using 6-bit binary and 2s complement
   -

Borrows $1\ 100000$

$$\begin{array}{r} 001111 \\ -010100 \\ \hline 111011 \end{array}$$

Example

Subtraction

1. Compute 7-15=-8 using 6-bit binary and 2s complement

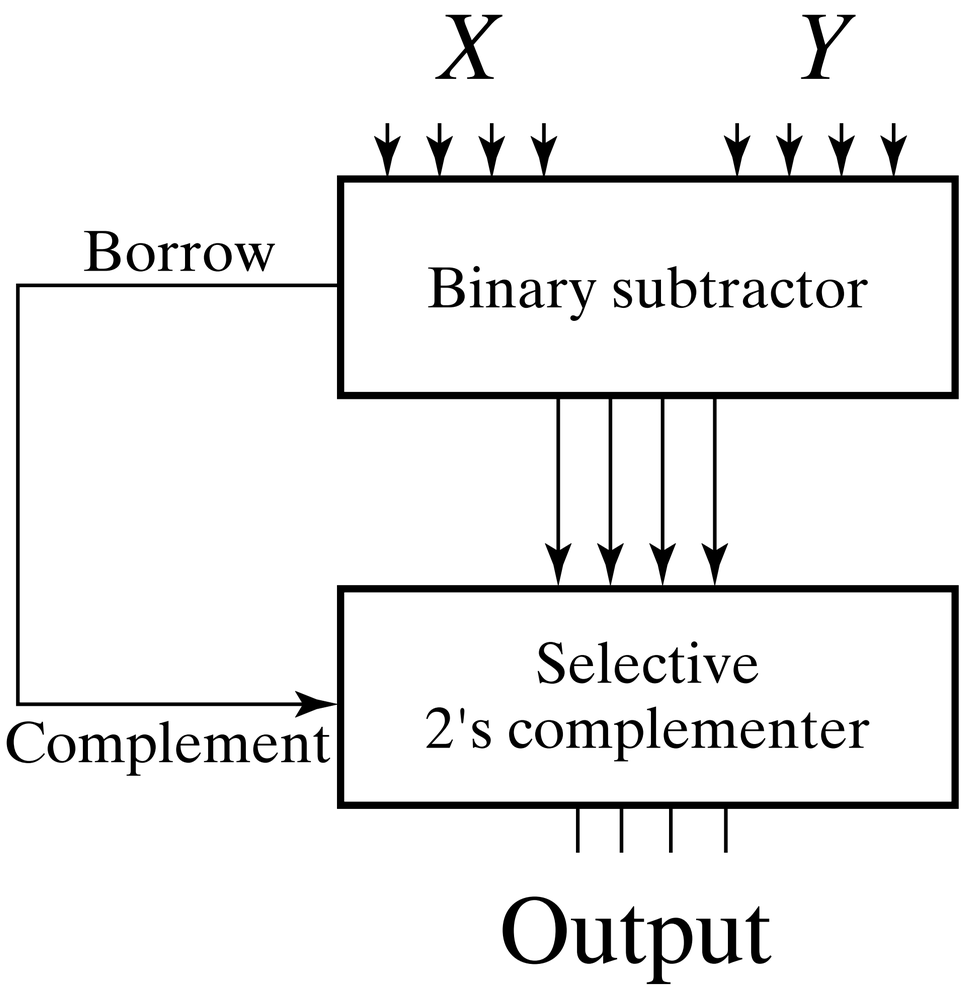
Example

Full Unsigned Subtraction

- Solution 1
  - Compare the Minuend and Subtrahend, switch places if the Subtrahend is greater, then add negative sign
- Solution 2
  - Use 2s compliment

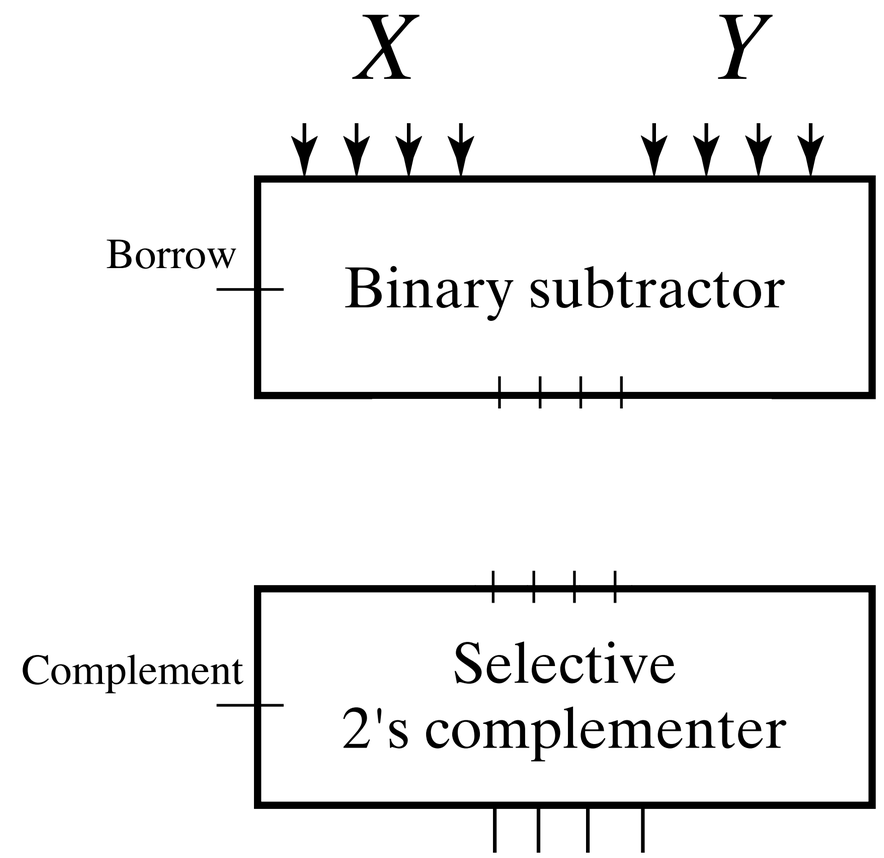
Concept

Full Unsigned Subtraction



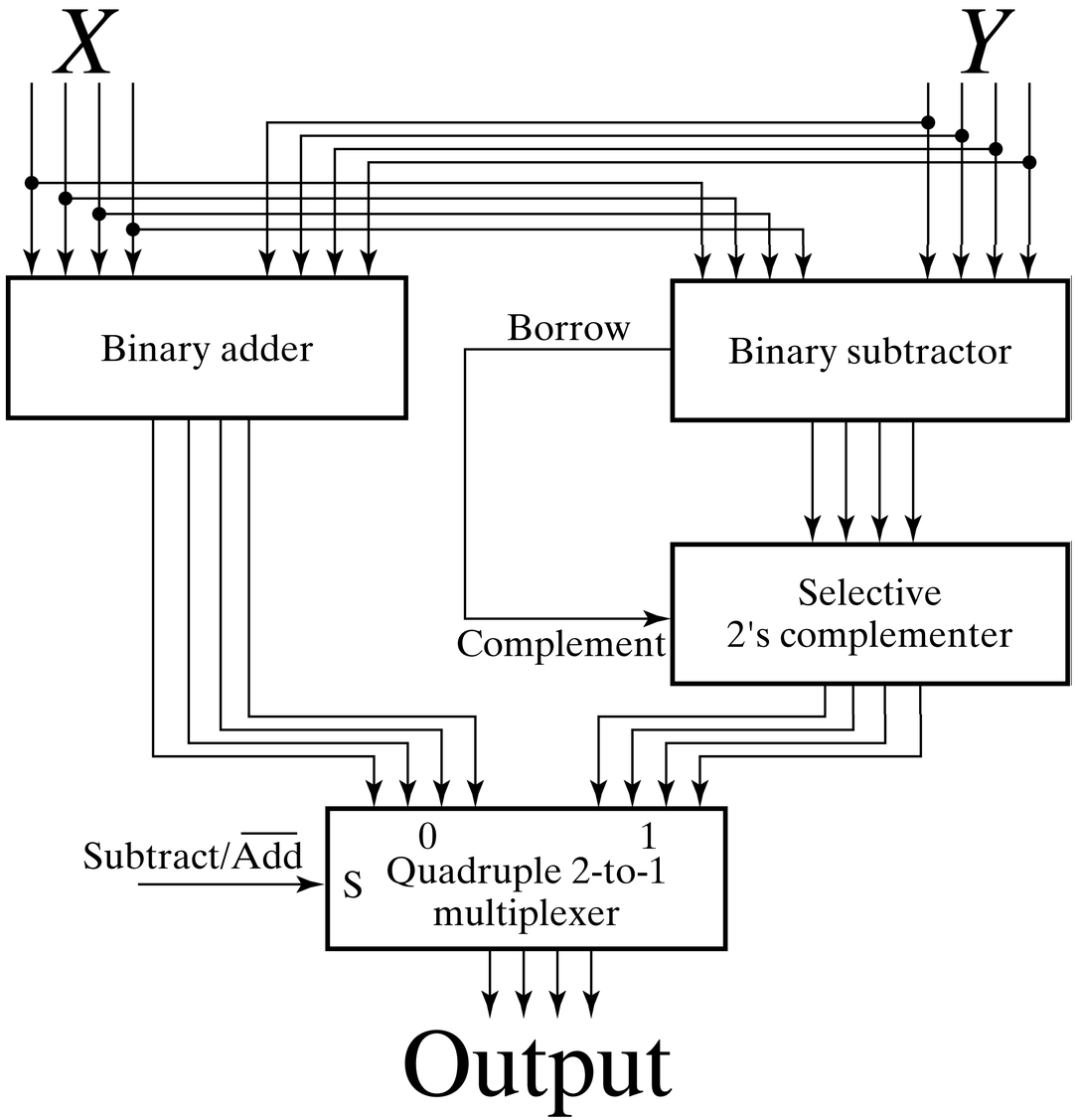
Concept

Full Unsigned Subtraction



Concept

Adder-Subtractor



Concept

Unsigned Binary Subtraction III

What do you mean we can do subtraction using an adder?

Summary

2s compliment

$$2^n - D = \overline{D} + 1$$

Subtraction using 2s Complement

- Input: X and Y
  - Y: 2s complement $Y' = \overline{Y} + 1$ ($\overline{Y} + 1 = 2^n - Y$)
  - $X - Y = X + (2^n - Y) - 2^n = X + Y' - 2^n$
  - Since $2^n = (10...0)_2$, and we only output n-bits, it can be discarded

Concept

Subtraction using 2s Complement

- 84 - 67 (10bit)

$$X = 0001010100$$

$$Y = 0001000011$$

2s complement $$Y' = 1110111101$$

$$X + Y' = 10000010001$$

discard carry $$X + Y' - 2^{10} = 0000010001$$ **correct!**

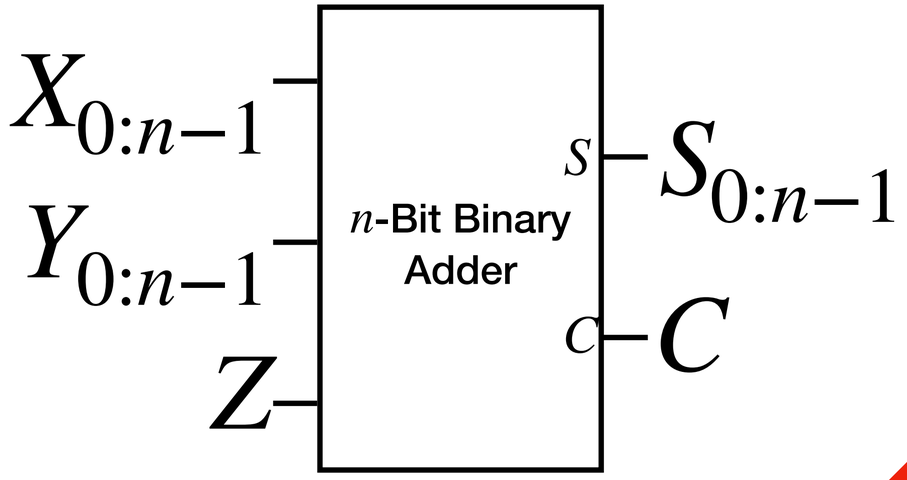
verify $$(84 - 67)_{10} = 17_{10} = 10001 = 0000010001$$

Example

Adder-Subtractor Unit

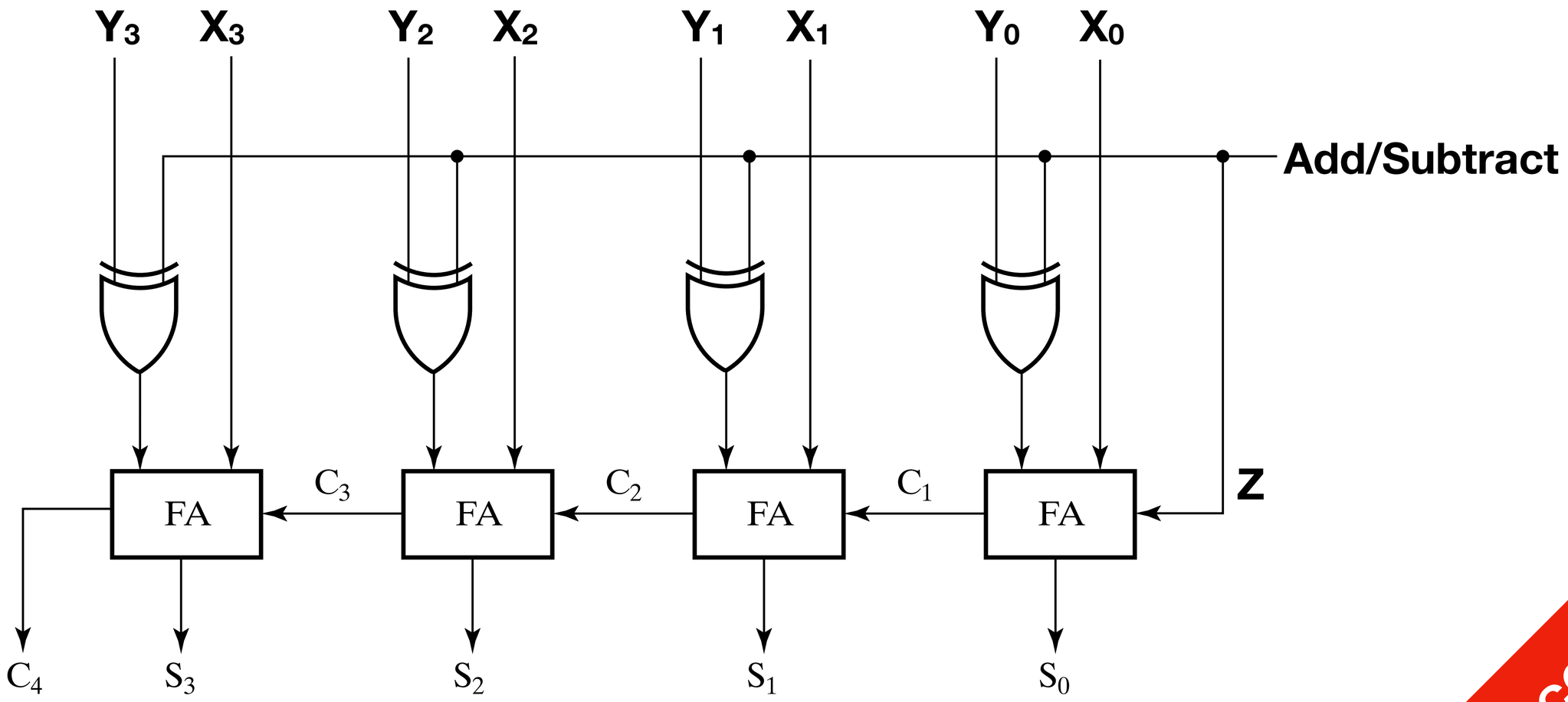
1. Adder
2. Complementer (Inverting and add 1)
    - Or just inverting, and then plus one

- Addition: $X + Y$
- Subtraction: $X - Y = X + \overline{Y} + 1 - 2^n$
    - We are using n-bit adder, 2^n can be disregarded
    - The plus 1 here can be Z input to the adder



Concept

Adder-Subtractor Unit



Concept

P0-2
Adder-Subtractor

Adder Subtractor Units (Unsigned)

- Binary Adder
- Binary Subtractor
- Binary Adder-Subtractor Unit, using Adder, Subtractor, Complementer and Multiplexer
- Binary Adder-Subtractor Unit using Adder and XOR

Review

Excercises

2s Complement
Subtraction using 2s Complement

Summary

Subtraction

- Compute 10-7 using 4-bit binary using Adder and 2s complement

Exercise

Subtraction

- Compute 24-17 using 8-bit binary using Adder and 2s complement

Exercise

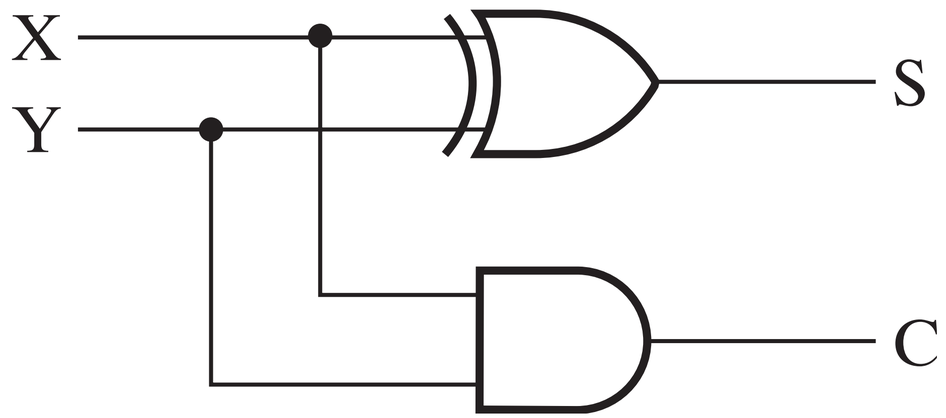
Hardware Description Language

VHDL (VHSIC-HDL): Very High Speed Integrated Circuit Hardware Description Language

Summary

Previous: 1-bit Half Adder

- Create a new component in VHDL called `HalfAdder1`
  - Input: X, Y
  - Output: S, C
  - Don’t use `AFTER`



Practice

Previous: 1-bit Half Adder

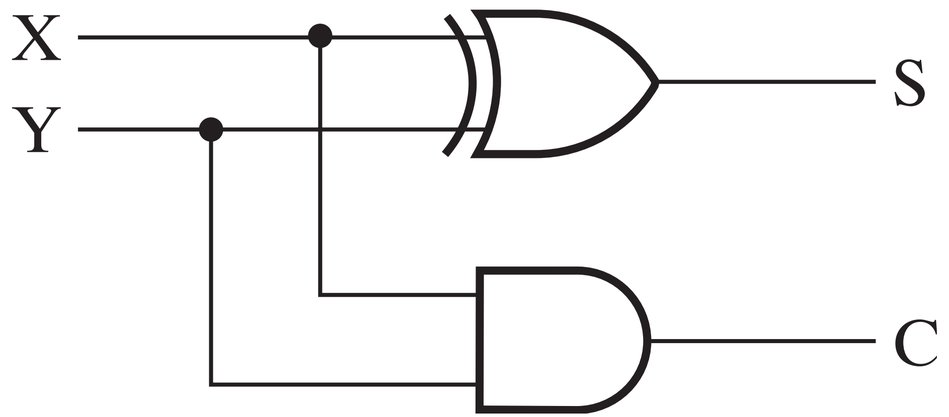
```
architecture arch1 of HalfAdder is

begin

     S <= X XOR Y;

     C <= X AND Y;

end arch1;
```



Practice

P4
VHDL

Today’s Tasks

- 1-bit Half Adder
- 1-bit Full Adder using Schema Diagram (Logic Circuit Diagram)
- 4-bit Full Adder using Schema Diagram (Logic Circuit Diagram)
- 4-bit Adder-Subtractor using Schema Diagram (Logic Circuit Diagram)

Exercise