Digital Electronics and Microprocessors

Class 19

CHHAYADEVI BHAMARE

Digital Arithmetic: Operations and Circuits(Chapter 6)

Arithmetic operations

- Binary addition
- Representing Signed Numbers
- Addition and subtractionin the 2’s complement systems.
- Multiplication and division of binary numbers
- BCD addition
- Hexadecimal Arithmetic

Digital Arithmetic: Operations and Circuits(Chapter 6)

Arithmetic Circuits

- Design of a full adder
- Parallel binary adder
- Complete parallel adder with registers
- Carry propogation
- IC arithmetic circuits (parallele adder, cascading of parallel adder, ALU IC etc)

Special Case in 2's-Complement Representation

Whenever a signed number has a 1 in the sign bit and all 0s for the magnitude bits, its decimal equivalent is -2^N, where N is the number of bits in the *magnitude*. For example,

$$1000 = -2^3 = -8$$
$$10000 = -2^4 = -16$$
$$100000 = -2^5 = -32$$

Thus, we can state that the complete range of values that can be represented in the 2's-complement system having N magnitude bits is

$$-2^N \text{ to } +(2^N - 1)$$

There are a total of 2^{N+1} different values, *including* zero.

Signed binary numbers: Possible representations

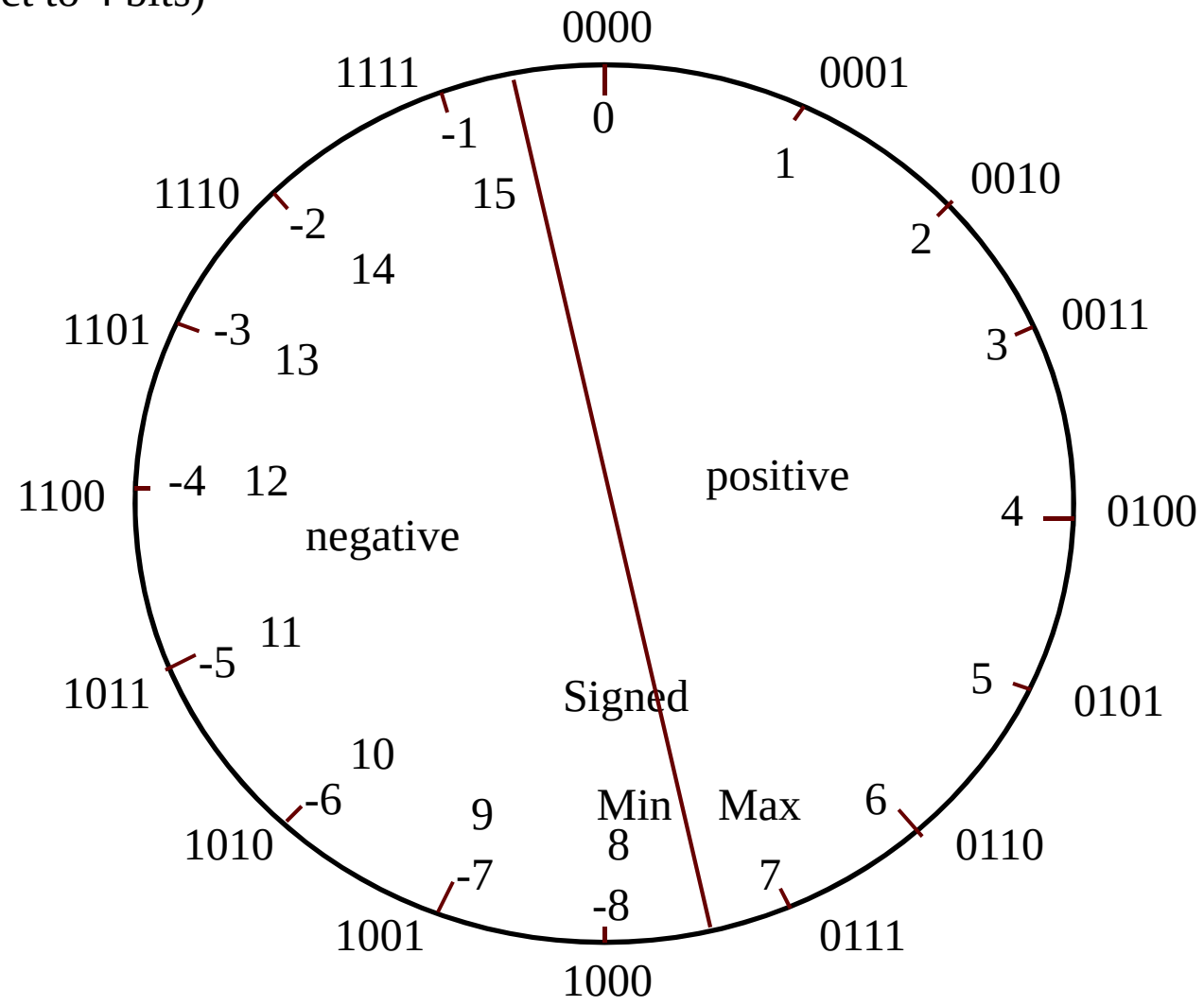
- Sign Magnitude: One's Complement Two's Complement

Sign Magnitude	One's Complement	Two's Complement
000 = +0	000 = +0	000 = +0
001 = +1	001 = +1	001 = +1
010 = +2	010 = +2	010 = +2
011 = +3	011 = +3	011 = +3
100 = -0	100 = -3	100 = -4
101 = -1	101 = -2	101 = -3
110 = -2	110 = -1	110 = -2
111 = -3	111 = -0	111 = -1

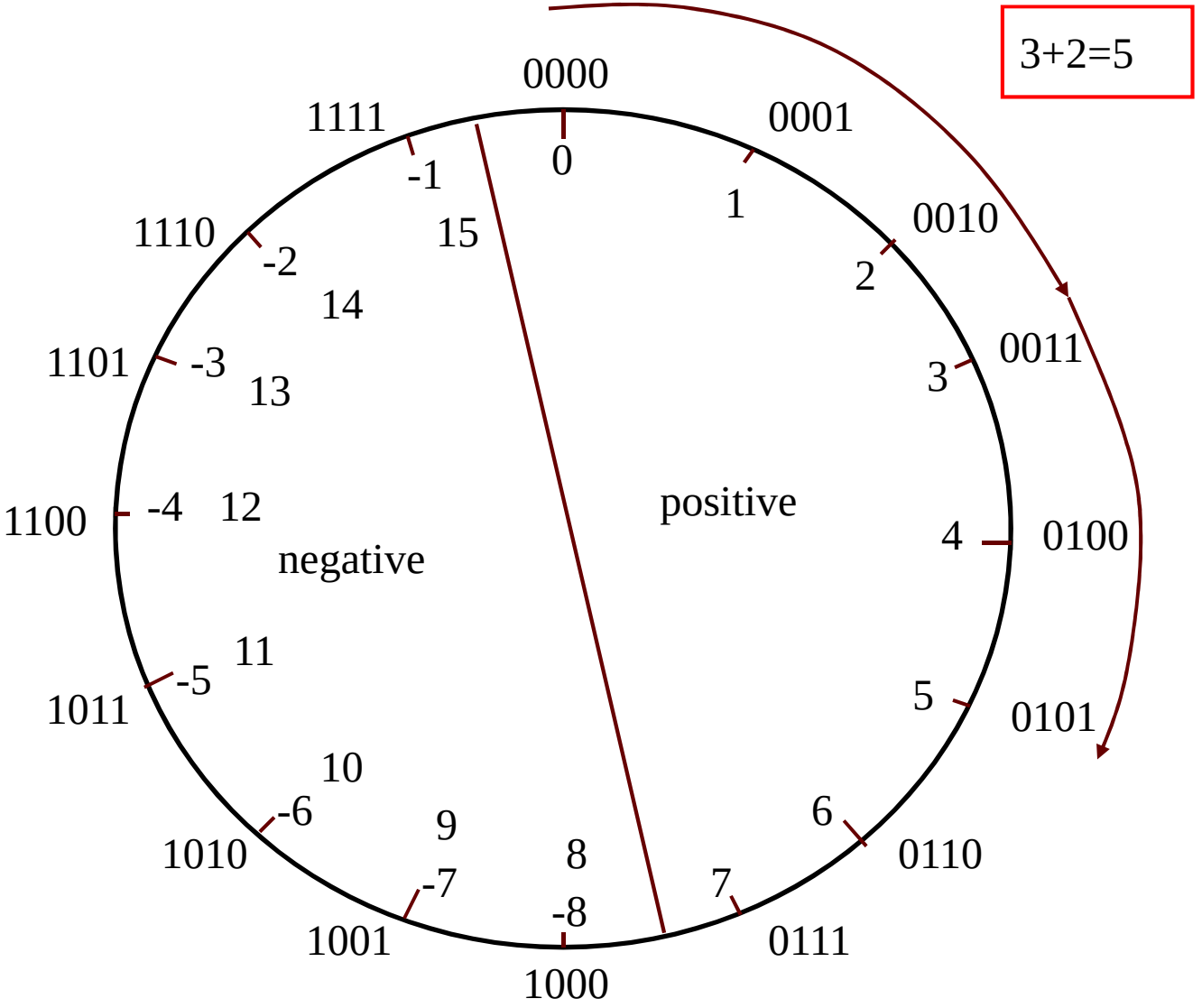
- Issues: balance, number of zeros, ease of operations
- Which one is best? Why?
- Pick the representation that made the hardware simple

Two’s complement

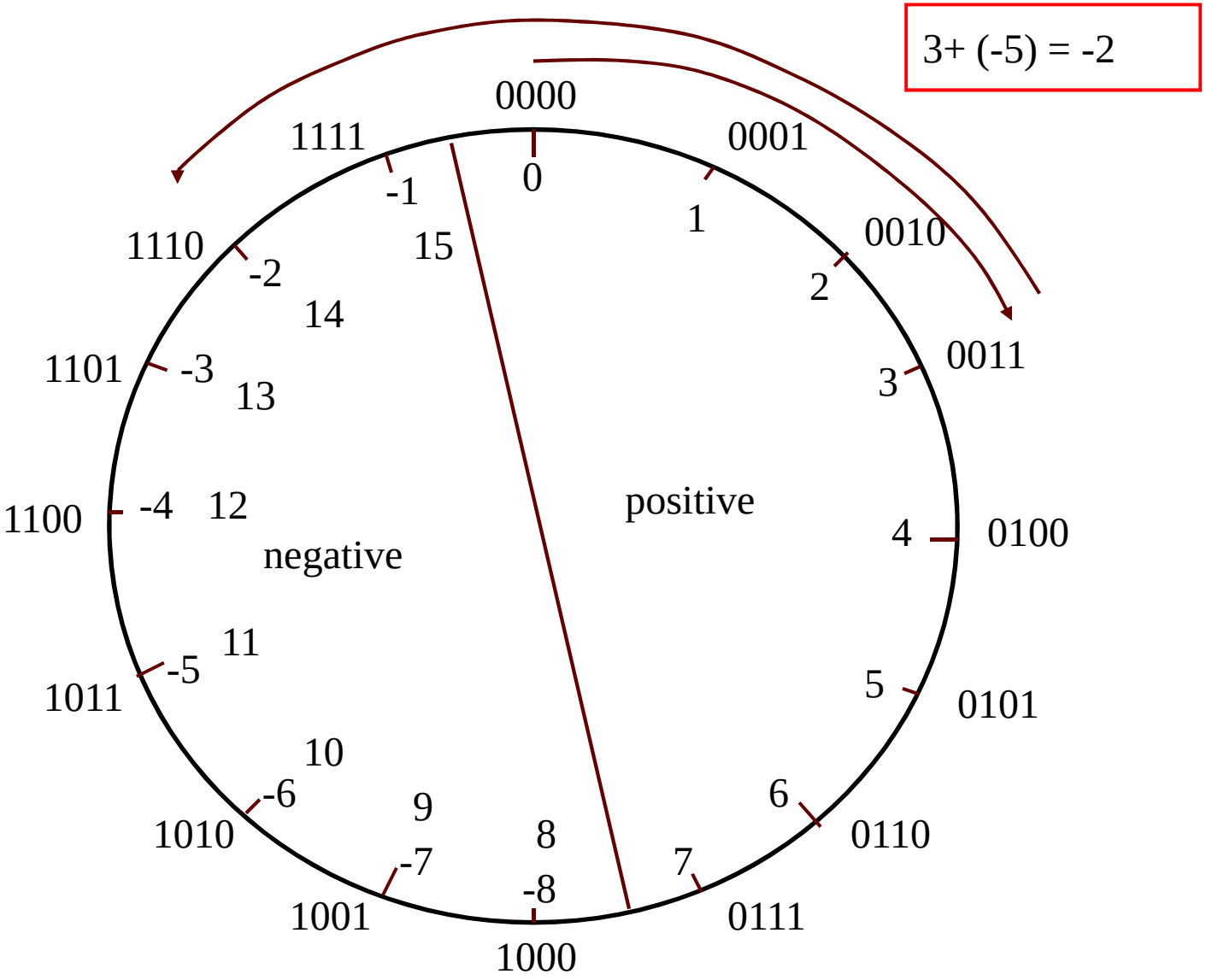
(let’s restrict to 4 bits)



Two's complement

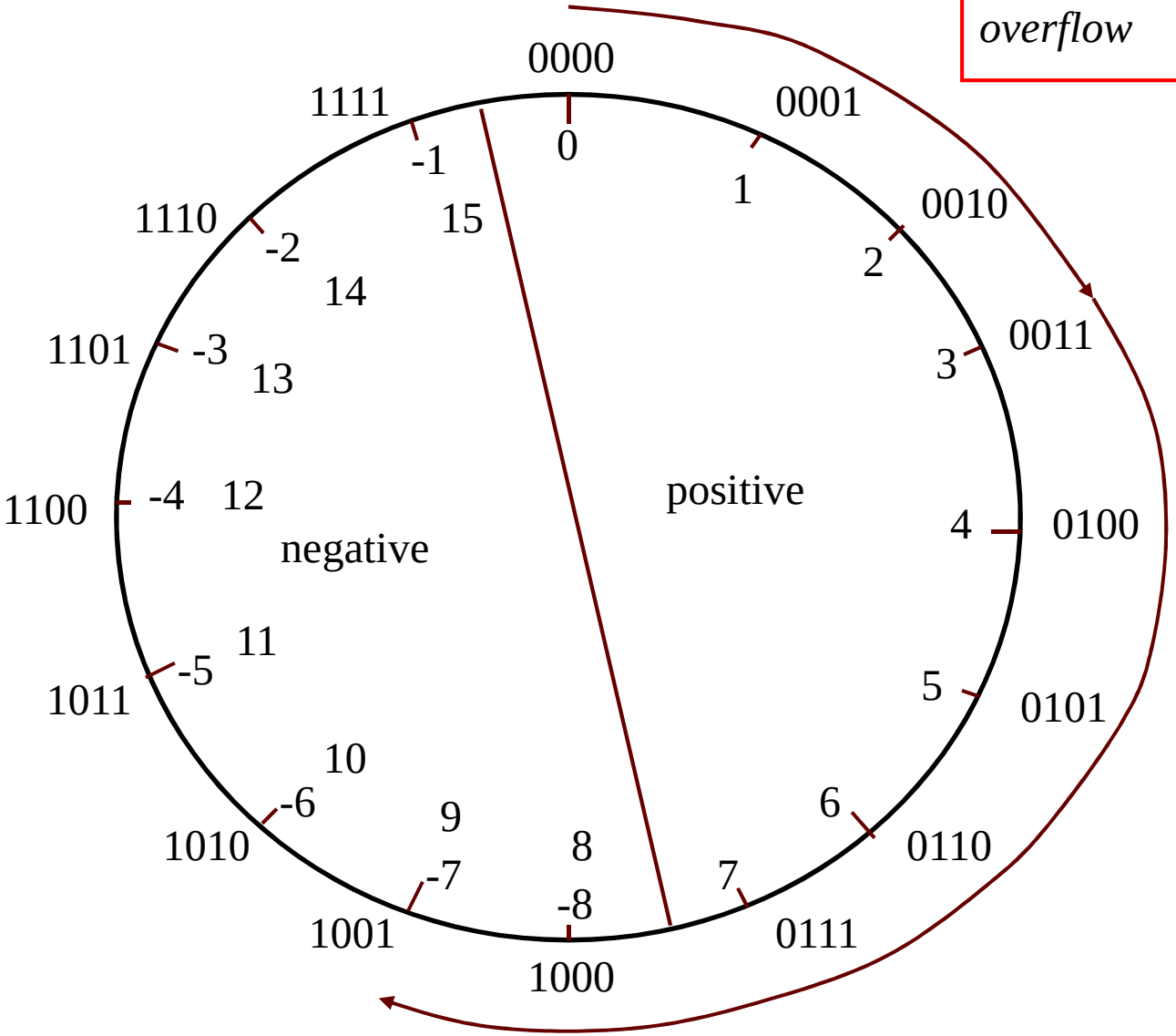


Two's complement



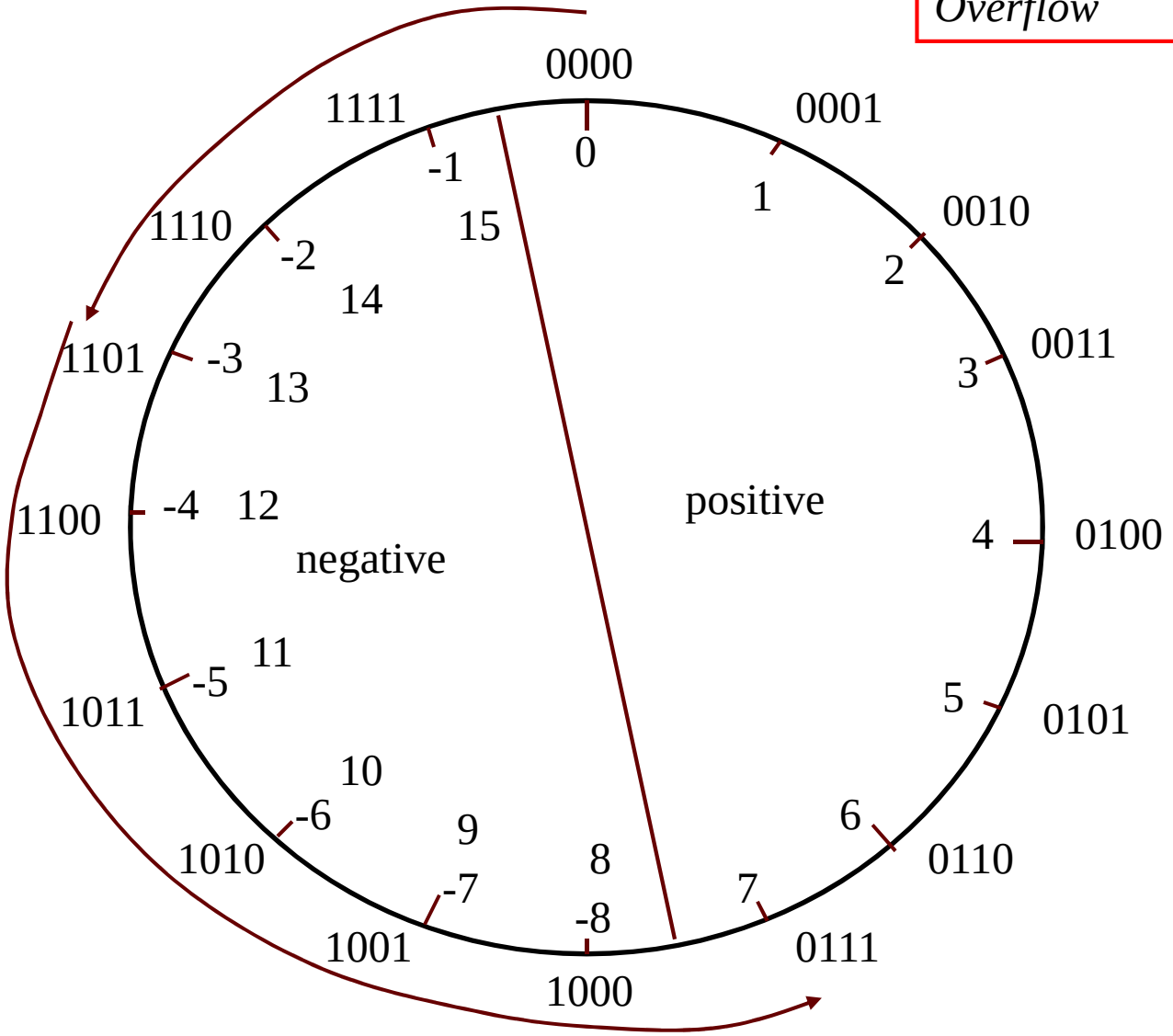
Two's complement

3+6 = -7 !!
overflow



Two's complement

-3 + (-6) = 7 !!
Overflow



BCD Addition

- When the sum of each decimal digit is less than 9, the operation is the same as normal binary addition.
- When the sum of each decimal digit is greater than 9, a binary 6 is added. This will always cause a carry.
- BCD subtraction – a more complicated operation – is out of scope for this course

Sum Equals 9 or Less

5	0101	← BCD for 5
+4	+0100	← BCD for 4
9	1001	← BCD for 9

45	0100 0101	← BCD for 45
+33	+0011 0011	← BCD for 33
78	0111 1000	← BCD for 78

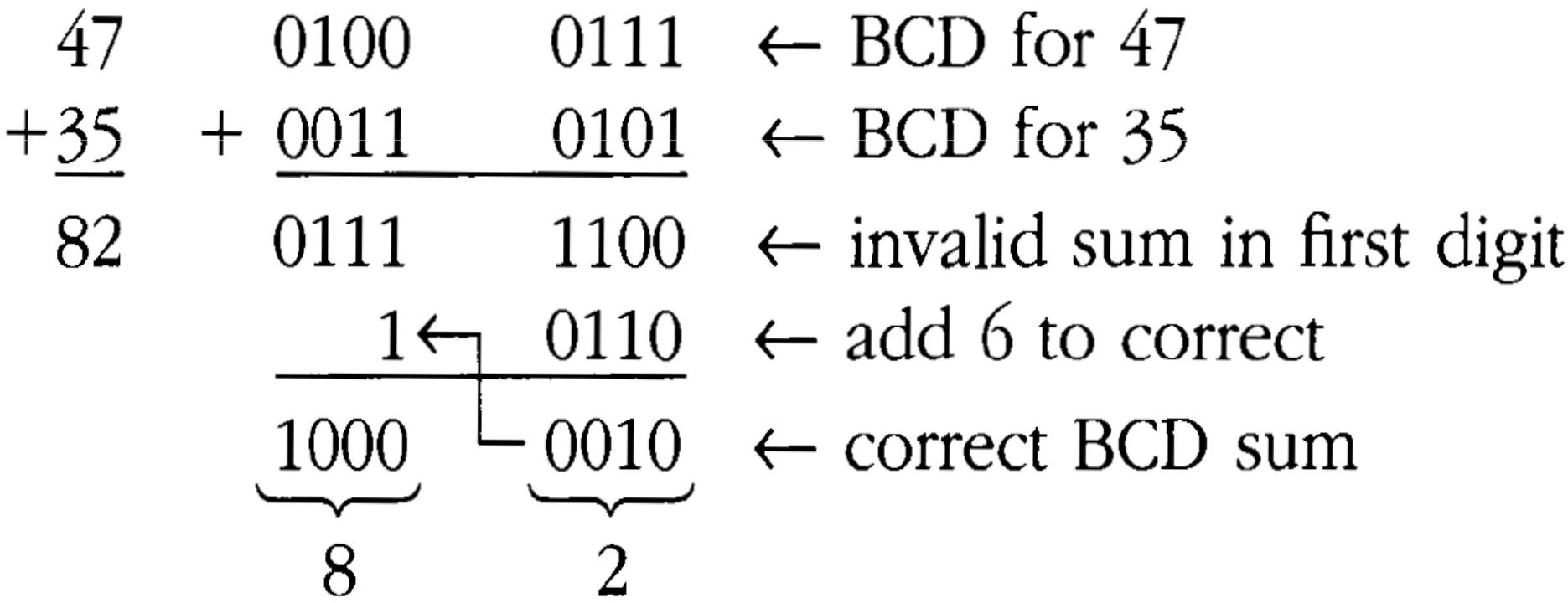
Sum greater than 9

6	0110	← BCD for 6
+7	+ 0111	← BCD for 7
+13	1101	← invalid code group for BCD

Sum greater than 9

	0110	← BCD for 6
+	0111	← BCD for 7
	1101	← invalid sum
	0110	← add 6 for correction
0001	0011	← BCD for 13
1	3	

Sum greater than 9



Perform this BCD addition

$$\begin{array}{r} 59 \\ +\underline{38} \\ 97 \end{array}$$

BCD addition procedure

1. Using ordinary binary addition, add the BCD code groups for each digit position.
2. For those positions where the sum is 9 or less, no correction is needed. The sum is in proper BCD form.
3. When the sum of two digits is greater than 9, a correction of 0110 should be added to that sum to get the proper BCD result. This case always produces a carry into the next digit position, either from the original addition (step 1) or from the correction addition.

Hexadecimal Arithmetic

- Hex Addition
  - Add the two hex digits in decimal, mentally inserting the decimal equivalent for those digits larger than 9.
  - If the sum is 15 or less, it can be directly expressed as a hex digit.
  - If the sum is greater than or equal to 16, subtract 16 and carry a 1 to the next digit position.
- Examples
  - Add the hex numbers 58 and 24
  - Add the hex numbers 58 and 4B
  - Add 3AF and 23C

Hex addition Examples

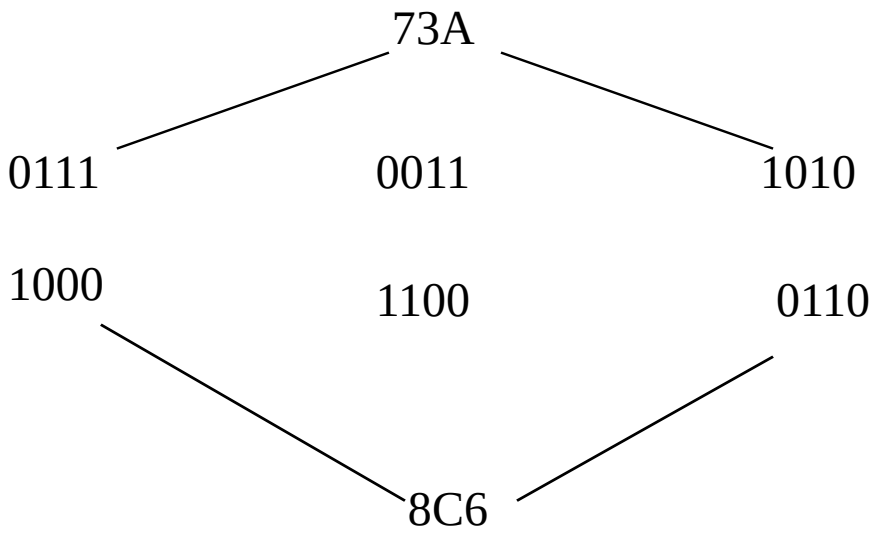
$$\begin{array}{r} 58 \\ +\,24 \\ \hline 7C \end{array} \qquad \begin{array}{r} 58 \\ +\,4B \\ \hline A3 \end{array} \qquad \begin{array}{r} 3AF \\ +\,23C \\ \hline 5EB \end{array}$$

Hex subtraction

- Hex subtraction – use the same method as for binary numbers.

Hex subtraction

- How to get the 2’s-complement of a hex number?
- Convert to binary, take the 2’s complement of the binary equivalent, and then convert back to hex.



- A quicker procedure: subtract each hex digit from F; then add 1.

F	F	F		
-7	-3	-A	}	← Subtract each digit from F
8	C	5		
		+1		← Add 1
8	C	6		← Hex equivalent of 2’s complement

Example

Subtract $3A5_{16}$ from 592_{16}.

Solution

First, convert the subtrahend (3A5) to its 2's-complement form by using either method presented above. The result is C5B. Then add this to the minuend (592):

$$\begin{array}{r} 592 \\ +\ C5B \\ \hline 1ED \end{array}$$

Disregard carry.

Ignoring the carry out of the MSD addition, the result is 1ED. We can prove that this is correct by adding 1ED to 3A5 and checking to see that it equals 592_{16}.

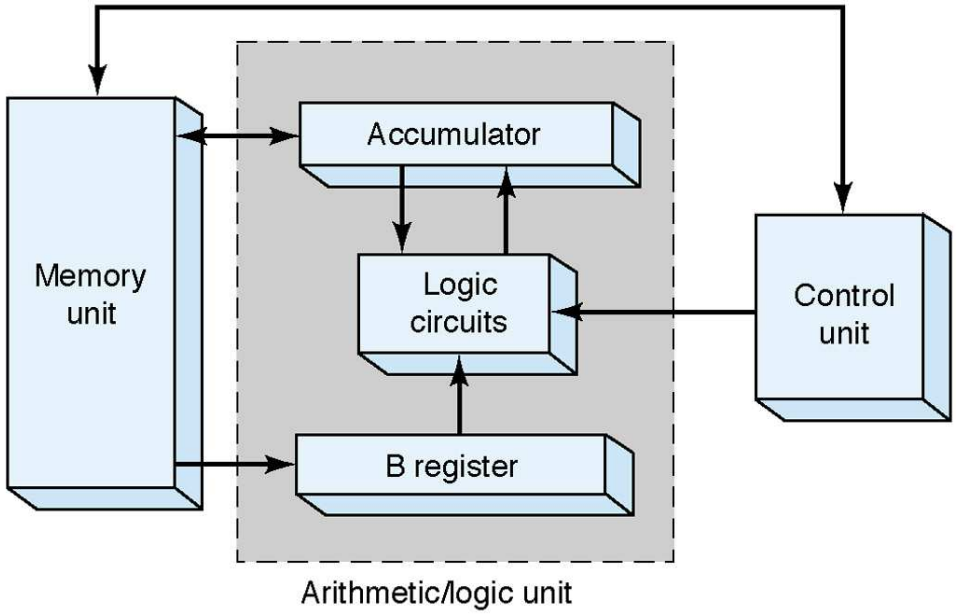
Note:- when the MSD is 8 or greater, the number being represented is negative, when the MSD is 7 or less, the number is positive.

Assignment 3

- 6-1 to 6-17

Arithmetic Circuits

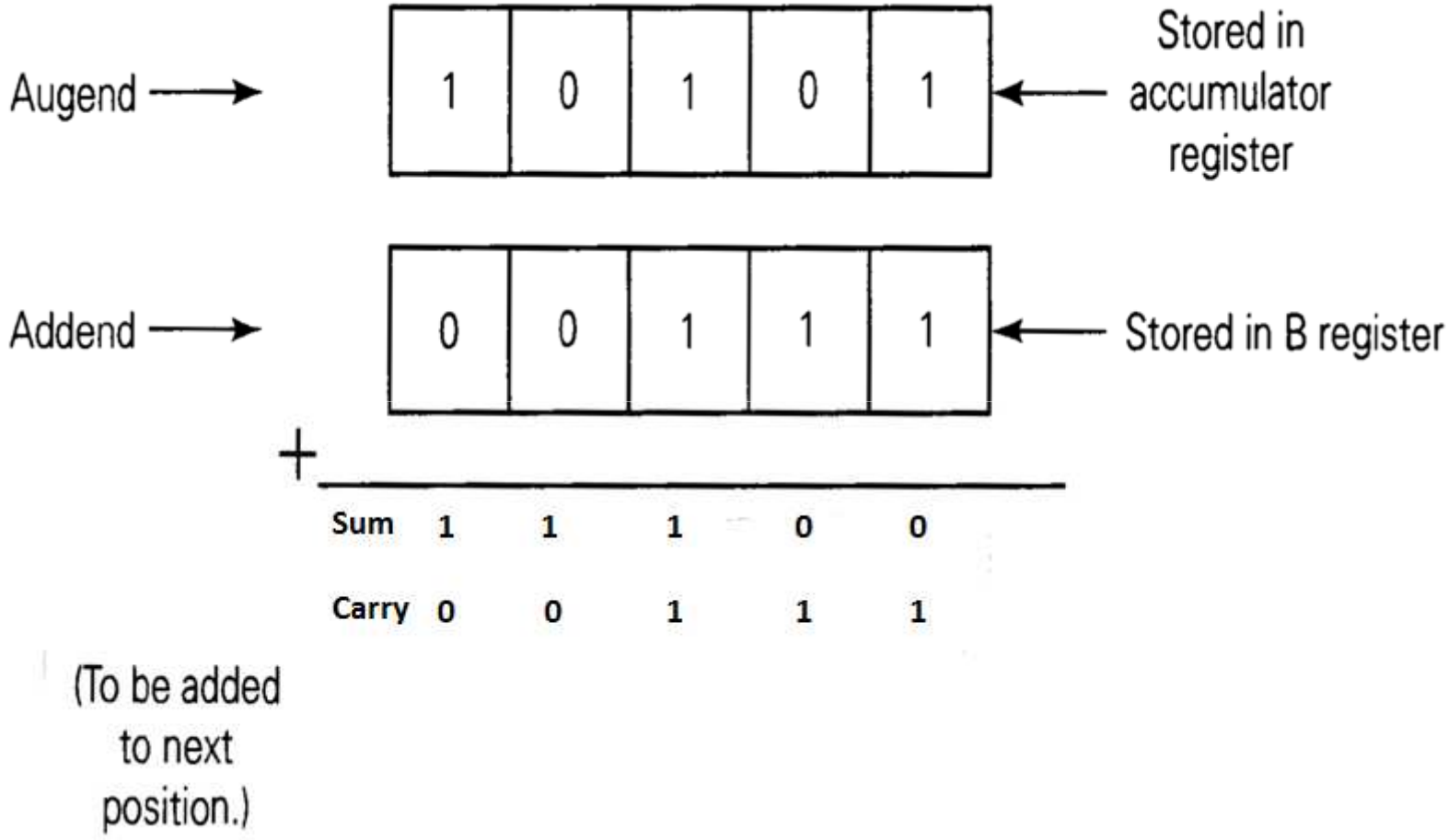
- An arithmetic/logic unit (ALU) accepts data stored in memory and executes arithmetic and logic operations as instructed by the control unit.



Arithmetic Circuits

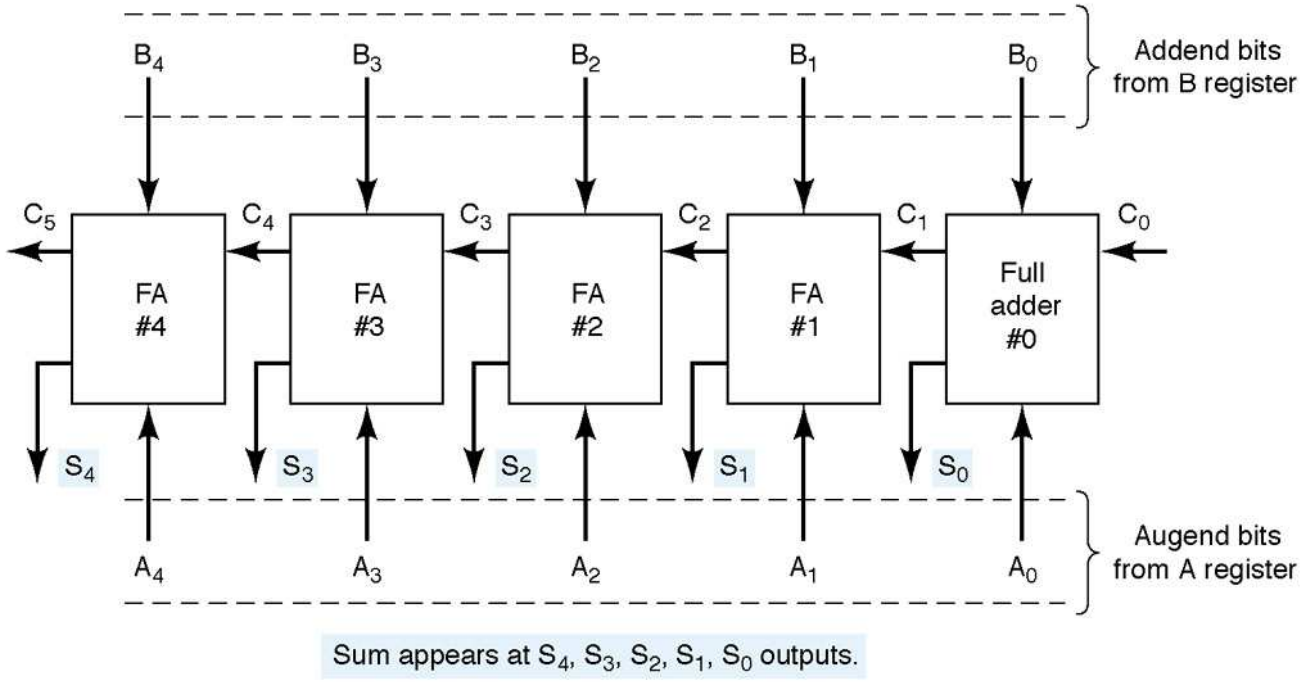
- Typical sequence of operations:
  - Control unit is instructed to add a specific number from a memory location to a number stored in the accumulator register.
  - The number is transferred from memory to the B register.
  - Number in B register and accumulator register are added in the logic circuit, with sum sent to accumulator for storage.
  - The new number remains in the accumulator for further operations or can be transferred to memory for storage.

Parallel Binary Adder



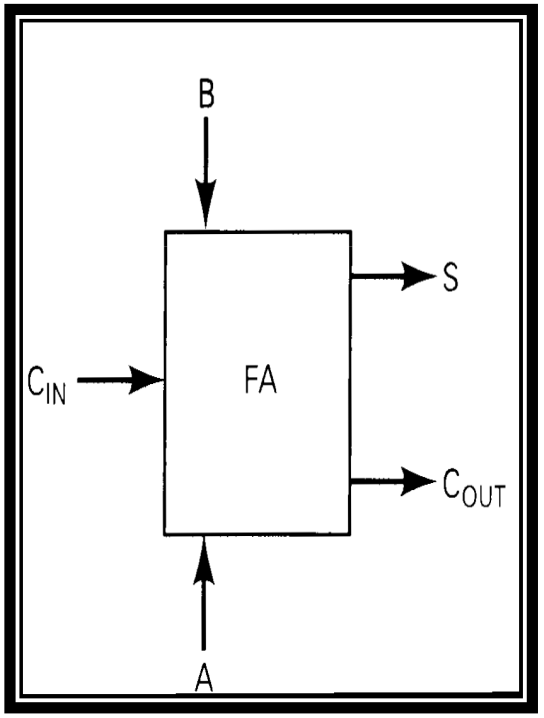
Parallel Binary Adder

- The A and B variables represent 2 binary numbers to be added. The C variables are the carries. The S variables are the sum bits.

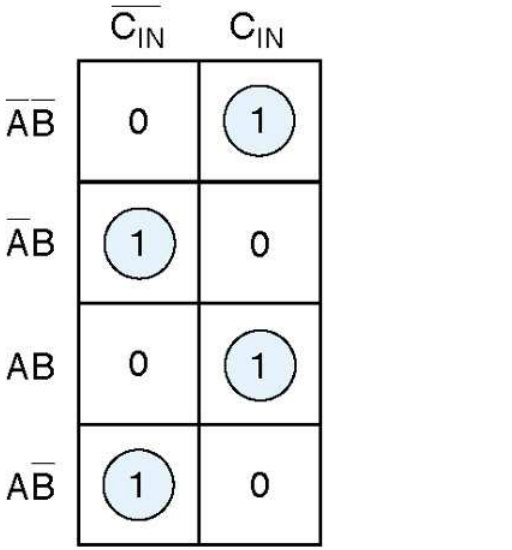


Design of a full addrer

Auend bit input	Adden d bit input	Carry bit input	Sum bit output	Carry bit output
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



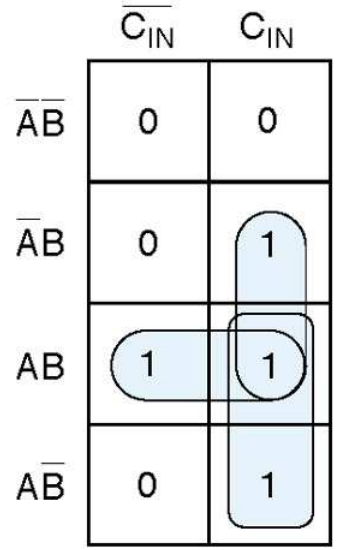
K Mappings for the full-adder outputs



K map for S

$S = \overline{A}\overline{B}C_{IN} + \overline{A}B\overline{C}_{IN} + ABC_{IN} + A\overline{B}\overline{C}_{IN}$

(a)

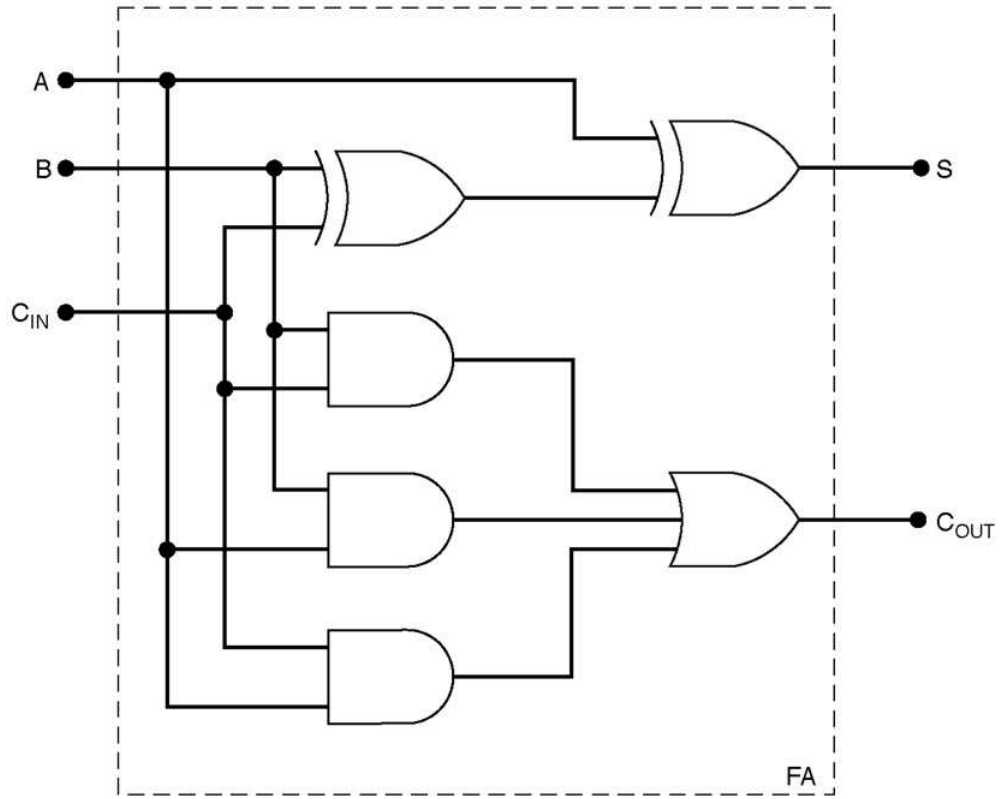


K map for C_{OUT}

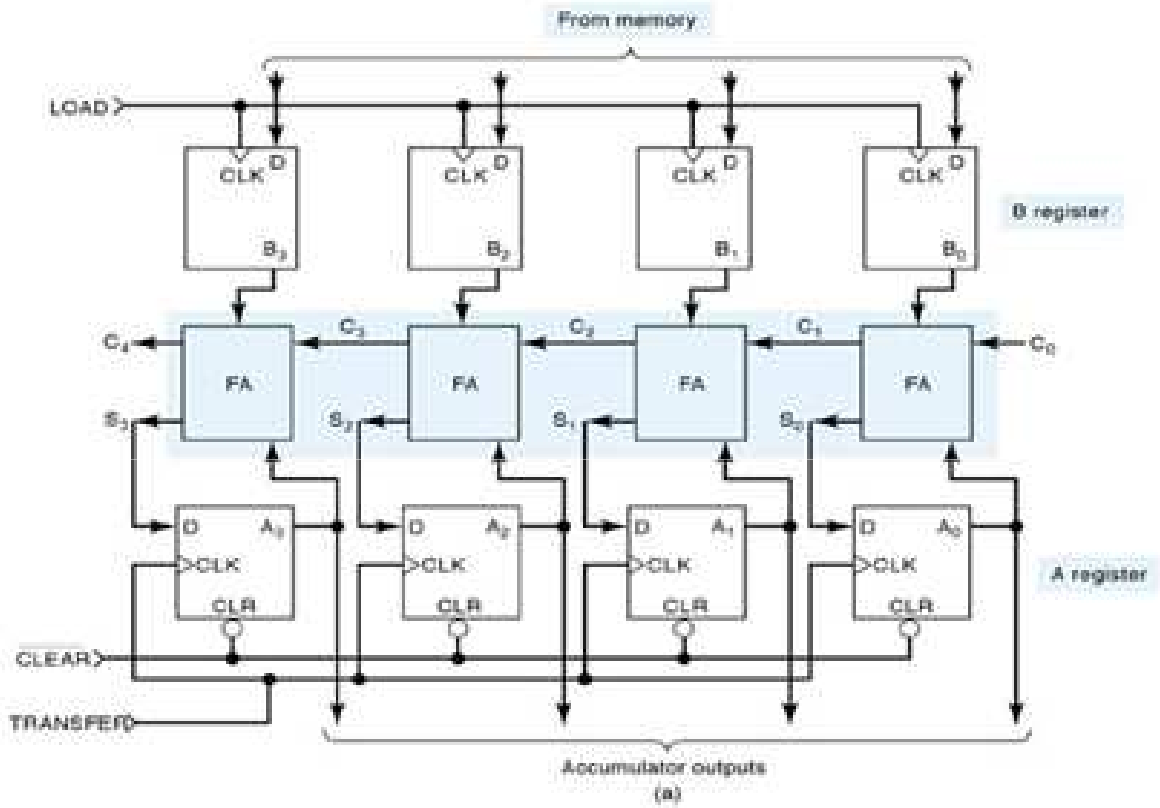
$C_{OUT} = BC_{IN} + AC_{IN} + AB$

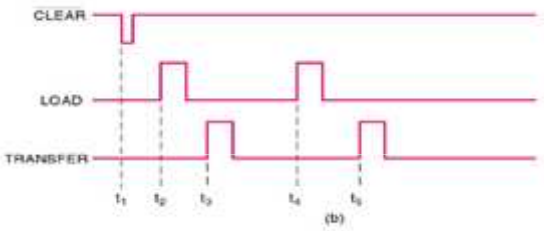
(b)

Complete circuitry for a full adder

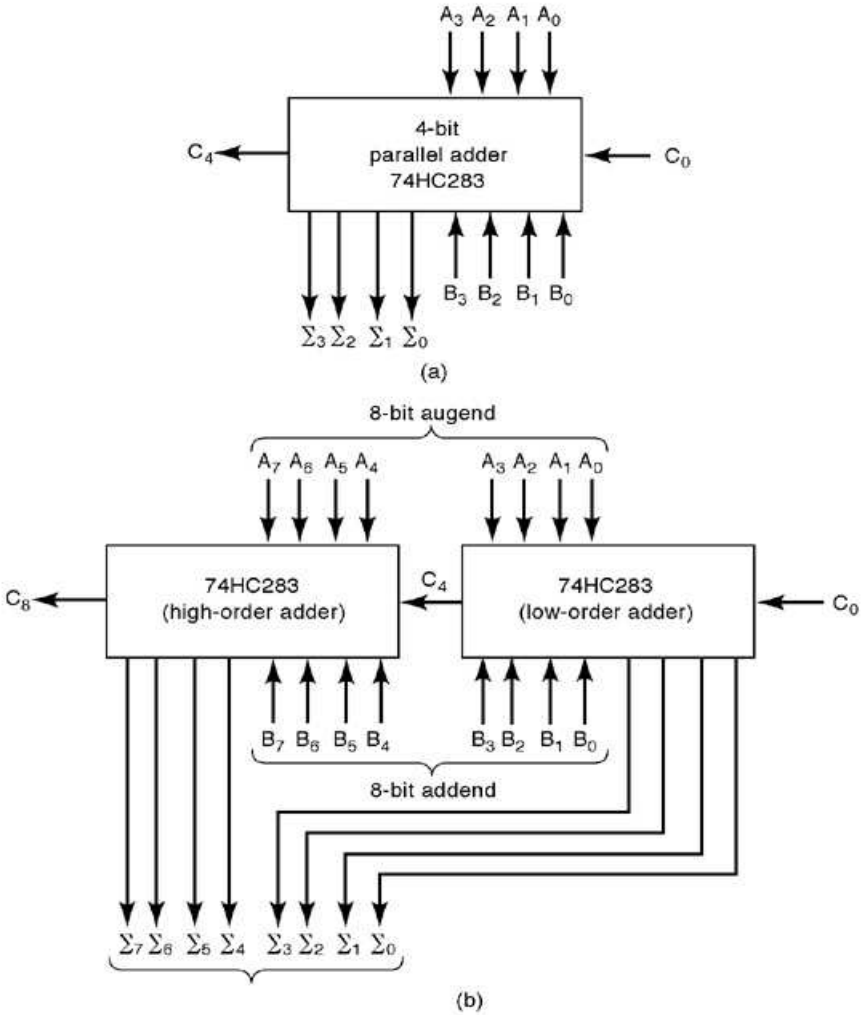


Complete parallel adder with registers





Integrated-Circuit Parallel Adder



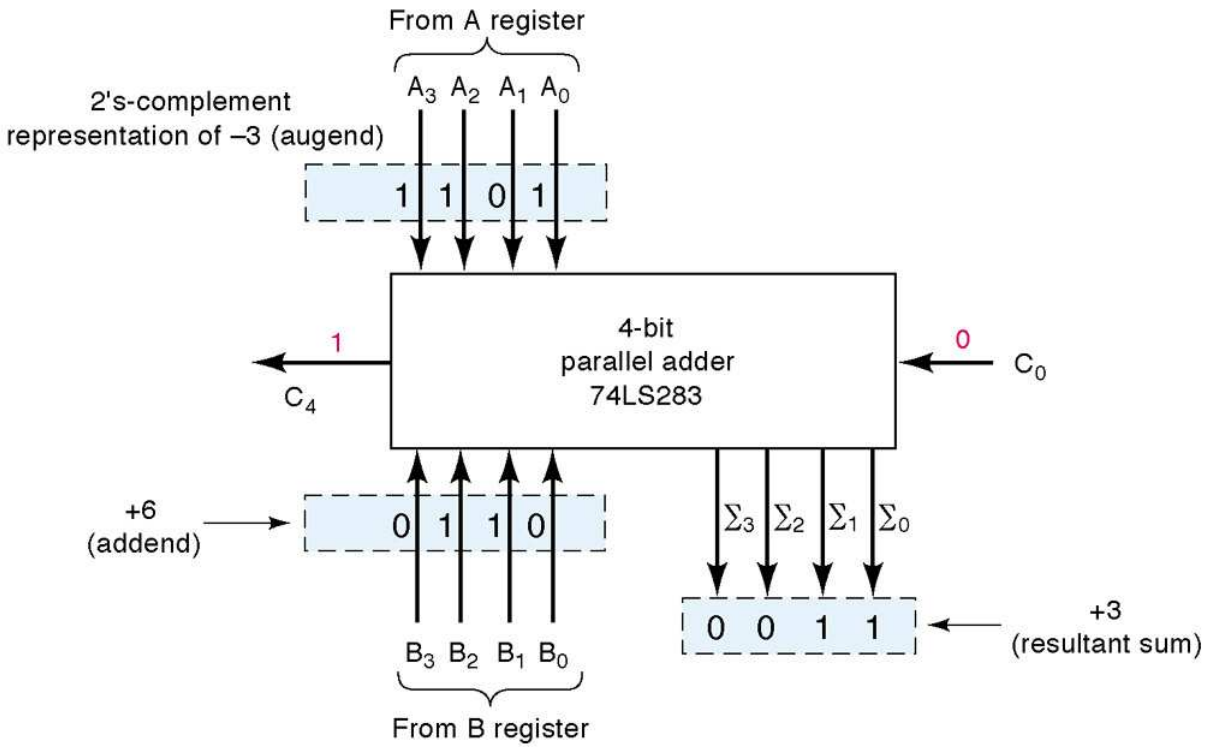
Example

- Determine the logic levels at the inputs and outputs of the eight-bit adder when 72_{10} is added to 137_{10}.

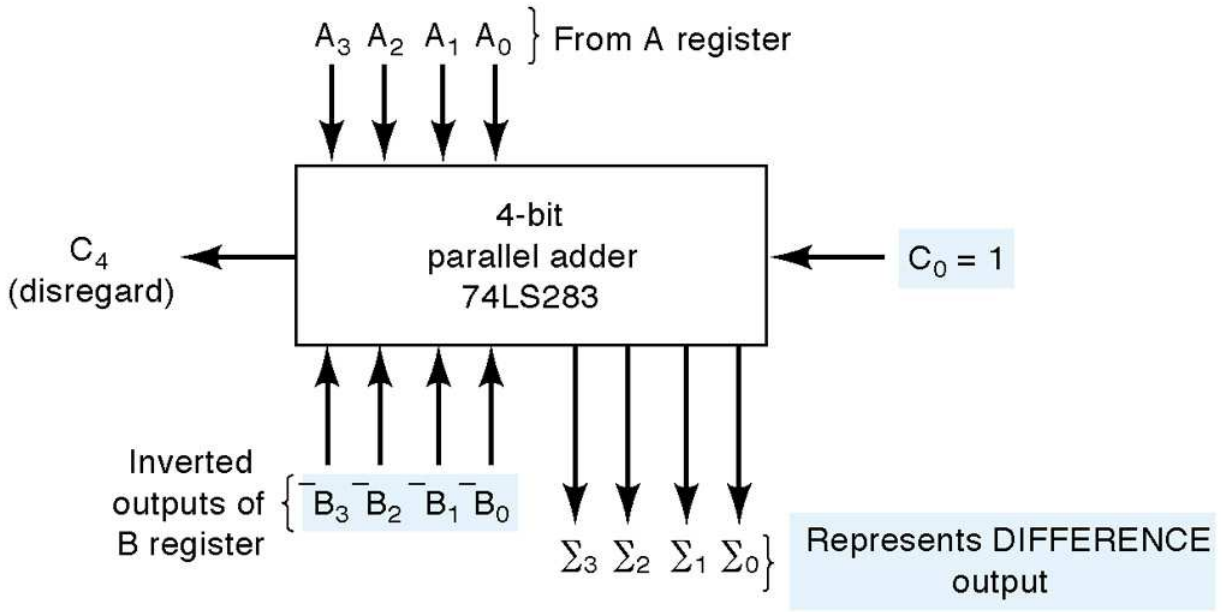
2’s Complement system

- The operations of addition and subtraction of signed numbers can be performed using only the addition operation if we use the 2’s complement form to represent negative numbers.

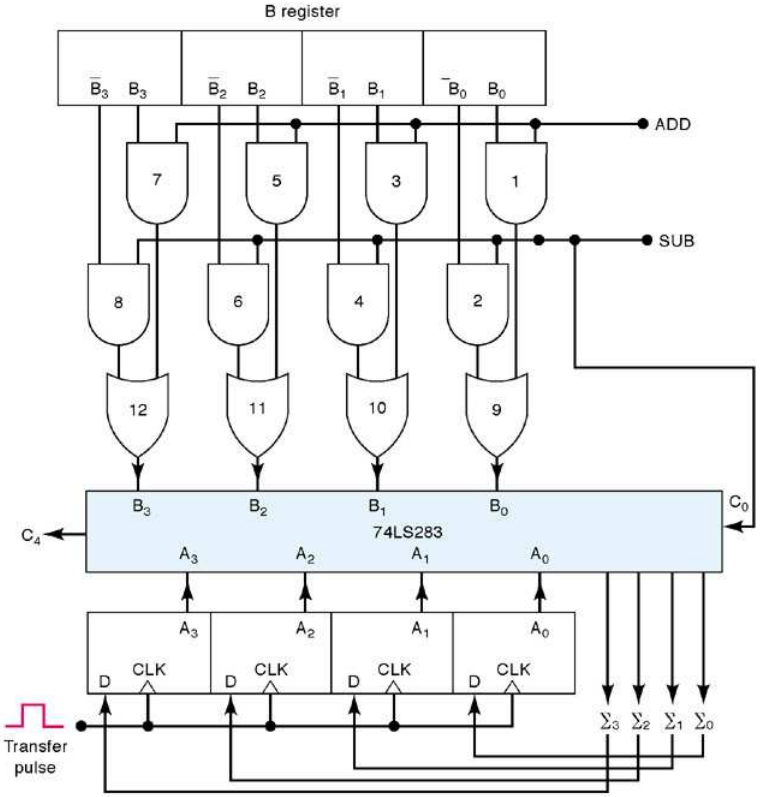
Addition


From A register
A_3 A_2 A_1 A_0
2's-complement
representation of –3 (augend)
1 1 0 1
1
C_4
4-bit
parallel adder
74LS283
0
C_0
+6
(addend)
0 1 1 0
Σ_3 Σ_2 Σ_1 Σ_0
0 0 1 1
+3
(resultant sum)
B_3 B_2 B_1 B_0
From B register

Subtraction



Combined Addition and Subtraction



BCD ADDER

- Add the BCD code groups for each decimal digit position; use ordinary binary addition.
- For those positions where the sum is 9 or less, the sum is in proper BCD form and no correction is needed
- When the sum of two digits is greater than 9, a correction of 0110 should be added to that sum to produce the proper BCD result. This will produce a carry to be added to the next decimal position.
  - $A_3A_2A_1A_0$ ← BCD code group
  - $\underline{B_3}\underline{B_2}\underline{B_1}\underline{B_0}$ ← BCD code group

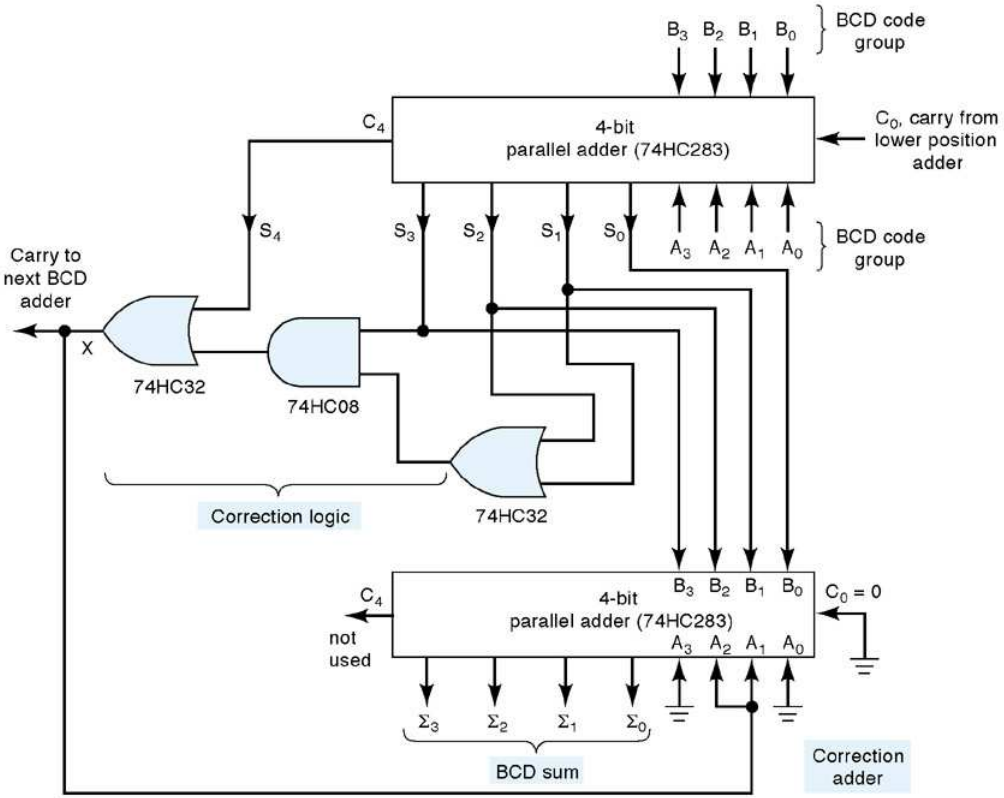
  $S_4S_3S_2S_1S_0$ ← straight binary sum

Truth table

S_4	S_3	S_2	S_1	S_0	
0	1	0	1	0	(10)
0	1	0	1	1	(11)
0	1	1	0	0	(12)
0	1	1	0	1	(13)
0	1	1	1	0	(14)
0	1	1	1	1	(15)
1	0	0	0	0	(16)
1	0	0	0	1	(17)
1	0	0	1	0	(18)

$X=S_4+S_3(S_2+S_1)$

A BCD adder


B3 B2 B1 B0
BCD code group
C4
4-bit parallel adder (74HC283)
C0, carry from lower position adder
S4 S3 S2 S1 S0
A3 A2 A1 A0
BCD code group
Carry to next BCD adder
X
74HC32
74HC08
74HC32
Correction logic
C4
4-bit parallel adder (74HC283)
B3 B2 B1 B0
C0 = 0
A3 A2 A1 A0
not used
Σ3 Σ2 Σ1 Σ0
BCD sum
Correction adder

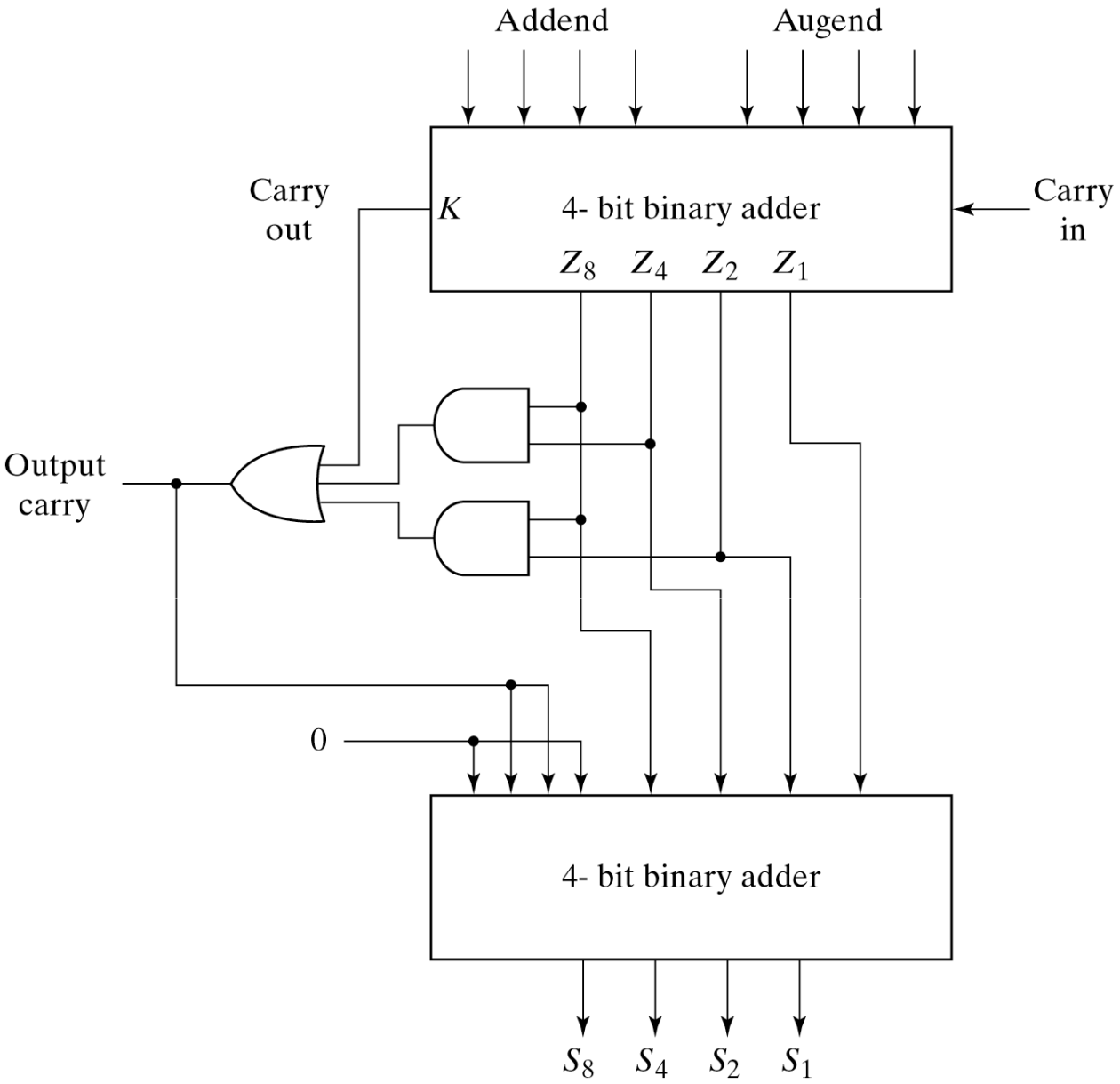
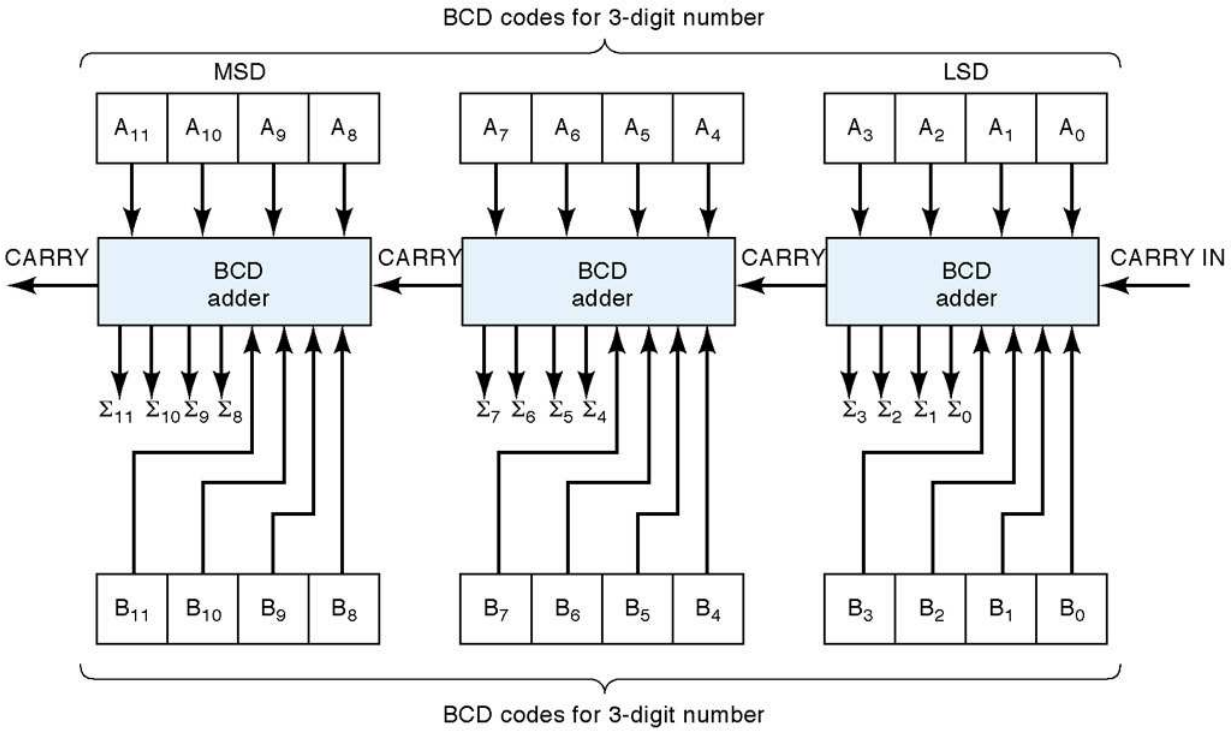


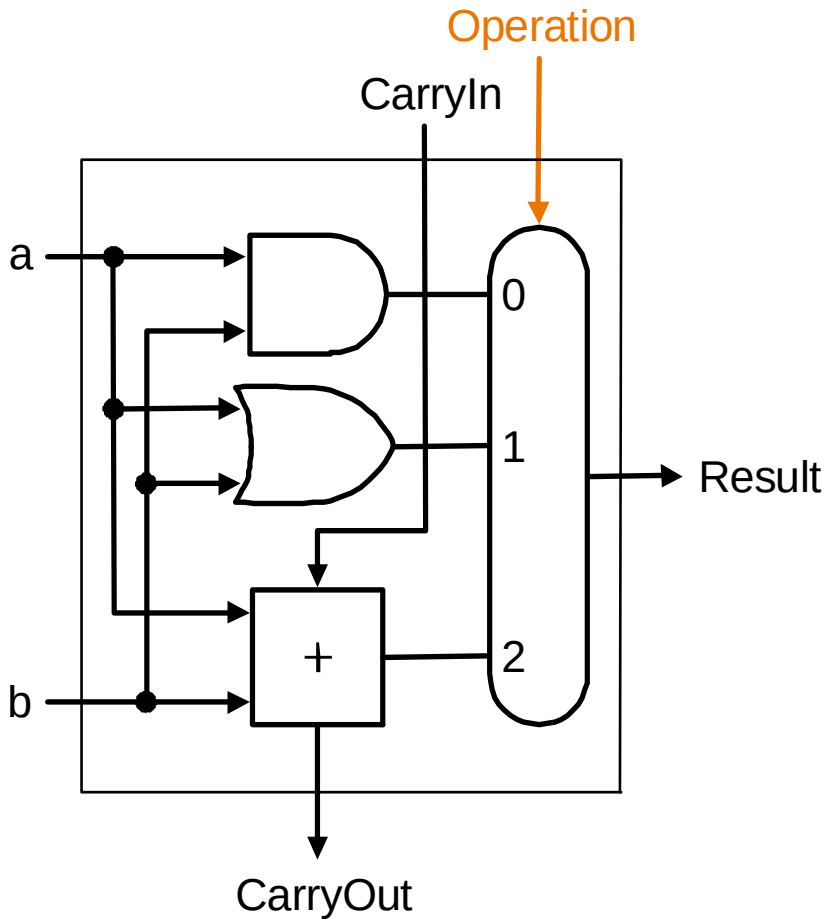
Fig. 4-14 Block Diagram of a BCD Adder

Example

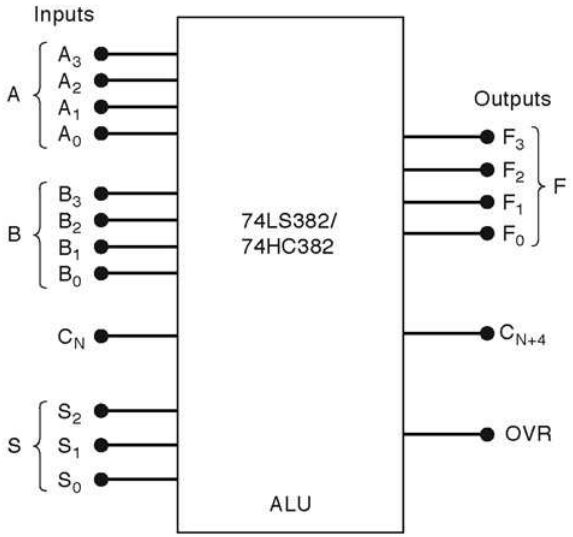
- Determine the inputs and outputs when the below circuit is used to add 538_{10} to 247_{10}. Assume CARRY IN=0.



ALU(1 bit)



ALU(4-Bit) Integrated Circuits



A = 4-bit input number
B = 4-bit input number
C_N = carry into LSB position
S = 3-bit operation select inputs

F = 4-bit output number
C_{N+4} = carry out of MSB position
OVR = overflow indicator

(a)

Function Table

S_2	S_1	S_0	Operation	Comments
0	0	0	CLEAR	$F_3F_2F_1F_0 = 0000$
0	0	1	B minus A	Needs $C_N = 1$
0	1	0	A minus B	Needs $C_N = 1$
0	1	1	A plus B	Needs $C_N = 0$
1	0	0	A ⊕ B	Exclusive-OR
1	0	1	A + B	OR
1	1	0	AB	AND
1	1	1	PRESET	$F_3F_2F_1F_0 = 1111$

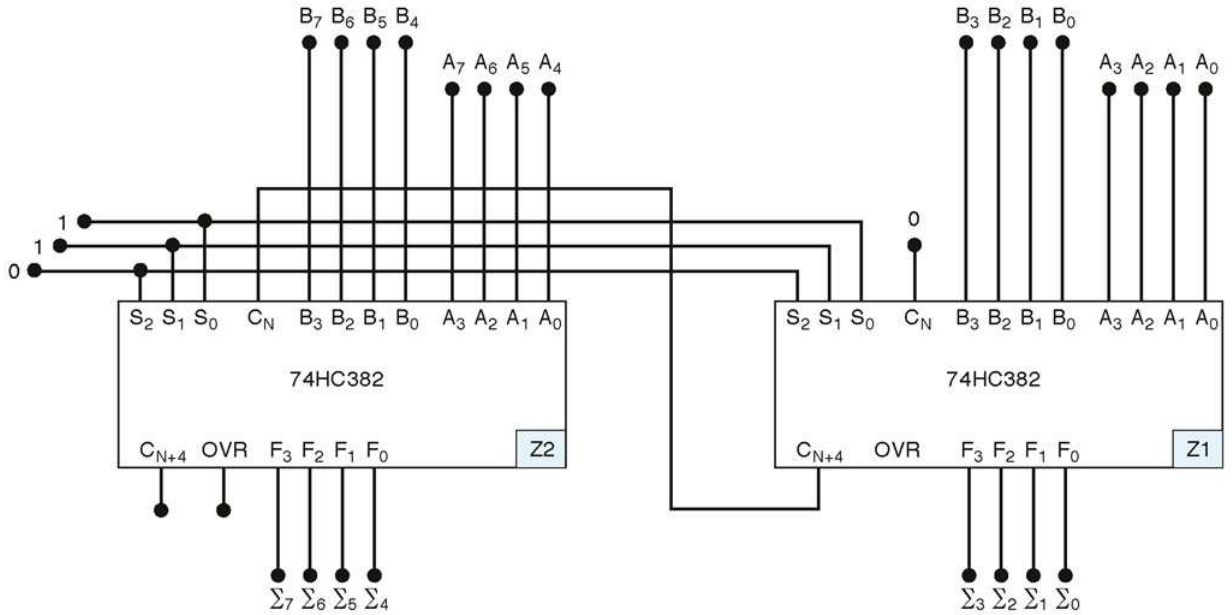
Notes: S inputs select operation.
OVR = 1 for signed-number overflow.

(b)

Operations

- CLEAR
- ADD
- SUBTRACT
- XOR
- OR
- AND
- PRESET

Expanding the ALU



Notes: Z1 adds lower-order bits.
Z2 adds higher-order bits.
Σ_7–Σ_0 = 8-bit sum.
OVR of Z2 is 8-bit overflow indicator.

Assignment 3

- 6-26 to 6-39