מעגלים לוגיים
בניית ALU

ארגון המחשב ושפת סף

מרצה: ***רועי אש***

אפקה
המכללה האקדמית
להנדסה בתל אביב

הקדמה: אלגברה בוליאנית - תזכורת

- משתנה בוליאני: משתנה המקבל את הערכים 0 או 1
- פונקציה בוליאנית: פונקציה של משנה/ים בוליאניים המחזירה 0 או 1
- כמה פונקציות בוליאניות יש למשתנה בודד X?
- נבנה טבלת אמת עם קלט (משתנים) ופלט (פונקציה):

X	F
0	0
1	0

X	F
0	1
1	1

X	F
0	0
1	1

X	F
0	1
1	0

NOT

- יש ארבע פונקציות אונאריות (של משתנה אחד) אבל רק NOT באמת מעניינת

כמה פונקציות בוליאניות של שני משתנים?

Y	X	FUNC
0	0	B0
0	1	B1
1	0	B2
1	1	B3

- 2 בחזרת מספר השורות בטבלת האמת
- מספר השורות בטבלת האמת זה שתיים בחזקת מספר המשתנים

אלגברה בוליאנית – חוקים (תזכורת)

- הסימון + מסמן OR
- הסימון • (כפל) מסמן AND
- הסימון ¯ (קו עליון מעל) מסמן NOT
- מספר חוקים בסיסיים באלגברה בוליאנית:

- Identity law: $A + 0 = A$ and $A \cdot 1 = A$
- Zero and One laws: $A + 1 = 1$ and $A \cdot 0 = 0$
- Inverse laws: $A + \overline{A} = 1$ and $A \cdot \overline{A} = 0$
- Commutative laws: $A + B = B + A$ and $A \cdot B = B \cdot A$
- Associative laws: $A + (B + C) = (A + B) + C$ and $A \cdot (B \cdot C) = (A \cdot B) \cdot C$
- Distributive laws: $A \cdot (B + C) = (A \cdot B) + (A \cdot C)$ and $A + (B \cdot C) = (A + B) \cdot (A + C)$

טבלאות אמת – Boolean Functions

- כל פונקציה בוליאנית ניתנת להצגה כטבלת אמת
  - טבלת אמת: קלט נקודתי ← מיפוי פלט
  - הפונקציה היא הפרדה של אפשרויות הקלט לשורות נפרדות, בעת שהפלט מייצג אפשרות אחת של תוצאה בוליאנית
- כל טבלת אמת ניתנת להצגה כסכום של מכפלות
  - לדוגמה טבלת האמת משמאל ניתנת להצגה כ:

Out = AB'C + ABC' + ABC

A	B	C	→	Out
0	0	0	→	0
0	0	1	→	0
0	1	0	→	0
0	1	1	→	0
1	0	0	→	0
1	0	1	→	1
1	1	0	→	1
1	1	1	→	1

כלל סכום המכפלה: Sum Of Products (SOP)

- התבוננו בערך הפונקציה בכל אחת משורות טבלת האמת
  - במידה והערך הוא 0, התעלמו
  - במידה והערך הוא 1 בצעו את המכפלה (AND) המתאימה
  - בתוך המכפלה ערך קלט אפס יקבל NOT
  - בסיום התהליך סכמו (OR) את המכפלות

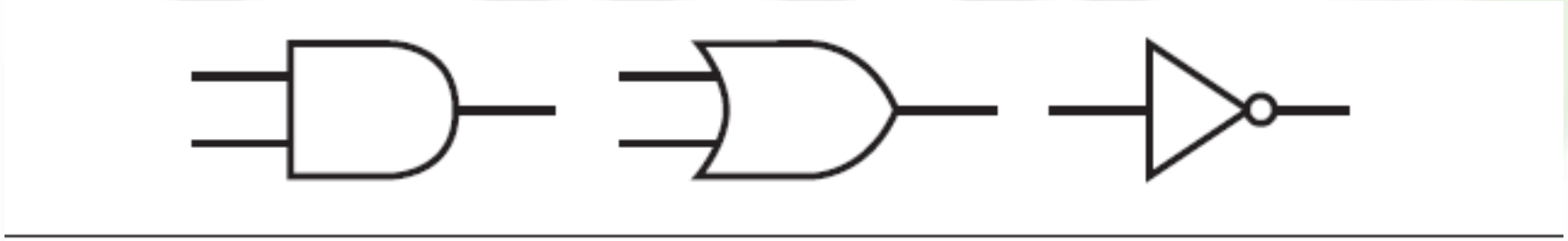
למעשה, מה שהכלל אומר: ניתן להציג כל פונקציה בוליאנית בעזרת AND, OR, NOT

A	B	FUNC: A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

A xor B = A’B + AB’

שערים לוגיים - תזכורת

- שערים עבור AND ו- OR, יכולים לקבל מספר קלטים, ופלט אחד (ע"ס החוקים בשקף הקודם, ניתן להרחיב שערים אלו למספר קלטים ופלט אחד)
- שער NOT, מקבל קלט אחד עם פלט אחד



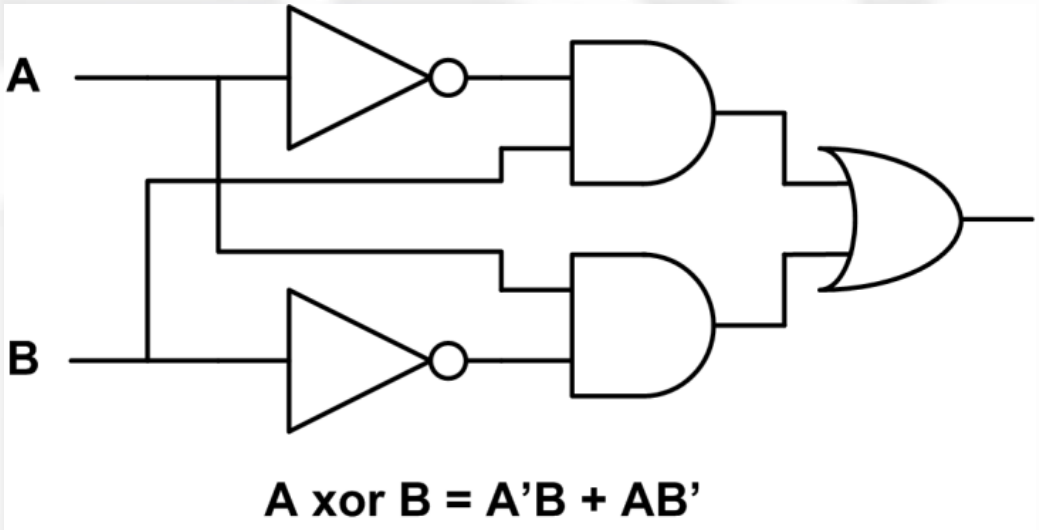
Standard drawing for an AND gate, OR gate, and an inverter

- רישום מקוצר ל NOT, ניתן על ידי רישום "בועה" על הקו. משמאל, למשל:

$$\overline{\overline{A} + B}$$

מעל לוגי: חיבור מספר שערים לוגיים

- בחיבור השערים: יש התפצלות/צומת מעקף קווים וקיצורים ע"פ כללי האלגברה הבוליאנית (רכיב מרובה רגליים not מקוצר)



- האם קיים הבדל בין המעגל של XOR לתיאור הבוליאני (מתמטי) של XOR?

זמן ההתייצבות של המעגל

נספח בניית רכיבים לוגיים בסיסים
The Basics of logic design

- הרכיבים אותם נבנה:
- ALU *Arithmetic Logic Unit - יחידה אריתמטית לוגית*
- Combinational logic המורכב מלוגיקה צירופית
- Mips Registers File (במצגת הבאה שנלמד)

A state element (Sequential logic) that consists of a set of registers that can be read and written by supplying a register number to be accessed.

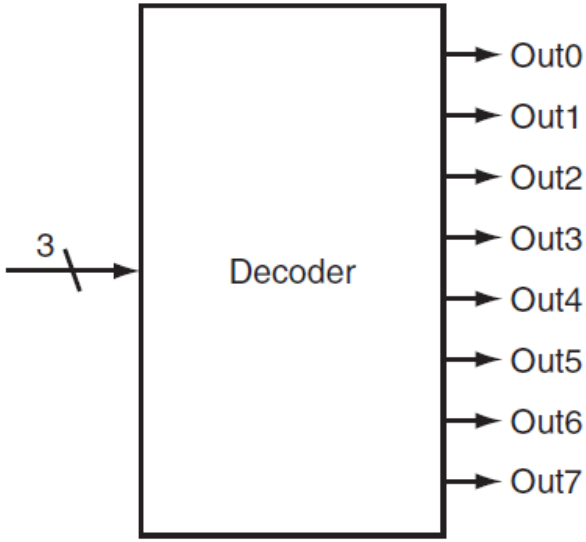
The ALU

יחידה אריתמטית לוגית Arithmetic Logic Unit

- ה ALU מורכב מלוגיקה צירופית- Combinational logic
- לוגיקה צירופית: שינוי בקלט גורם באופן ישיר לשינוי בפלט (לאחר זמן השהיה)
- בשונה מלוגיקה סדרתית -sequential logic (בחלק השני של המצגת) אשר השינוי בפלט הינו תלוי שעון (זיכרון \ אוגר).

- שני רכיבים מרכזיים בלוגיקה צירופית הינם:
  - מפענח\מקודד (decoder /encoder)
  - מרבב\ מפלג (Multiplexer/Demultiplexer) או בקיצור mux

מפענחים - Decoders

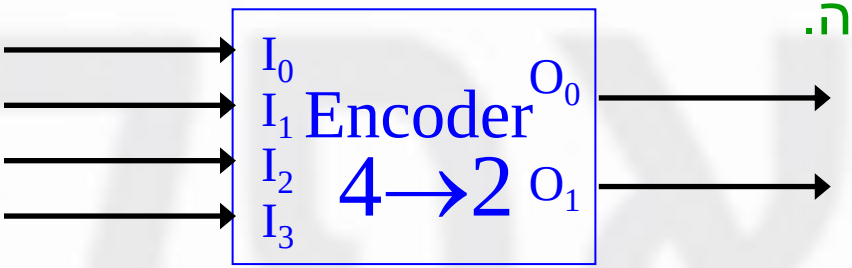


Inputs			Outputs							
12	11	10	Out7	Out6	Out5	Out4	Out3	Out2	Out1	Out0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

- כל צירוף של קלטים מאפשר בדיוק '1' אחד בפלט
- עבור n קווי קלט – יש 2^n קווי פלט

מקודדים - Encoders

- ממש פונקציה "הפוכה" למפענח:
- 2^n קוי כניסה.
- n קוי יציאה.

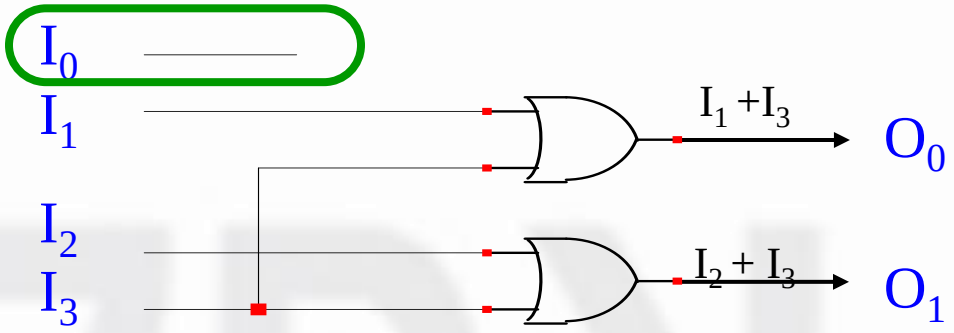


קלט				פלט		
I_3	I_2	I_1	I_0	O_1	O_0	
0	0	0	1	0	0	→0
0	0	1	0	0	1	→1
0	1	0	0	1	0	→2
1	0	0	0	1	1	→3

- קלט שאינו "אונארי" יביא ליציאה שגויה או לא מוגדרת.

הגדרה "טובה" יותר של מקודד?

מימוש מקודדים:



- קו I_0 אינו "מחובר".
- מכיוון שקל לממש מקודד ע"י שערי OR בלבד וכן יש "להיזהר" לא להגדיר קלט לא חוקי ⇐ ישנן הרחבות למקודדים מורכבים וכלליים יותר.

מקודד סדר עדיפויות – Priority Encoder

- 2^n כניסות.
- n יציאות + יציאת Valid.

בודק "תקינות" הקלט.

מקודד עדיפויות – Priority Encoder:

- הפלט מציין את הביט הראשון (MSB) שהינו "1".

I_3	I_2	I_1	I_0	O_1	O_0	V	
0	0	0	0	0	0	0	
0	0	0	1	0	0	1	0
0	0	1	∅	0	1	1	1
0	1	∅	∅	1	0	1	2
1	∅	∅	∅	1	1	1	3

not valid

Valid

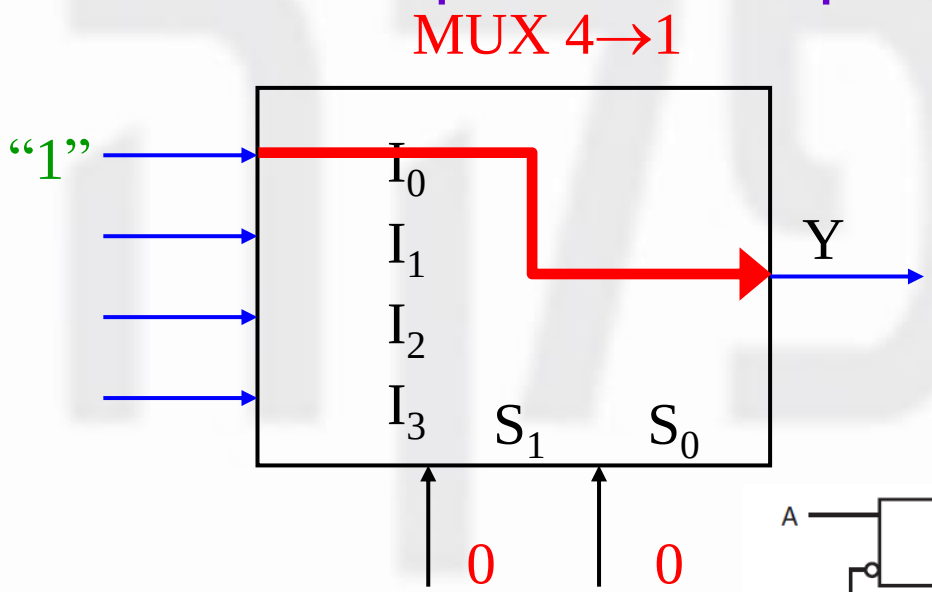
$$O_1 = I_2 + I_3$$

$$O_0 = I_3 + I_1 * \overline{I_2}$$

$$V = I_0 + I_1 + I_2 + I_3$$

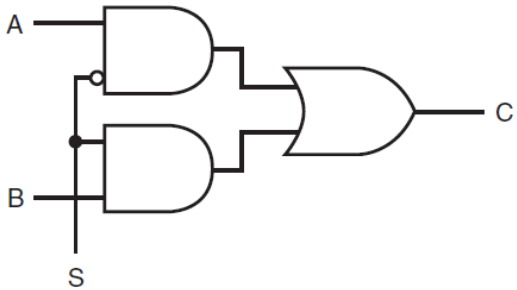
מרבבים – Multiplexor:

מרבב הוא כינוי להתקן אלקטרוני המממש פעולה בסיסית וחשובה הנקראת ריבוב בו מתבצעת בחירה של אחד מכמה ערוצי קלט לערוץ פלט, בהתאם לערך בכניסות הבקרה



- 2^n קווי כניסה.
- n קוי ברירה / בחירה / מיעון
  - בהינתן x קלטים
  - צריך $\log_2 x$ קווי בחירה
- קו יציאה יחיד.

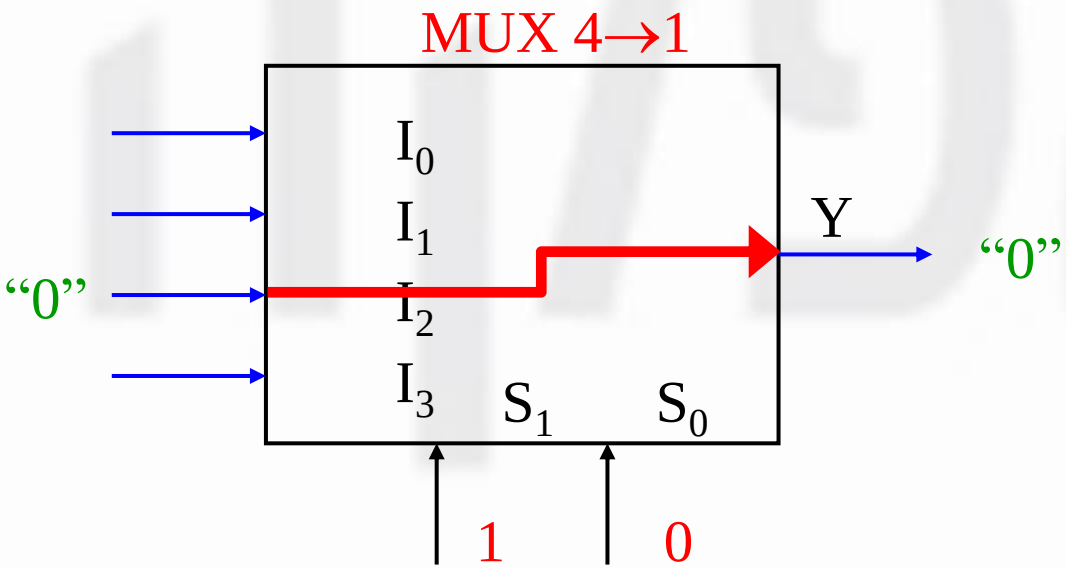
"1"



מרבבים – Multiplexor:

מרבב הוא כינוי להתקן אלקטרוני המממש פעולה בסיסית וחשובה הנקראת ריבוב בו מתבצעת בחירה של אחד מכמה ערוצי קלט לערוץ פלט, בהתאם לערך בכניסות הבקרה

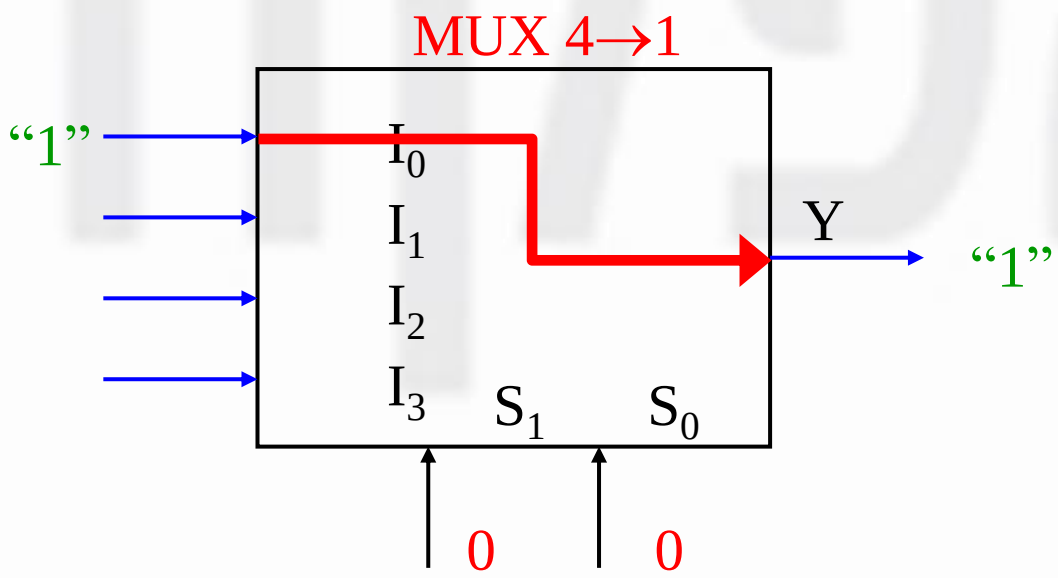
- 2^n קוי כניסה.
- n בוררים / בחירה / מיעון.
- קו יציאה יחיד.



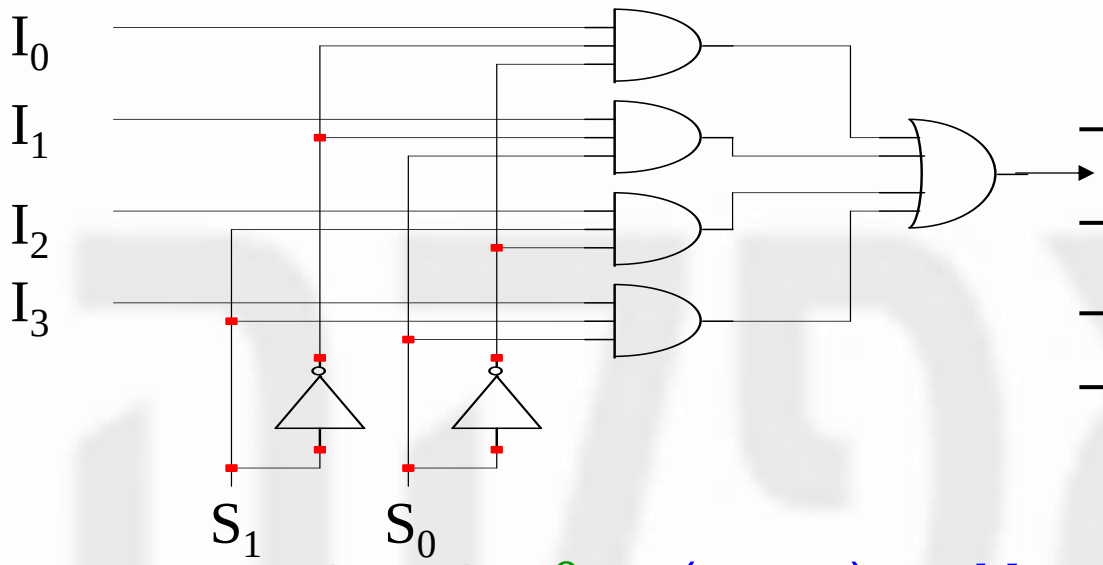
מרבבים – Multiplexor:

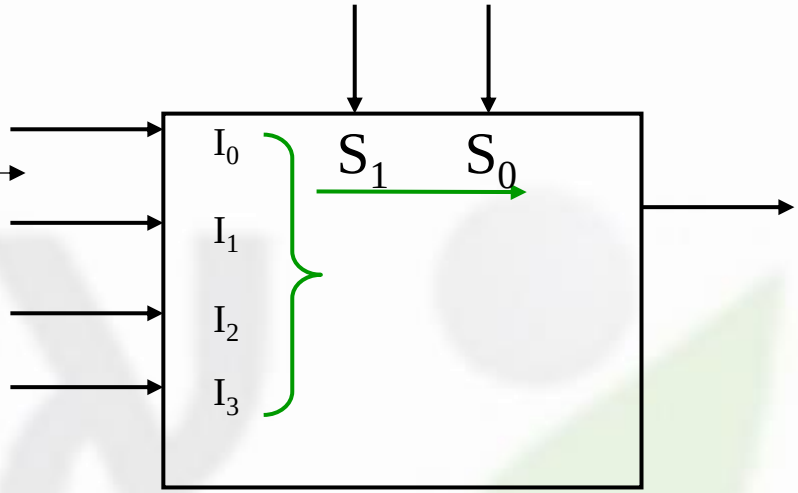
ריבוב: שידור מספר רב של יחידות מידע באמצעות מספר קטן יותר של קווים או ערוצים. מרבב ספרתי בורר קו יחיד מבין קוי כניסה ומכוון את המידע הבינארי אל קו יציאה יחיד.

- 2^n קוי כניסה.
- n בוררים / בחירה / מיעון.
- קו יציאה יחיד.

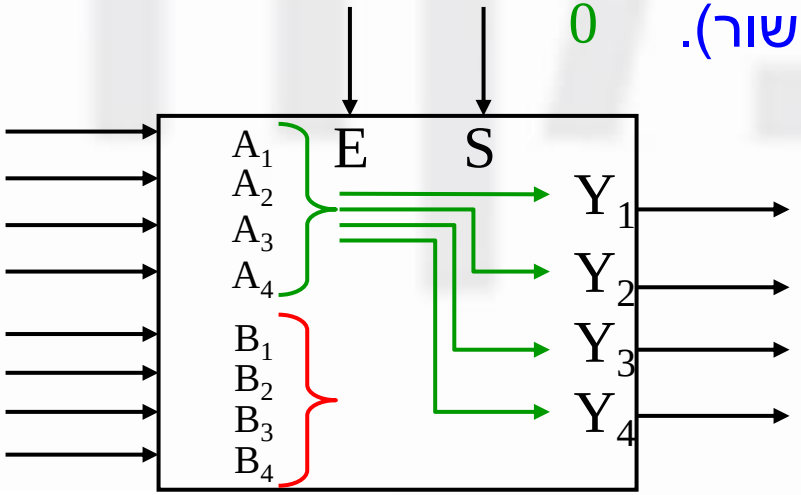


מימוש מרבב





- בדומה למפענחים מוסיפים קו Enable (אפשור).
- ניתן לברור בין קבוצות של קוים (רוחב פס).



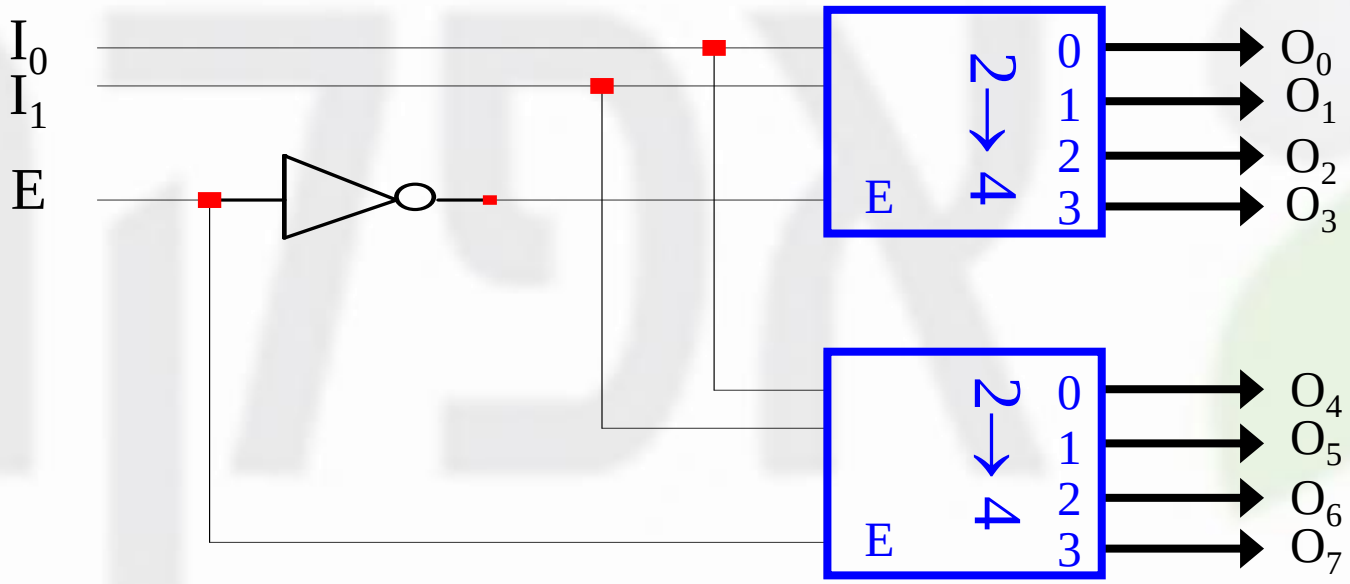
טבלת הפונקציה

E	S	Output Y
0	∅	All O's
1	0	Select A
1	1	Select B

שימוש ב – Enable:

(Enable – אפשור)

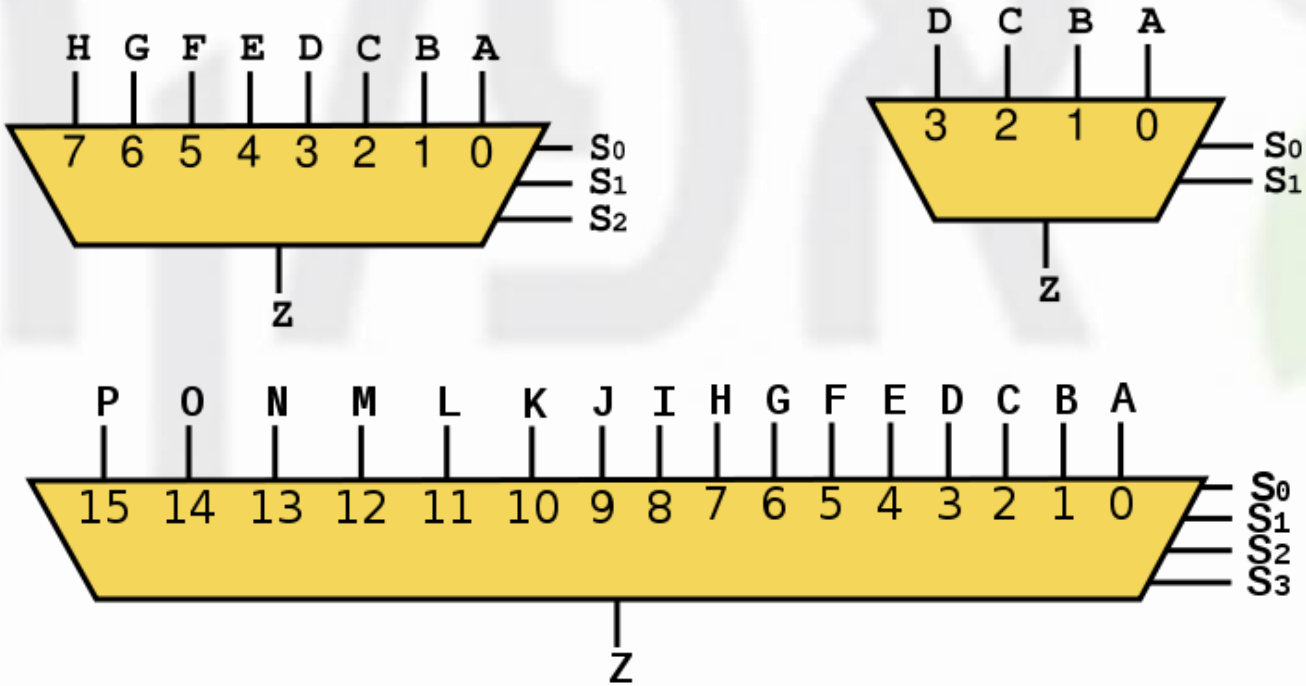
נממש $n \to 2^n$ Decoder ע"י שני $n-1 \to 2^{n-1}$, ושימוש בקו Enable:



- כאשר E=0 יופעל ה-Decoder העליון ולכן יבחר ע"י I_0 I_1 את היציאה המתאימה להיות 1.
  במקרה זה ה-Decoder התחתון יוציא אפסים.
- התפקוד משתנה (עליון יוציא 4 אפסים) כאשר E=1.

מרבבים- Multiplexors (MUXes)

עקרונית ניתן קיימים גדלים שונים של מרבבים
פעולת הריבוב (בו מתבצעת בחירה של אחד מכמה ערוצי קלט לערוץ פלט,
בהתאם לערך בכניסות הבקרה.) הינה פעולה חשובה ביותר בבניית מעגלים לוגיים



הסימון בספר ל 2→1 MUX נראה כך

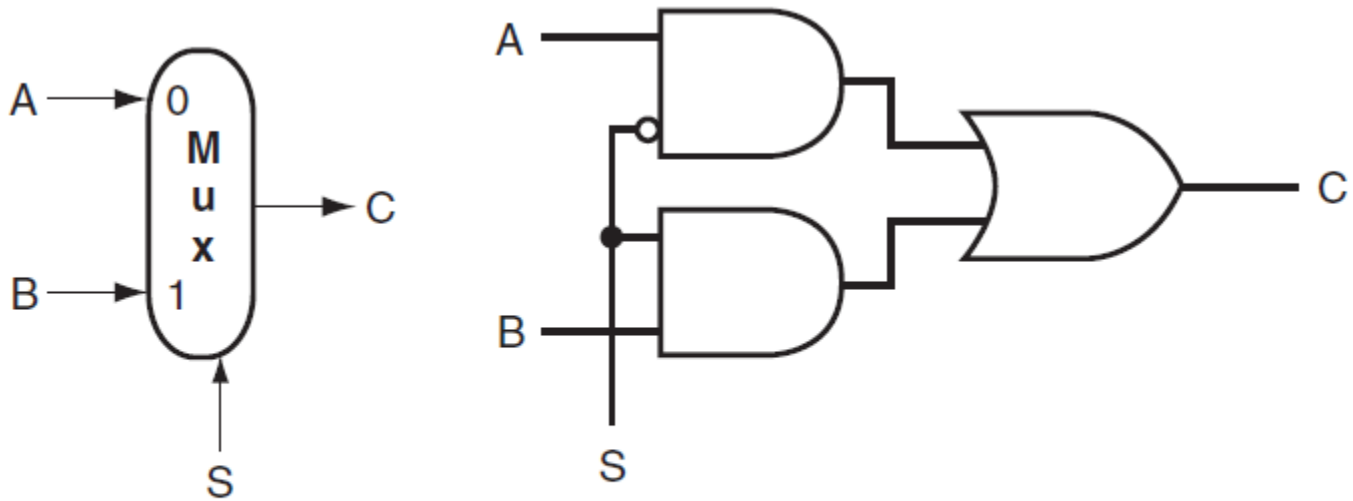
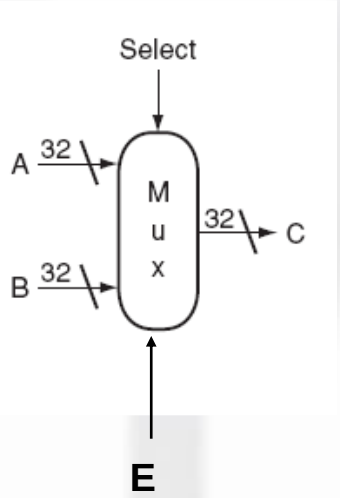


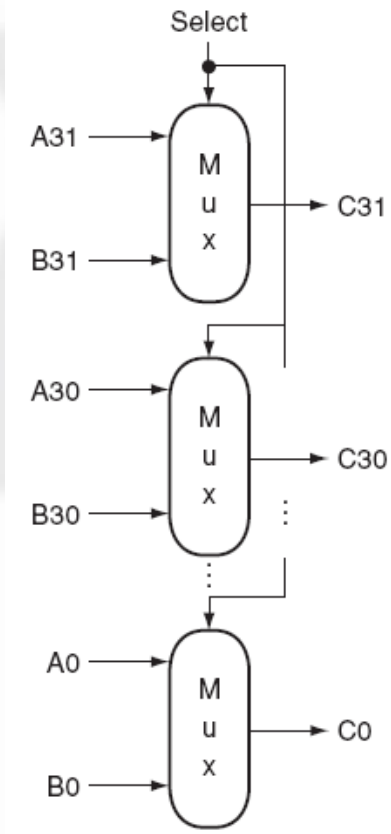
FIGURE B.3.2 A two-input multiplexor on the left and its implementation with gates on the right.

ניתן לממש ריבוב ברוחב פס שונה (bus)

לדוגמא מרבב 2 ל 1 ברוחב פס 32



בתוספת האיפשור (Enable)

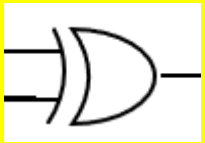


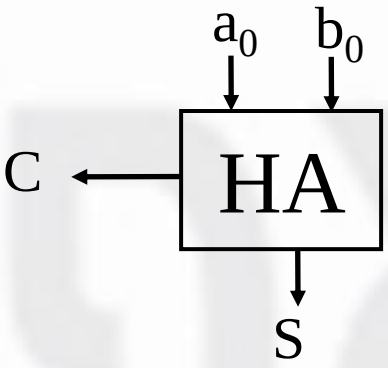
חצי מחבר – Half Adder

חצי מחבר : מקבל 2 סיביות ומחזיר את סכומן (mod 2) ואת הנשא.

סימונים ל XOR

$\oplus$

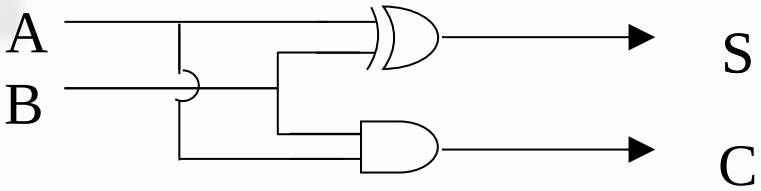




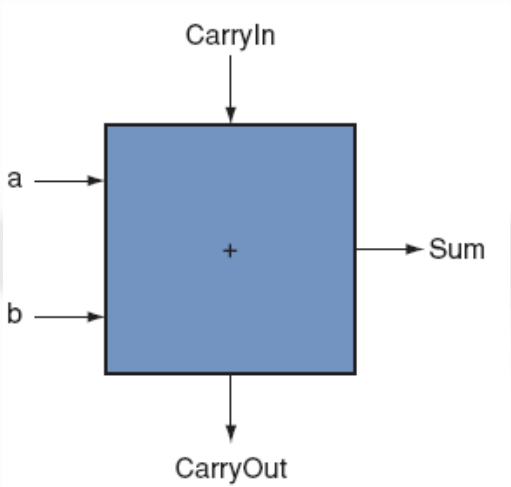
a	b	s	c
0	0	0	0
1	0	1	0
0	1	1	0
1	1	0	1

$$S = a \oplus b$$

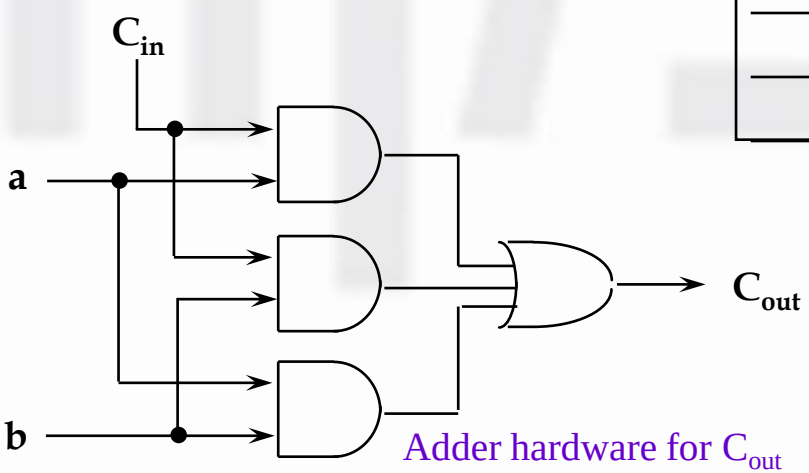
$$C = a \bullet b$$



Full Adder



inputs			outputs	
a	b	C_{in}	sum	C_{out}
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



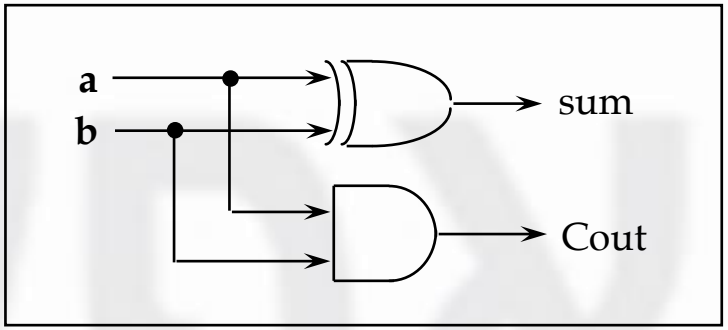
Adder hardware for C_{out} in 2 layers

$$\text{sum} = (\bar{a} \cdot \bar{b} \cdot C_{in}) + (\bar{a} \cdot b \cdot \overline{C_{in}}) + (a \cdot \bar{b} \cdot \overline{C_{in}}) + (a \cdot b \cdot C_{in}) = a \oplus b \oplus C_{in}$$

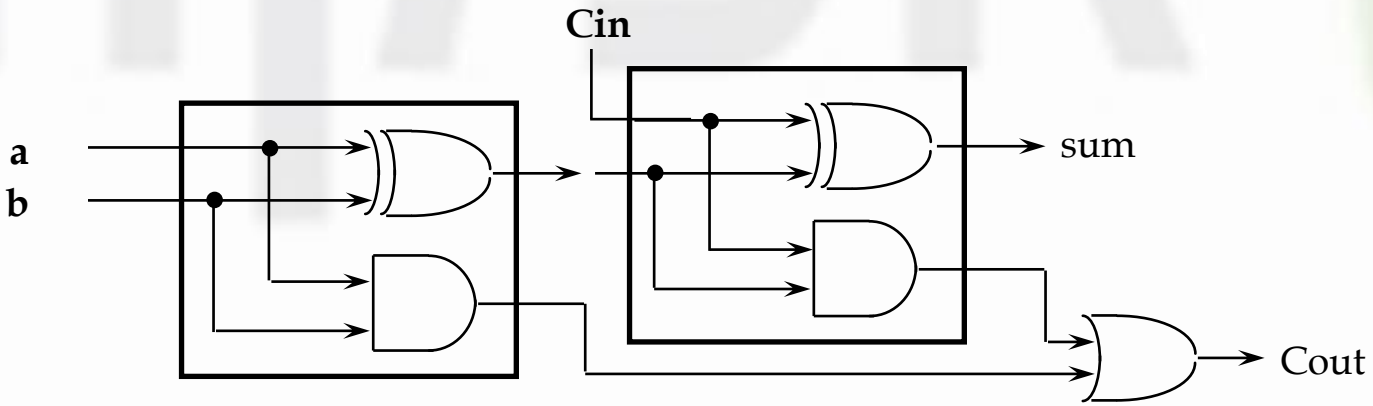
$$C_{out} = (b \cdot C_{in}) + (a \cdot C_{in}) + (a \cdot b) = ((a \oplus b) \cdot C_{in}) + (a \cdot b)$$

Full Adder from Half Adders

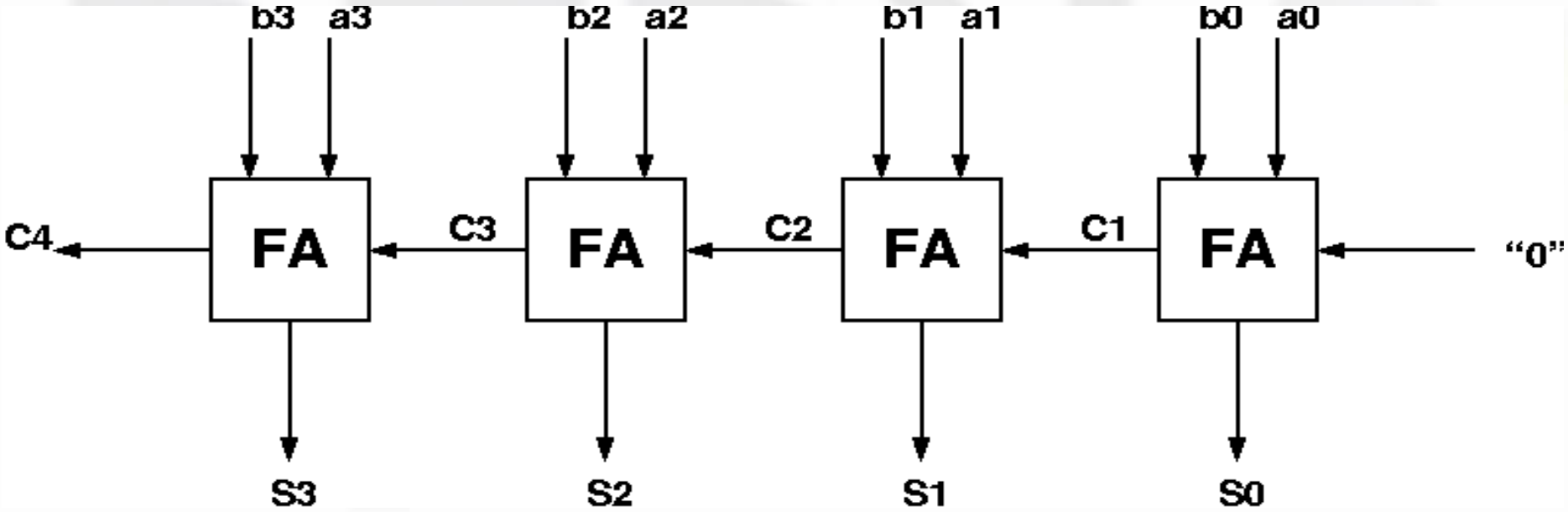
Half adder



Full adder from 2 half adders + or gate

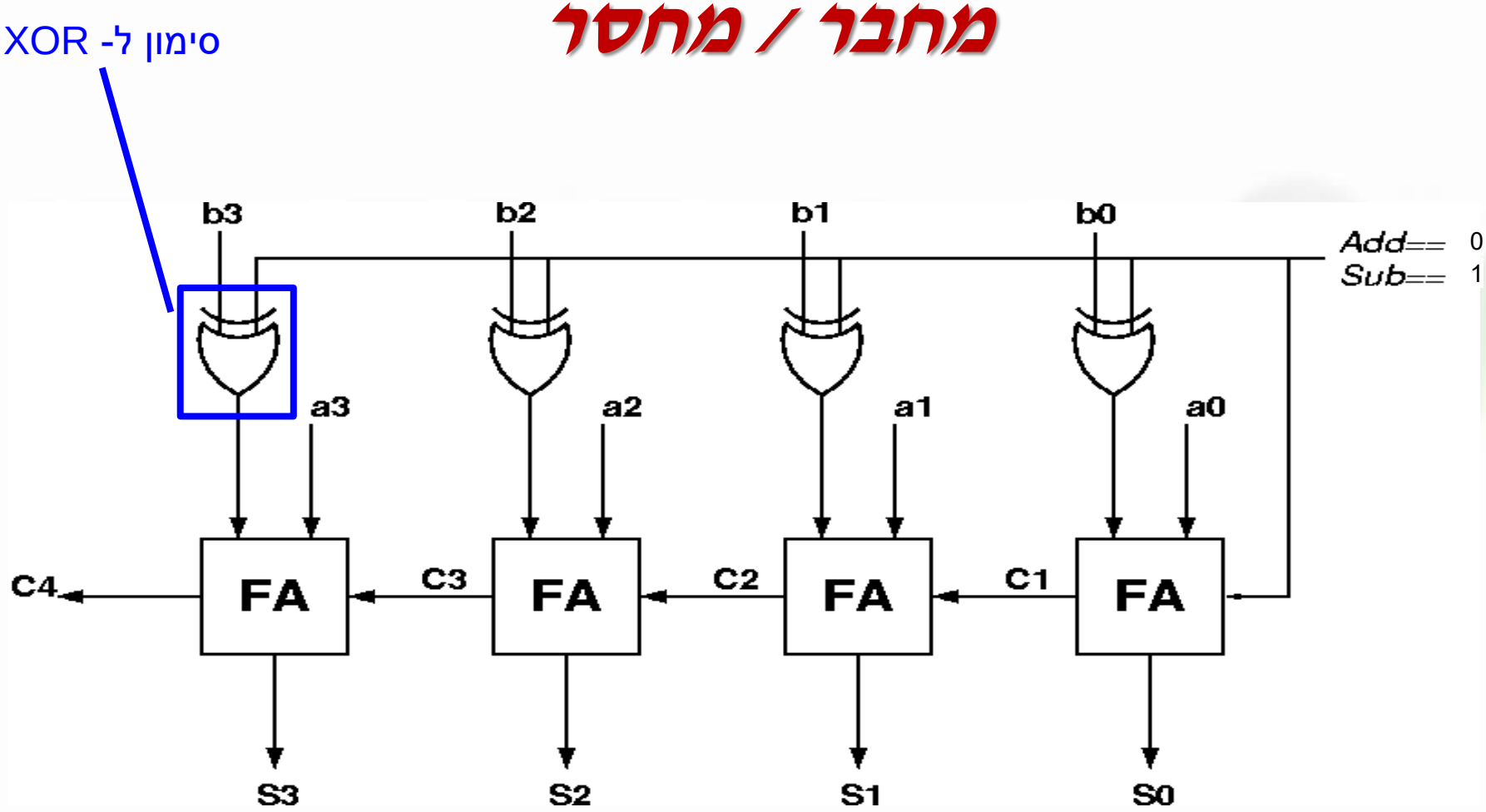


מחבר נשא גלי-Ripple Carry Adder



סביות הנשא מועברות בטור (מעין צורת גל)
לכן זמן החיבור יהיה ביחס ישר לגודל המחבר

מחבר / מחסר


סימון ל- XOR
b3
b2
b1
b0
Add== 0
Sub== 1
a3
a2
a1
a0
C4
FA
C3
FA
C2
FA
C1
FA
S3
S2
S1
S0

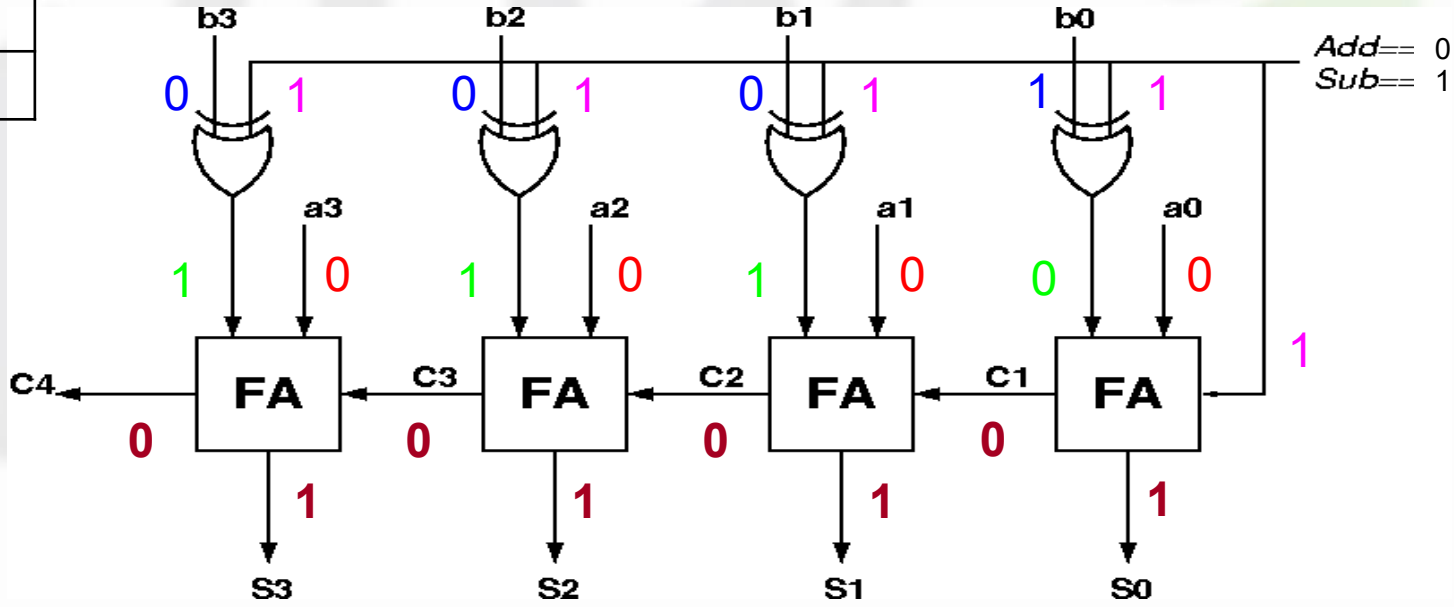
מחבר / מחסר-דוגמה

Inputs			outputs	
a	b	c_{in}	s	c_{out}
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

נחשב את:

a 0000

b -0001



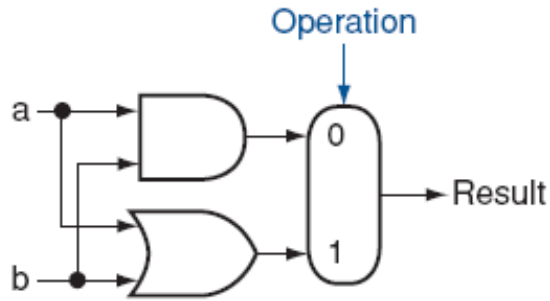
וקיבלנו את התוצאה:

1111

The ALU

- The ALU provides the basic logical (AND, OR, NOR, NAND) and arithmetic (Addition, Subtraction in 2's complement → invert + 1 and Set on less than) functions
- Shift, multiplication and division are usually outside the basic ALU.

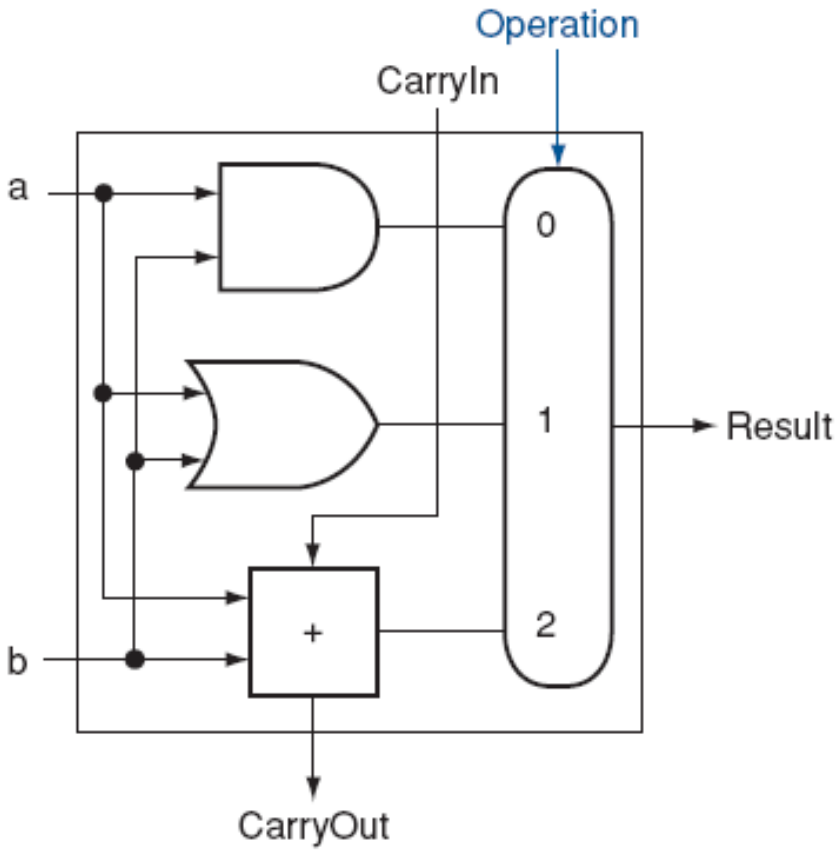
Logical operations



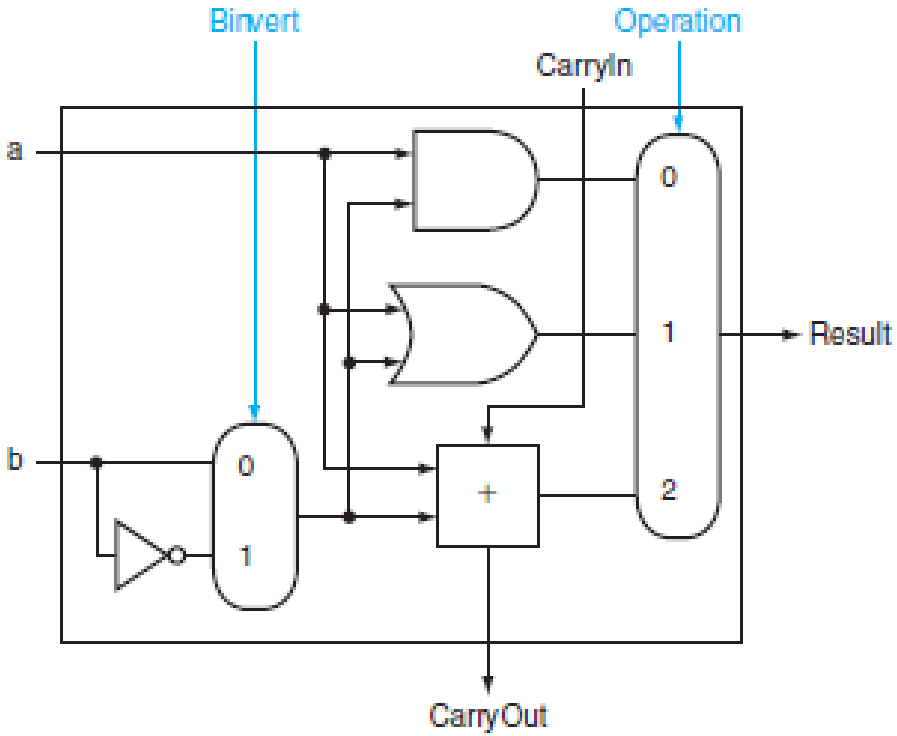
1 bit logical unit for AND/OR operations

1 Bit Simple ALU

A 1-bit ALU that performs AND, OR, and addition



1 bit Enhanced ALU



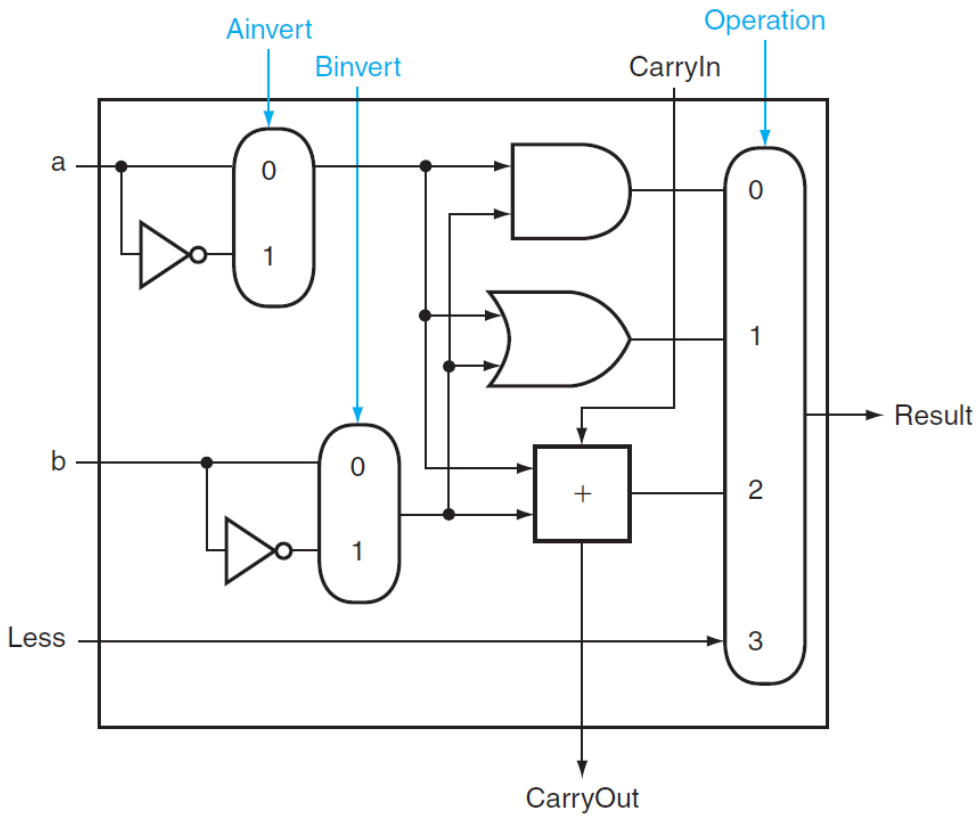
- הוספת אפשרות של חיסור
- חיסור, זה למעשה חיבור של "משלים 2"
- Binvert – קובע האם מדובר בחיבור או חיסור
- כאשר מדובר בחיסור – CarryIn הוא 1 בכדי לממש את "תוספת 1" לאחר היפוך הספרות ב- "משלים 2"

2's complement: use CarryIn = 1

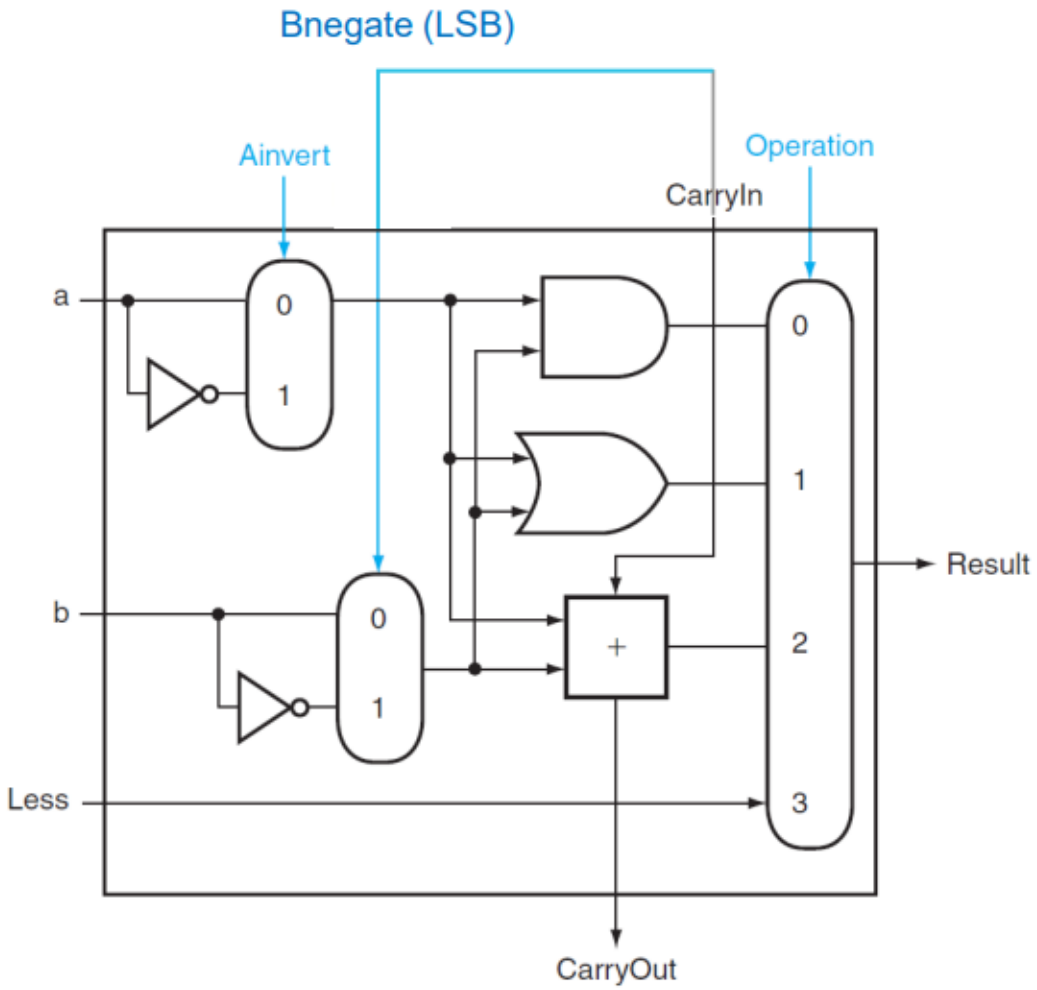
$$a + \overline{b} + 1 = a + (\overline{b} + 1) = a + (-b) = a - b$$

1 Bit Enhanced ALU

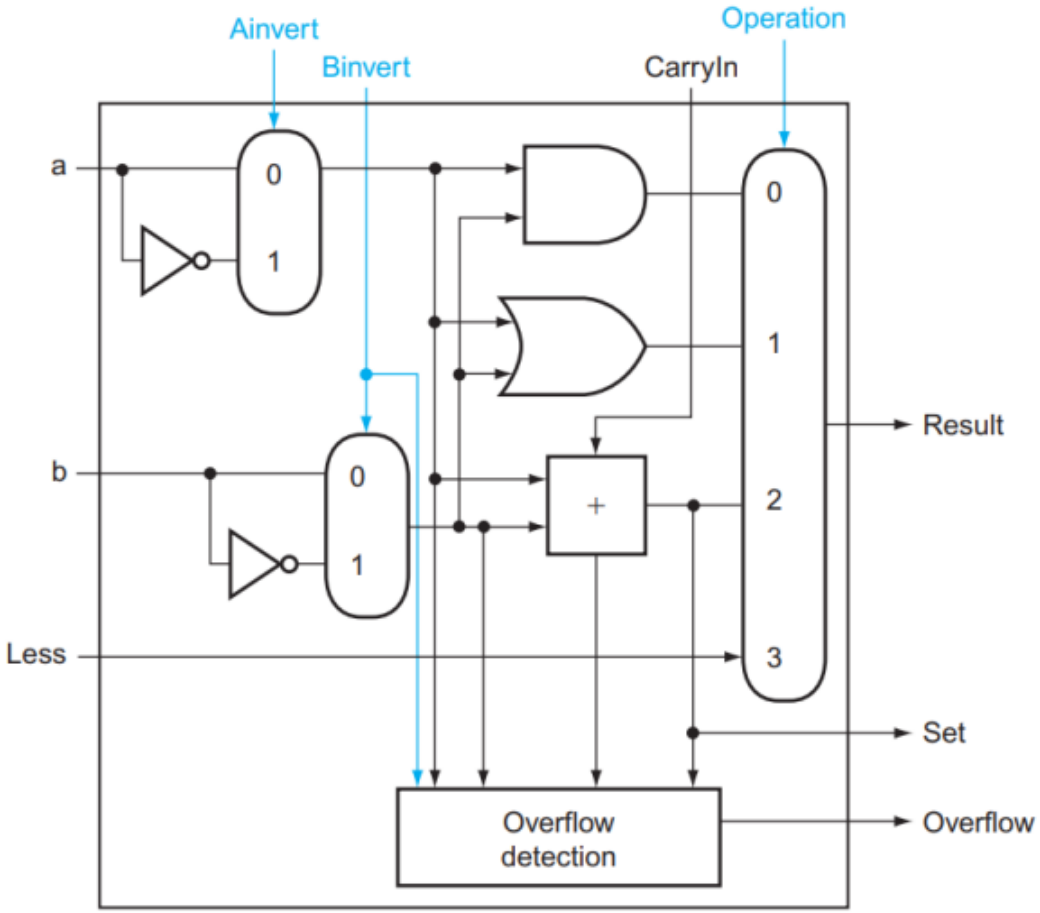
Enhanced for Ainvert and Binvert and SLT



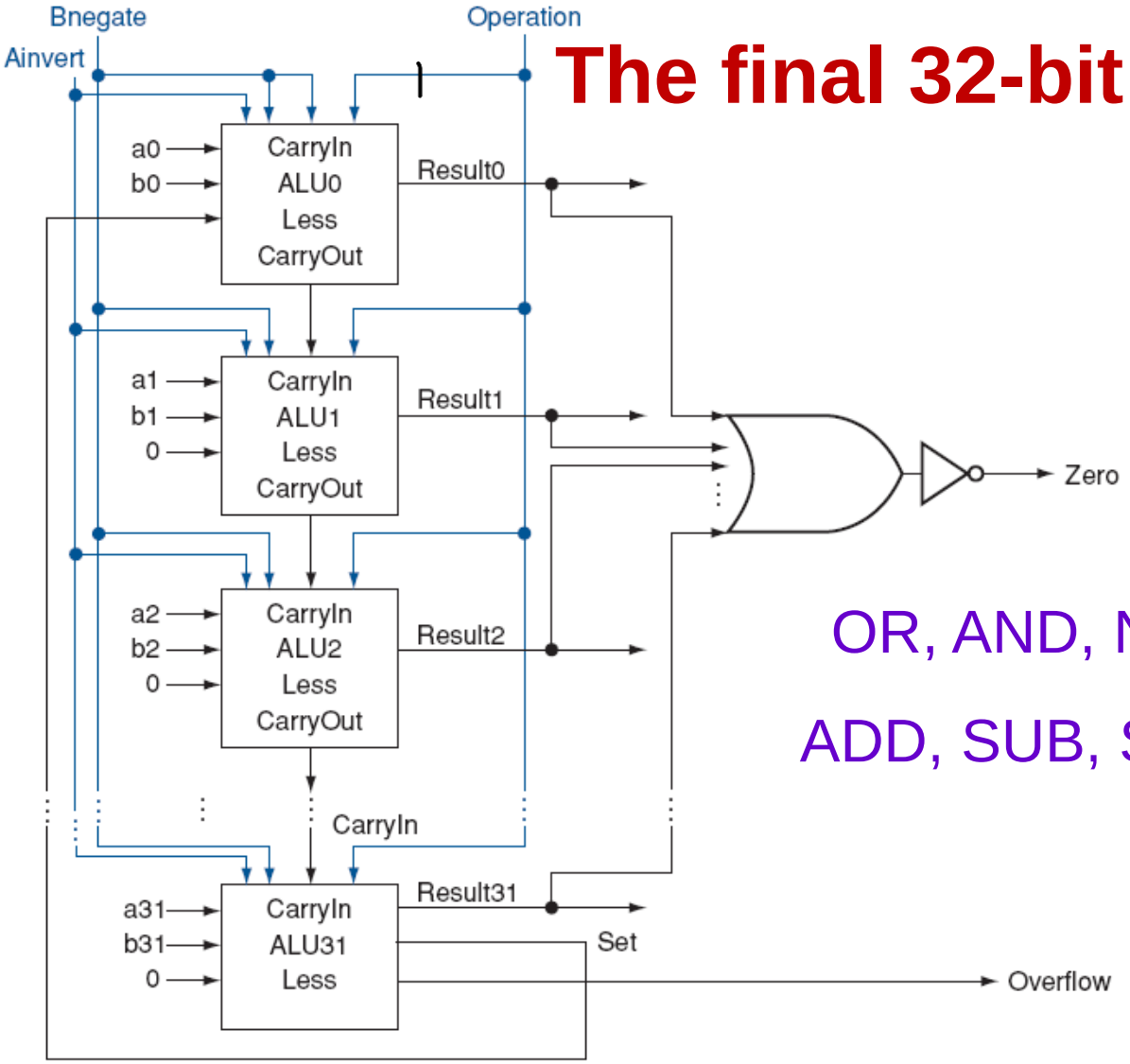
ALU – LSB (הביט הנמוך ביותר)



ALU – MSG (הביט הגבוה ביותר)



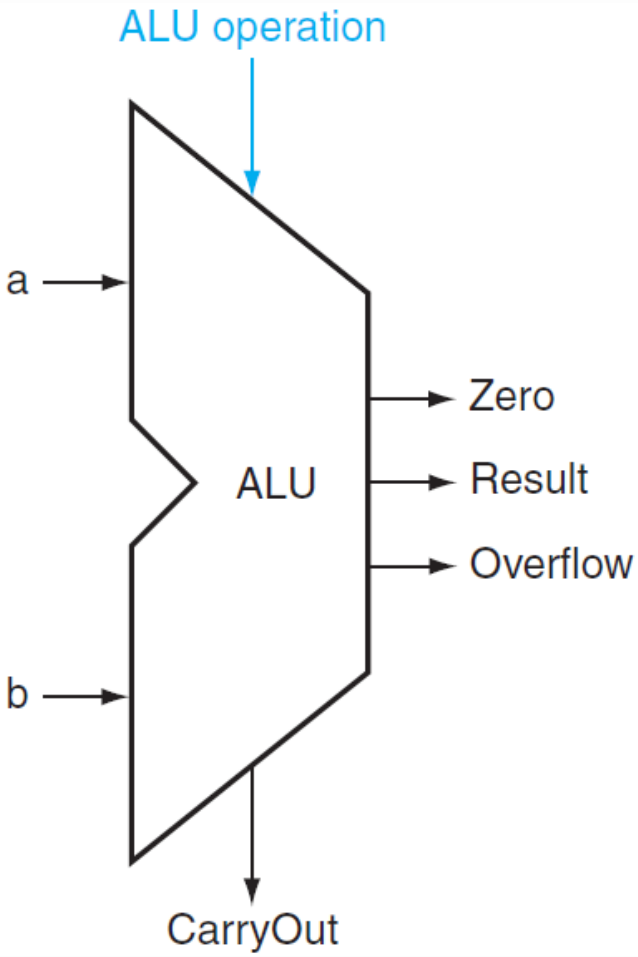
The final 32-bit ALU operations



OR, AND, NOR, NAND,

ADD, SUB, SLT (Z,C,O,S)

ALU control line (4 bits)



יש 4 סיביות קלט פעולה (כפי שראינו בשקף קודם):

- Ainvert – ביט 3
- Bnegate – ביט 2
- Operation – ביטים 0 ו-1

משמעות הפעולות האפשריות:

- **0000** - אל תהפוך את a. אל תהפוך את b ובצע פעולת and.
- **0001** - אל תהפוך את a. אל תהפוך את b ובצע פעולת or.
- **0010** - אל תהפוך את a. אל תהפוך את b ובצע פעולת חיבור.
- **0110** - אל תהפוך את a. הפוך את b ובצע פעולת חיבור.
  כלומר בצע a-b.
- **0111** - אל תהפוך את a. הפוך את b ובצע פעולת slt.
- **1100** - הפוך את a. הפוך את b ובצע פעולת and.
  כלומר בצע nor.
- **1101** - הפוך את a. הפוך את b ובצע פעולת or.
  כלומר בצע nand. (לא מופיע בספר)

אינדיקציות (חיווי/דגל) על פעולות ה ALU

- **Z (Zero) –** נדלק (מקבל 1) אם התוצאה של הפעולה היא 0 (בכל 32 הסיביות)
  - חישבו למה צריך כזו אינדיקציה?
- C (Carry) – זה ה $CarryOut_{31}$ הנשא האחרון (השמאלי), אינדיקציה לגלישה לפי שיטת ייצוג ללא סימן
- O (Overflow) – זה $CarryOut_{31}$ xor $CarryIn_{31}$ אינדיקציה לגלישה לפי משלים ל- 2, נקרא גם גלישה אריתמטית.

  בהמשך נראה שגלישה זו עלולה לגרום להפסקת התוכנית (פסיקה) בפקודות מסוימות
- S (Sign) – זה פשוט סיבית הסימן $Result_{31}$

  לפי שיטת משלים ל- 2, אינדיקציה למספר שלילי הוא 1 ב MSG

סיימנו...

שאלות?

אפקה
המכללה האקדמית
להנדסה בתל אביב