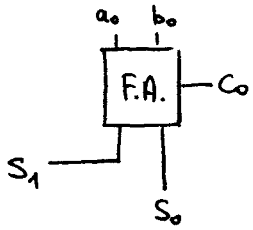
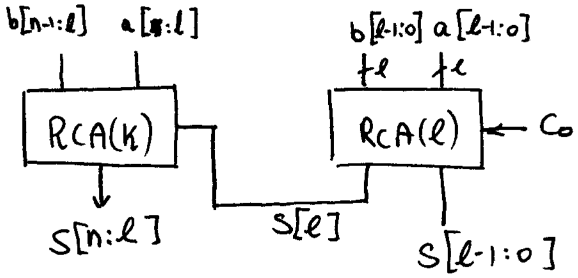
פתרון תרגיל מס' 3.

שאלה 1) ה RCA בנוי מחיבור של Full Adders :



ומחובר בצורה הבאה (RCA(n)) :

$n=k+l$



הוכחת נכונות :

צ"ל : $\langle a\rangle+\langle b\rangle=\langle s\rangle$

נגדיר $a',b'\in\{0,1\}^k$ $\quad a=a'\cdot a''$

$b'',a''\in\{0,1\}^l$ $\quad b=b'\cdot b''$

מתקיים :

$\langle a\rangle=2^l\cdot\langle a'\rangle+\langle a''\rangle$

$\langle b\rangle=2^l\cdot\langle b'\rangle+\langle b''\rangle$

$\langle a\rangle+\langle b\rangle=\langle a''\rangle+\langle b''\rangle+2^l(\langle a'\rangle+\langle b'\rangle)$

ע"י הנחת אינדוקציה

$=\langle s''\rangle+2^l s[l]+2^l(\langle a'\rangle+\langle b'\rangle)$

נשים לב כי $s''\in\{0,1\}^l$ ולכן יש להוסיף את הביט $s[l]$.

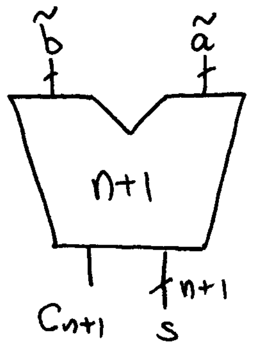
$=\langle s''\rangle+2^l(\langle a'\rangle+\langle b'\rangle+s[l])=$

ושוב ע"י הנחת אינדוקציה

$=\langle s''\rangle+2^l\langle s'\rangle=\underline{\underline{\langle s\rangle}}$

(שאלה 2)

נציע לגלות ביט ovf ע"י מימוש מחבר ל n+1 ביטים.



נכניס למחבר את a ו b
עם Sign extention :

$\tilde{a} = a_n \cdot a$ $\quad a_n = a_{n-1}$

$\tilde{b} = b_n \cdot b$ $\quad b_n = b_{n-1}$

הראנו בכיתה ש $[\tilde{a}] = [a]$

מתי יש overflow ? כאשר $z \notin T_n$.

ראינו בכיתה : $z = [\tilde{a}] + [\tilde{b}] + C_0 \in T_{n+1}$

כאשר $z \in T_n$ אז הביט העליון הוא שמירת הרחבת סימן ולכן $S_n = S_{n-1}$ (אמ"מ)

מכאן ש $ovf = XOR(S_n, S_{n-1})$

* בדרך לספר : $ovf = XOR(C_{n-1}, C_n)$

$S_{n-1} = a_{n-1} \oplus b_{n-1} \oplus C_{n-1}$

$\Rightarrow C_{n-1} = a_{n-1} \oplus b_{n-1} \oplus S_{n-1}$

$\Rightarrow ovf = XOR(a_{n-1}, b_{n-1}, S_{n-1}, C_n)$

(שאלה 3)

צריך להוכיח כי N מקיימת אסוציאטיביות עבור האופרטור * :

$$(a*b)*c = a*(b*c)$$

מקרה I: $a=0$ ⇐ בשני האגפים נקבל 0.

מקרה II: $a=2$ ⇐ בשני האגפים נקבל 2.

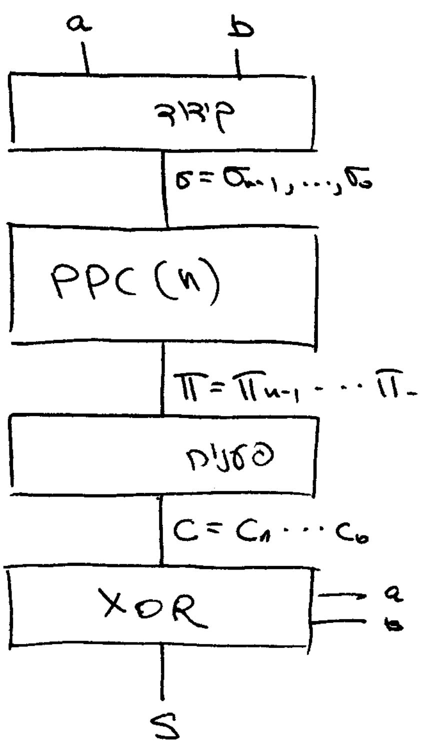
מקרה III: $a=1$: מתקיים $1*x = x$

$$\underbrace{(1*b)}*c = 1*(b*c)$$

$$b*c = b*c$$

יש צורך באסוציאטיביות כיוון שהמעגל ה PPC
משנה את סדר ביצוע המכפלות דורש קיום אסוציאטיביות.

(שאלה 4) איך נראה CLA ?



הערה: נוהגים לקרוא לערכי Σ
($\Sigma = \{0,1,2\}$) את K,P,G :

G (=2) - generate - יוצר carry

P (=1) - propogate - מקדם את carry קיים

K (=0) Kill - "הורג" carry.

כיצד כדאי לבחור את קידוד σ כדי לקבל PPC יעיל
וזול ?

הגדרה I: קידוד אונרי : $k, p, g \in \{0,1\}^3$

כלומר $\Pi_u : \Sigma \to \{0,1\}^3$

$$\Pi_u(0) = (0,0,1)$$

$$\Pi_u(1) = (0,1,0)$$

$$\Pi_u(2) = (1,0,0)$$

כיצד נקבל שער לקידוד הזה :

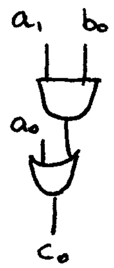
$$(c_2, c_1, c_0) = (a_2, a_1, a_0) * (b_2, b_1, b_0)$$

- מתי $c_0 = 1$? (kill) כאשר $a = kill$

או כאשר $a = propogate, b = kill$:

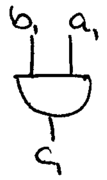
$$c_0 = 1 \Leftrightarrow (a_0 = 1) \vee (a_1 = 1 \ \& \ b_0 = 1)$$

מימוש



- מתי $c_1 = 1$? (propogate) כאשר $a = propogate$, $b = propogate$

$$c_1 = 1 \Leftrightarrow a_1 = 1 \ \& \ b_1 = 1$$

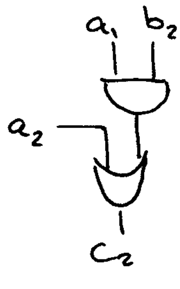


- מתי $c_2 = 1$? (generate) : כאשר $a = generate$ או

כאשר $a = propogate, b = generate$:

$$c_2 = 1 \Leftrightarrow (a_2 = 1) \vee (a_1 = 1 \ \& \ b_2 = 1)$$

מימוש:



סיכום עבור הצעה זו

$$D_u(*) = 2$$

$$cost_u(*) = 3\cdot cost\{AND\} + 2\cdot cost\{OR\}$$

בכיתה פתרנו המשוואה למחיר ועלות של $PPC(n)$
כתלות ב $D(*)$, $C(*)$ לכן נשאר רק להציב.

הצעה II: קידוד בינארי $\qquad \pi_b : \Sigma \to \{0,1\}^2$

נציג קידוד בינארי כך ש $\qquad \langle \pi_b(\sigma)\rangle = \sigma$

כלומר:

$$\pi_b(0) = 0 \quad 0$$

$$\pi_b(1) = 0 \quad 1$$

$$\pi_b(2) = 1 \quad 0$$

הערה: נהוג לקרוא לביט השמאלי g כי רק עבור $g=1$
מקבלים $\sigma=2$ וזה generate, לביט הימני p כי
עבור $p=1$ מקבלים $\sigma=1$ וזה propogate.
עבור $p=0$, $g=0$ זה kill ונשים לב ש $g=1, p=1$
הוא מצב אסור.

אם כן, $\qquad (c_1, c_0) = (a_1, a_0) * (b_1, b_0)$

- מתי $c_0 = 1$ (propogate)? כאשר $a=b=$propogate

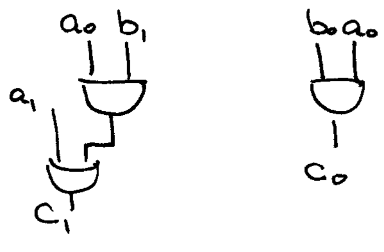
כלומר $\qquad c_0 = 1 \Leftrightarrow a_0 = 1 \ \& \ b_0 = 1$

- מתי $c_1 = 1$ (generate)? כאשר $a_1=$generate

או כאשר $a=$propogate, $b=$generate כלומר,

$$c_1 = 1 \Leftrightarrow (a_1 = 1) \vee (a_0 = 1 \ \& \ b_1 = 1)$$

מימוש:



האם יתכן כי נקבל במוצא $C_0=1$, $C_1=1$?

קל לראות שלא כמו שעבור (a_1,a_0) ו (b_1,b_0)

רק אחד מכל זוג יכול להיות 1 לכן כאשר $C_0=1$

אז בהכרח $C_1=0 \Leftarrow a_1=0, b_1=0 \Leftarrow a_0=1, b_0=1$

וכאשר
$$\left.\begin{array}{l} C_0=0 \Leftarrow a_0=0 \Leftarrow a_1=1 \\ \text{או} \\ C_0=0 \Leftarrow b_0=0 \Leftarrow a_0=1, b_1=1 \end{array}\right\} \Leftarrow C_1=1$$

העלות וההשהיה:

$D_b(*)=2$

$cost_b(*)=2\cdot cost\{AND\}+cost\{OR\}$

הצעה III: קידוד בינרי כאשר כלא אין מצב

אופר אופטי:

	g	p
0	0	0
1	0	1
2	1	0
	1	1

$$(C_1,C_0)=(a_1,a_0)*(b_1,b_0)$$

כלו אם $C_0=1$ זה לא אומר שהוא בהכרח propogate.

נשאיר את המימוש כמו בהצעה II:

$C_0 = a_0 \& b_0$

$C_1 = a_1 \vee (a_0 \& b_1)$

ונניח זה שאם a ו b שניהם מקודדים 1 ו $C_0=1$

אז בהכרח $C_1=1$.

מה התועלת בקידוד שכזה? (לשמירה, ההשהיה והמחיר זהים).

נסתכל על המערכת בכללותה :

~~עבור הקידוד~~, אמרנו כי בשלב הראשון הוקטור a,b מקודדת ל σ .
נסמן את ביטי σ_i ע"י (g_i, p_i) . אזי σ_i מקיים
עבור הקידוד השני :

a_i	b_i	σ_i: g_i	σ_i: p_i
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

$$\rightarrow g_i = AND(a_i, b_i)$$
$$p_i = XOR(a_i, b_i)$$

במוצא ה PPC של π_i מיוצג ע"י שני סיביות (g_{π_i}, p_{π_i})
כאשר $c_{i+1} = 1$ אמ"מ $\pi_i = 2 \Leftarrow g_{\pi_i} = 1$
כלומר, הפענוח מיידי .
לבסוף קבלת ה s_i-ים ע"י $s_i = XOR(a_i, b_i, c_i)$

$$\Rightarrow s_i = XOR(p_i, g_{\pi_{i-1}})$$

אבל, שברוב הטכנולוגיות $D(OR) < D(XOR)$.

נראה מה קורה בקידוד השלישי : כאן קידוד σ_i יכול להתבצע כך :

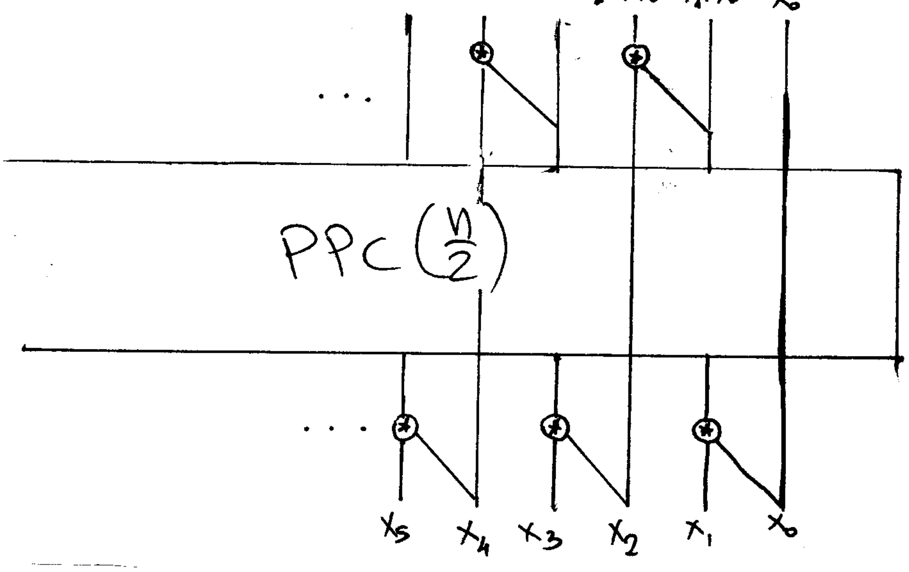
a_i	b_i	σ_i: g_i	σ_i: p_i
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0
		1	1

$$\Rightarrow g_i = AND(a_i, b_i)$$
$$p_i = OR(a_i, b_i) \quad (\text{או } XOR)$$

הפענוח מיידי כמו קודם : $c_{i+1} = 1 \Leftrightarrow g_{\pi_i} = 1$
והחיבור הסופי $S_i = XOR(a_i, b_i, c_i)$ אלא שכעת $XOR(a_i, b_i)$
לא נתון ע"י P_i אלא יש לחשב בנפרד , חישוב זה יכול
להתבצע במקביל לחישוב העיקרי ולכן אינו מוסיף השהייה .

כלומר, שילמנו על $XOR + OR$ כדי להוציא את ה XOR
מהמסלול הקריטי ולהחליפו ב OR .

(5)



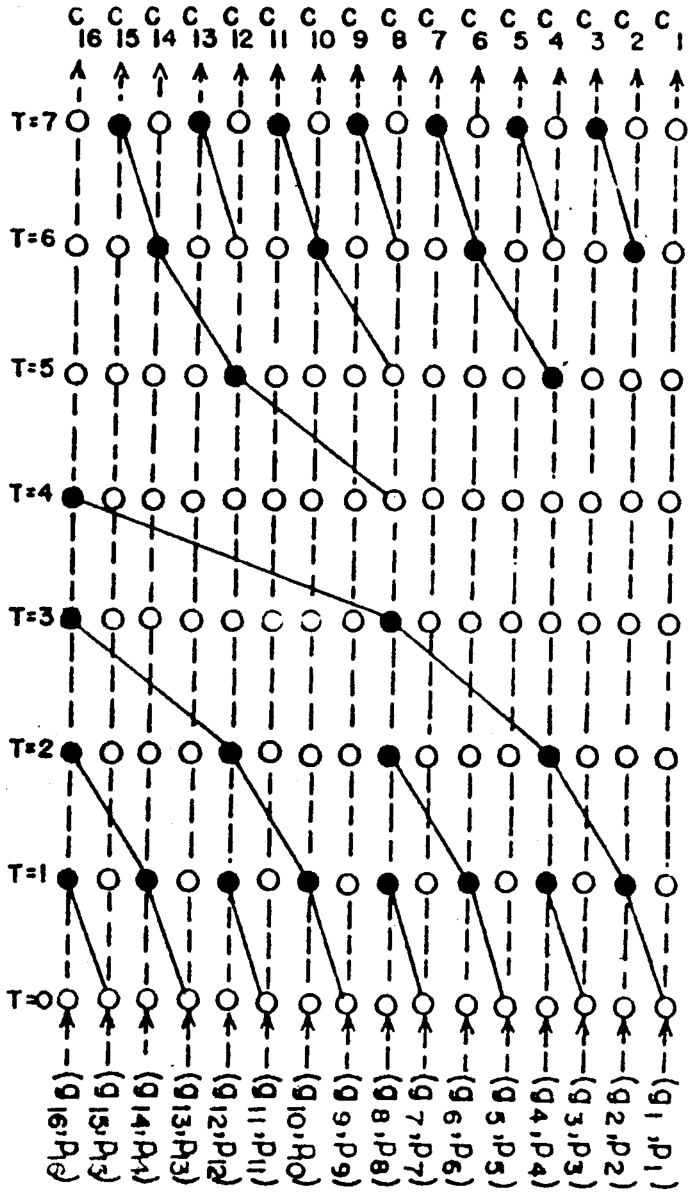
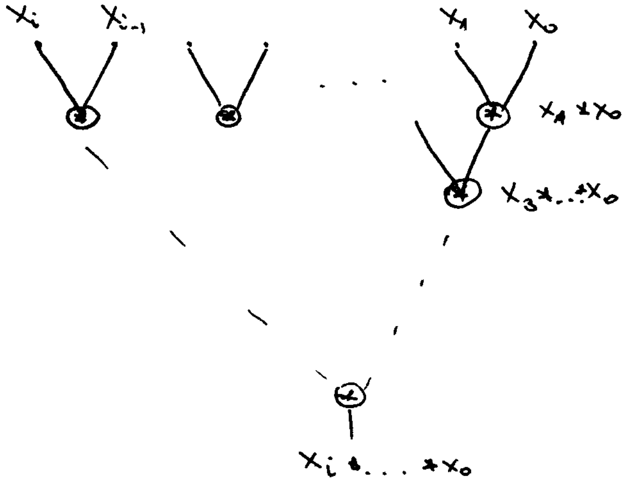


Fig. 5. Computation of all the carries for $n = 16$.

(שאלה 6)

מימוש כל prefix : $x_i * \dots * x_0$ בנפרד

ע"י עץ בינרי



נקרא לזה $Prefix_i$

השהיה :

$$D(\text{מעגל}) = \lceil \lg n \rceil \, D \cdot D(*)$$

$$C(\text{מעגל}) = \sum_{i=1}^{n} C(Prefix_i) = \sum_{i=1}^{n} (i-1)\, C(*) = C(*)\, \Omega(n^2)$$