מיתוג ומערכות ספרתיות

Reference book:
- "Computer engineering – hardware design" by Morris Mano, Prentice Hall
- "Digital Design" by Morris Mano Prentice Hall

נושאים:

• קודים ומערכות בינאריות	Binary Numbers and Codes
• מעגלים ספרתיים – אלגברה בוליאנית	Digital Circuits – Boolean Algebra
• פישוט פונקציות בוליאניות	Analysis of boolean systems
• לוגיקה צירופית	Decoders, Encoders, Mutiplexers
• לוגיקה סדרתית	Sequential Logic Filp-Flops (דלגלגים)
• אוגרים ומונים	Registers and counters
• יחידות הזיכרון	Memory Units
• מבנה אוגרים ופעולות מכונה	Register transfer and computer organization
• יחידת הבקרה ופעולותיה	Control until
• פקודות מכונה ושיטות מיעון	Computer instructions and addressing modes

מבנה המחשב

מיתוג ומערכות ספרתיות

- אתר הקורס:
  www.cs.huji.ac.il/~dlocs
- קיימים שני news-groups
- ספרים בספרית הרמן (Digital Design ,Morris Mano)
- מרצה: עידו ברגמן.
- שעות קבלה: ראשון 10:00-12:00 (Ross 22).
- מתרגלים: תמיר חזן
  ליביא שפיגלמן
- שני תרגילי סימולציה, הראשון מגן 5% והשני חובה 10%
- תרגילים תאורטים מגן 10% – ראו אתר הקורס

ייצוג מספרים במחשב

מספר מיוצג בבסיס r:

$$\ldots a_5 a_4 a_3 a_2 a_1 a_0 . a_{-1} a_{-2} a_{-3} a_{-4} \ldots$$

$a_i \in \{0,1,\ldots,r-1\}$ כאשר לכל i

המרה לייצוג בבסיס 10:

$$\sum_{i=0}^{n} a_i r^i + \sum_{i=1}^{k} a_{-i} r^{-i} = \sum_{i=-k}^{n} a_i r^i =$$

$$= a_n r^n + a_{n-1} r^{n-1} + \ldots + a_1 r^1 + a_0 + a_{-1}\frac{1}{r} + a_{-2}\frac{1}{r^2} + \ldots a_{-k}\frac{1}{r^k}$$

דוגמא:

* $(201.21)_3 = 2\cdot 3^2 + 1\cdot 3^0 + 2\cdot 3^{-1} + 1\cdot 3^{-2} = 18 + 1 + \frac{2}{3} + \frac{1}{9} = (19\frac{7}{9})_{10}$

* $(10110.011111111\ldots)_2 = \underbrace{2^4 + 2^2 + 2^1}_{=22} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} + \ldots \rightarrow (22.5)_{10}$

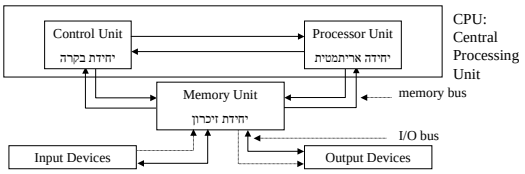
  10110.100000000…

  $(10110.1) = (22.5)_{10}$

מבנה מחשב – סכימה כללית:

מודל פון – ניימן: פקודות ונתונים מאוחסנים באותה יחידת זיכרון.

מודל אוניברסלי: מכונת טיורינג אוניברסלית (Turing Machine)



- הייצוג הפנימי במחשב וכן הפעולות האריתמטיות והלוגיות הנן כולן בינאריות.
- המרות מבסיס בינארי לבסיס אחר (בדר"כ עשרוני) הנן לצורך הצגת תוצאות.
- בסיסים נוספים לצורך הצגה שהנם שימושיים:

Octal (8), Hexadecimal (16)

↓ ↓

0…7 0…9ABCDEF

- המעבר מבינארי לאוקטל או הקסדצימל הינו פשוט ומתבצע ע"י קיבוץ:

$$\underbrace{100}_{4}\,\underbrace{010}_{2}\,\underbrace{111}_{7}.\underbrace{001}_{1}\,\underbrace{000}_{0}\,\underbrace{110}_{6}\,\underbrace{111}_{7})_8$$

$$\underbrace{0011}_{3}\,\underbrace{1001}_{9}\,\underbrace{1100}_{C}.\underbrace{0001}_{1}\,\underbrace{1000}_{8}\,\underbrace{1010}_{A})_{16}$$

מעבר בין בסיסים:

יהי X מספר להמרה מבסיס 10 לבסיס r. נבצע חילוקים מתמשכים ב - r:

$X / r \rightarrow X^{(1)}, S^{(1)}$ שארית

$X^{(1)}/r \rightarrow X^{(2)}, S^{(2)}$ שארית

$X^{(2)}/r \rightarrow X^{(3)}, S^{(3)}$ שארית

התוצאה: $(S^{(n)}S^{(n-1)}\ldots S^{(2)}S^{(1)})_r$

דוגמא: $(35)_{10} \Rightarrow (?)_2$

$35/2 \Rightarrow (17,1)$
$17/2 \Rightarrow (8,1)$
$8/2 \Rightarrow (4,0)$
$4/2 \Rightarrow (2,0)$
$2/2 \Rightarrow (1,0)$
$1/2 \Rightarrow (0,1)$

התוצאה $(100011)_2$

חיבור מספרים בינאריים:

$a_i \in \{0,1\}$ $b_i \in \{0,1\}$

$o_i \in \{0,1\}$ $c_i \in \{0,1\}$ +

$a_{n-1} \ldots a_3 a_2 a_1 a_0$

$b_{n-1} \ldots b_3 b_2 b_1 b_0$

נשא סופי C_n C_2 C_1

"O_n" $o_{n-1} \ldots o_3 o_2 o_1 o_0$

$O_i = (a_i + b_i + c_{i-1}) \bmod 2$

$$C_i = \begin{cases} 1 & a_i + b_i + c_{i-1} \geq 2 \text{ אם} \\ 0 & \text{אחרת} \end{cases}$$

- פעולת החיבור ניתנת למימוש פשוט ע"י מעגל ספרתי.

חיבור מספרים:

חיבור בבסיס r:

$\ldots a_3 a_2 a_1$

$+$

$\ldots b_3 b_2 b_1$

C_2 C_1

$\ldots o_3 o_2 o_1$

$O_1 = (a_1 + b_1) \bmod r$

$C_1 = 1$ iff $a_1 + b_1 \geq r$ (0 otherwise)

תזכורת: $a_i \in \{0,1,\ldots,r-1\}$

$b_i \in \{0,1,\ldots,r-1\}$

$O_i = (a_i + b_i + c_{i-1}) \bmod r$

$C_i = [(a_i + b_i + c_{i-1}) \overset{?}{\geq} r]$

ייצוג ע"י המשלים ל-2: (2's Complement)

זהו יצוג של מספרים שלמים בני n ביטים. מוסיפים "ספרת סימן" כך שיש לנו n+1 ספרות סה"כ.

- מספר חיובי ← ספרת הסימן 0.
- מספר שלילי ← ספרת הסימן 1.

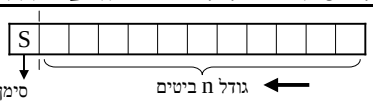
דוגמא: $n = 2$ $n + 1 = 3$ $N = 2^3 = 8$

000 :0	100 :-4	(4=8-4)
001 :1	111 :-1	(7=8-1)
010 :2	110 :-2	(6=8-2)
011 :3	101 :-3	(5=8-3)

בייצוג בשיטת 2's Comp. למספר 2^n אין ייצוג חיובי.

טווח המספרים הינו: $-2^n, -2^n+1, \ldots, -1, 0, 1, 2, \ldots, 2^n-1$

יצוגם של מספרים שליליים – יצוג ע"י גודל וסימן



S=0 ⇐ המספר חיובי

S=1 ⇐ המספר שלילי

דוגמא: n=2

חיוביים:	שליליים:
000 :0	100 :-0
001 :1	101 :-1
010 :2	110 :-2
011 :3	111 :-3

- בשיטה זו יש יצוג כפול ל – 0.
- קשה לבצע פעולות אריתמטיות.
- משתמשים בייצוג זה בייצוג מספרים בנקודה צפה (floating point) לצורך הצגת המונה (exponent).

חישוב המשלים ל – 2:

הגדרה: המשלים ל-2 של X הוא $2^{n+1} - X$.

חישוב המשלים:

א. התקדם מימין והשאר אפסים ללא שינוי עד לספרה הראשונה שהיא "1".

ב. השאר את ה- "1" הראשון ללא שינוי.

ג. החלף כל ספרה משמאל ל- "1" הראשון ב- NOT שלה.

0 1 1 0 1 1 0 0

1 0 0 1 0 1 0 0

משלים ל –2 - מבנה אלגברי:

- n ביטים.
- מוסיפים ביט סימן.
- X חיובי ⇐ הייצוג הוא (0, X)
- X שלילי ⇐ הייצוג הוא $(2^{n+1} - |X|)$
- הגדרה: המשלים ל-2 של המספר X הוא $2^{n+1} - X$.

הצרוף $\underbrace{a_n a_{n-1} a_{n-2} \ldots a_2 a_1 a_0}_{n+1 \text{ ביטים}}$ מייצג את המספר: $X = \sum_{i=0}^{n-1} a_i 2^i - a_n 2^n$

- $a_n = 0$ ⇐ X חיובי, $X = \sum_{i=0}^{n-1} a_i 2^i$
- $a_n = 1$ ⇐ X שלילי, $X = \sum_{i=0}^{n-1} a_i 2^i - 2^n$

דוגמא 1: n=3. רוצים לחשב 5 – 3. נזדקק ל - n+1 = 4 סיביות.

$3 = 0011_2$ $(-5) = 2^4 - 5 = 0\,1\,0\,1_2$

$1\,0\,1\,1_2$

$$\begin{array}{r} 0011 \\ +\,1011 \\ \hline 1110 \\ -2 \equiv -0010_2 \end{array}$$

שלילי

2's Comp. of:

דוגמא 2: n=3. רוצים לחשב 3 – 5. נזדקק ל - n+1 = 4 סיביות.

$5 = 0101_2$ $(-3) = 2^4 - 3 = 0\,0\,1\,1_2$

$1\,1\,0\,1_2$

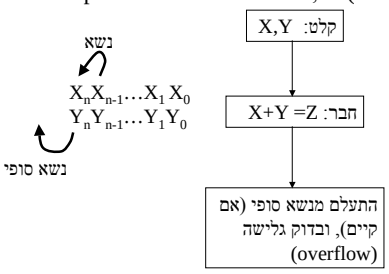
$$\begin{array}{r} 0101 \\ +\,1101 \\ \hline 0010 \end{array}$$

"1" נשא סופי- התעלם.

• מתעלמים מהנשא הסופי. הוא חשוב בבדיקה של גלישה (overflow) שתיתכן כאשר מחברים שני חיוביים (שליליים) והתוצאה גדולה (קטנה) מדי לייצוג ב– n ביטים.

חיבור בשיטת משלים ל –2:

קלט: X,Y מספרים בינאריים שלמים (חיוביים או שליליים) בעלי n ספרות וספרת סימן (n+1). X,Y מיוצגים בשיטת 2's Complement



אם לא היתה גלישה, הפלט הוא תוצאת החיבור, מיוצגת ב- 2's Comp.

נכונות האלגוריתם- המשך:

• מניחים כי אין overflow.

ג. X אי שלילי ו- Y שלילי, ונניח ש- X < |Y|:

$$X + N - |Y| = N + (X + Y) = N - |X+Y|$$

ד. X, Y שליליים:

$$(N - |X|) + (N - |Y|) = N + [N - (|X| + |Y|)]$$
$$= N + (N - |X + Y|)$$

נשא סופי- התעלם

נכונות האלגוריתם:

• מניחים כי אין overflow.

חלוקה למקרים:

א. X אי שלילי ו- Y אי שלילי:
התוצאה אי שלילית והנכונות ברורה.

ב. X אי שלילי ו- Y שלילי, ונניח ש- X >= |Y|:

$$X + N - |Y| = (X + Y) + 2^{n+1}$$

נשא סופי- התעלם

חיבור בשיטת משלים ל – 1

קלט: X, Y מספרים בגודל של n סיביות, מיוצגים בשיטת משלים ל –1 בעזרת n+1 סיביות.

פלט: X + Y, התוצאה ב – 1's comp.

ביצוע: א. חבר X + Y בעזרת האלגוריתם הסטנדרטי.
ב. אם יש נשא סופי חבר אותו אל התוצאה (נשא מעגלי)

דוגמאות עבור n=3:

2 - 4

$$\begin{array}{r} 0010 \\ -0100 \quad +\,1011 \\ \hline 1101 \end{array}$$

שלילי

-0010 = -2

5 - 3

$$\begin{array}{r} 0101 \\ -0011 \quad +\,1100 \\ \hline 1\;0001 \\ +\;\;1 \\ \hline 0010 \end{array}$$

ייצוג בשיטת המשלים ל – 1:

כמו קודם, נדון במספרים בני n סיביות, סיבית n+1 היא סימן.

דוגמא: n=2

000 :0	111 :-0	
001 :1	110 :-1	(7-1 = 6)
010 :2	101 :-2	(7-2 = 5)
011 :3	100 :-3	(7-3 = 4)

אלגוריתם לחישוב משלים ל – 1: הפוך כל ספרה לחוד 0←1, 1←0.
דרך נוספת לחישוב המשלים ל –2: חשב משלים ל –1 וחבר 1+ לתוצאה

סמנטיקה: $X = \sum_{i=0}^{n-1} a_i 2^i - a_n(2^n - 1)$

• ל "0" ייצוג כפול.
• טווח המספרים בני n סיביות: $-(2^n-1), -(2^n-2), \ldots, -0, +0, 1, 2, \ldots, 2^n-1$

השוואה: 1's Complement vs 2's Complement

- ניתן לחשב משלים ל –2 ע"י ביצוע משלים ל –1 וחיבור של 1+ לתוצאה.
  (פעולת השלמה מורכבת יותר עבור משלים ל – 2)
- ביצוע החיבור פשוט יותר עבור משלים ל –2.
  (נשא סופי אך לא מעגלי)
- ייצוג יחיד ל – 0 במשלים ל– 2.
  ייצוג כפול ל – 0 במשלים ל– 1.
- טווח מספרים:

משלים ל – 2: $-2^{n-1}, -2^{n-1}+1, \ldots, -1, 0, 1, \ldots, 2^{n-1}-1$

משלים ל – 1: $-2^{n-1}+1, \ldots, -1, 0, 1, \ldots, 2^{n-1}-1$

השימוש במשלים ל-2 יותר נפוץ, בעיקר בגלל שהחבור פשוט יותר כי לא צריכים לטפל בנשא.

חיבור בעזרת המשלים ל–1 — נכונות:

- גם כאן, כמו הוכחת הנכונות עבור המשלים ל- 2, ההוכחה נעשית ע"י חלוקה למקרים.
- ההוכחה נשארת כתרגיל לסטודנט.

בדיקת גלישה – Overflow:

- מספרים בני n -1 ספרות, ביט הסימן n.

a_n	a_{n-1}	•••	a_1	a_0

- חיבור שני חיובים:

n	n-1	
0	1	•••
+ 0	1	•••
= 1	0	•••

אין נשא סופי (n+1), יש נשא ל - n

- חיבור שני שליליים:

1	0	0	•••
+ 1	1	0	•••
= 1 0	1	•	•••

יש נשא סופי (n+1), אין נשא ל – n

Overflow condition: $C_n \oplus C_{n-1} = 1$

נשא סופי לעומת גלישה:

- נשא Carry: הינו מצביע על תופעה תקינה ומשמש לקביעת הסימן.
- גלישה Overflow: מצביע על פעולה לא תקינה כתוצאה מחיבור שני מספרים גדולים יותר מדי.

דוגמא: עובדים עם מספרים בני 3 ביטים בשיטת משלים ל- 2. לכן נקצה 4 ביטים למספר.

- Carry: 7 - 3

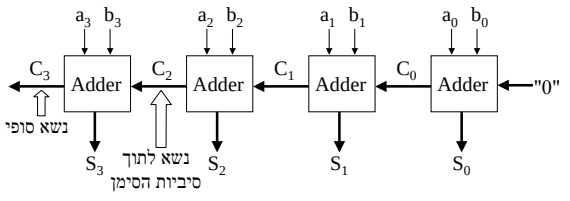
7: 0 1 1 1
3: 0 0 1 1
-3: 1 1 0 1
+
4: 1 0 1 0 0 נשא סופי

- Overflow: 5 + 7

7: 0 1 1 1
+
5: 0 1 0 1
-4: 1 1 0 0

בדוגמה זו אין נשא סופי. התוצאה התקלקלה בגלל גלישה.

בדיקת גלישה – מימוש:



Overflow iff $C_n \oplus C_{n-1} = 1$

במצב תקין (ז"א ללא overflow):

- אם $\bar{a}$ ו - $\bar{b}$ חיוביים לא אמור להיות נשא לתוך המסכם האחרון, ולא יכול להיות נשא סופי.
- אם $\bar{a}$ ו - $\bar{b}$ שליליים אמור להיות נשא לתוך המסכם האחרון, וחייב להיות נשא סופי.

קודים מתקני שגיאות

← טעויות אחסון כתוצאה ממתחים אקראיים.

← שינוי של ביט יחיד יכול לגרום לתוכנית שלמה לא לעבוד.

← קודים מתקני שגיאות (או מגלי שגיאות)

Error Correcting Codes:

Linear Codes, Hamming, Reed – Solomon …

Error Detecting Codes:

Checksum, Parity Checking …

Example: הוסף ביט לכל 4 ביטים. הביט החדש יספור זוגיות

$S = (X_1 + X_2 + X_3 + X_4) \bmod 2$

X_1	X_2	X_3	X_4	S
1	0	0	1	0
1	1	1	0	1

קודים וייצוגים

ייצוג של ערכים לא בינאריים מבוצע ע"י שמירת מחרוזות של ביטים.

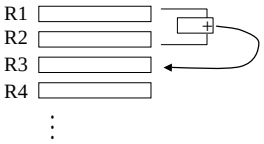
דוגמא: BCD – Binary Coded Decimal

0	0001	…	7	0111
1	0001	…	8	1000
2	0010	…	9	1001

קודים אלפאנומרים: ASCII, EBCDIC (IBM)

A	1000001	0	0110000
B	1000010	1	0110001
C	1000011	⋮	
⋮		9	0111001
Z	1011010		
$	0100100		
.	0101110		
(	0101000		
=	0111101		

אוגרים ואחסון ביטים (בינארי)

(signed)	char	8 bit
	short	16 bit
	int	32 bit
	long int	64 bit

- אוגרים (Registers) הינם אוסף של תאי זיכרון מהירים מאוד (זמן גישה כתדר שעון המחשב) שבעזרתם מבצעים פעולות אריתמטיות.
- מחשב מכיל ALU המבצע חיבורים מהירים תוך כדי בדיקת נכונות התוצאה (overflow)