בסיסים והמרות

טווח היצוג של מספר שלם אי-שלילי בן n ספרות בבסיס b נע בין 0 לבין $\left(b^n-1\right)$.

בהינתן מספר בבסיס $2\le b$, המיוצג ע"י הביטוי $\left(d_n d_{n-1}\ldots d_1 d_0 . d_{-1} d_{-2}\ldots d_{-k+1} d_{-k}\right)_b$ כאשר $\forall i: d_i \in \{0,1,\ldots,b-1\}$, המרתו לייצוג עשרוני (בבסיס הדצימלי, בסיס-10) תיעשה בפשטות ע"י: $\left(\sum_{i=0}^{n} d_i \cdot b^i\right)+\left(\sum_{i=1}^{k} d_{-i} \cdot b^{-i}\right)=\sum_{i=-k}^{n} d_i \cdot b^i$.

המרה בכיוון ההפוך תיעשה באופן הבא: בכל שלב $0\le i$ נחלק את המספר העשרוני A ב-b, כאשר d_i שווה לשארית חלוקה זו. להלן תיאור האלגוריתם ב-pseudocode:

$i \leftarrow 0$
while $A > 0$
  $d_i \leftarrow A$ **mod** b
  $A \leftarrow \left\lfloor \frac{A}{b} \right\rfloor$
  $i \leftarrow i+1$

כאשר ממירים מספר מן הבסיס הבינארי (בסיס-2) לבסיסים של חזקות אחרות של 2, כגון הבסיס האוקטלי (בסיס-8) או ההקסדצימלי (בסיס-16), נעשה זאת בפשטות ע"י קיבוץ k ספרות בינאריות (binary digit - bit) לפי חזקת הבסיס 2^k:

$$\left(101100111000\right)_2 = \underbrace{101}_{5}\ \underbrace{100}_{4}\ \underbrace{111}_{7}\ \underbrace{000}_{0} = \left(5470\right)_8 \qquad 8=2^3$$

$$= \underbrace{1011}_{\mathrm{B}}\ \underbrace{0011}_{3}\ \underbrace{1000}_{8} = \left(\mathrm{B38}\right)_{16} \qquad 16=2^4$$

ובאותו אופן בכיוון ההפוך, נהפוך כל ספרה בבסיס 2^k ל-k-יה מתאימה של ביטים.

קודים

http://www.cs.huji.ac.il/course/2004/dlocs/ido_lectures/1+2.ppt
http://www.cs.huji.ac.il/course/2004/dlocs/handouts/04/class.pdf
http://www.cs.technion.ac.il/~cs234145/new_tutorials/t2.pdf

יצוג בינארי של מספרים שלמים - בעזרת ביט סימן (Sign and Magnitude)

בהינתן מספר בינארי $d_n d_{n-1}\ldots d_1 d_0$, הביט השמאלי ביותר d_n (most significant bit - msb) מציין את סימנו של המספר והשאר מייצגים את ערכו המוחלט (magnitude), כך שערכו העשרוני יהיה: $A=\left(-1\right)^{d_n}\sum_{i=0}^{n-1}2^i d_i$ (כלומר 0 חיובי, 1 שלילי).

- קל לממש את היצוג, אך מסובך לממש ביצוע פעולות אריתמטיות
- בשיטה זו קיים יצוג כפול למספר 0 $\left(000\ldots=+0=-0=100\ldots\right)$
- סקירת ערכים עבור byte אחד (8 ביט): $0,1,\ldots,126,127,0,-1,\ldots,-126,-127$
- טווח היצוג עבור n ספרות + ביט סימן (כנ"ל): $-2^n < A < 2^n$

יצוג שלמים בעזרת משלים ל-1 (1's Complement)

(ידוע גם כ-"diminished radix complement" עבור הבסיס הבינארי).

בדומה לשיטת "Sign and Magnitude", גם כאן הביט השמאלי (msb) מציין את סימנו של המספר (ואף באותו אופן). השוני בין השיטות נובע מכך שמספרים שליליים מבוטאים ע"י היפוך כל הביטים.

חישוב הערך העשרוני עפ"י הערך הבינארי: $\left(d_n d_{n-1} \ldots d_1 d_0\right)_2 = -d_n\left(2^n - 1\right) + \sum_{i=0}^{n-1} 2^i d_i$

- קל יחסית לממש ביצוע פעולות אריתמטיות
- קיים יצוג כפול למספר 0 $\left(000\ldots = +0 = -0 = 111\ldots\right)$
- טווח היצוג עבור n ספרות + ביט סימן (כנ"ל): $-2^n < N < 2^n$
- סקירת ערכים עבור בייט אחד: $0,1,\ldots,126,127,-127,-126,\ldots,-1,0$

חיבור (וחיסור כחיבור של הנגדי):

- כל ביט מחושב בנפרד, מה-lsb הימני עד ל-msb השמאלי, עם הנשא מן החישוב הקודם
- ה-msb הוא "נשא מעגלי" (cyclic carry) - אם כתוצאה מן החיבור הוא שווה ל-1, מוחקים אותו ומחברים אותו לסכום (כלומר ה-lsb הימני ביותר מחליף את ערכו)
- אם מחברים 2 מספרים שווי-סימן וסימן התוצאה מתהפך, נאמר שהיתה "גלישה" (overflow)
- קיום נשא הוא מצב תקין, קיום גלישה מסמן חישוב לא חוקי מיסודו
- בדיקת גלישה: XOR בין שני הנשאים האחרונים (השמאליים)

יצוג שלמים בעזרת משלים ל-2 (2's Complement)

(ידוע גם כ-"radix complement" עבור הבסיס הבינארי).

בדומה לשיטת "1's Complement", גם כאן הביט השמאלי (msb) מציין את סימנו של המספר (ואף באותו אופן). השוני בין השיטות נובע מכך שמספרים שליליים מבוטאים ע"י המשלים + 1.

חישוב הערך העשרוני עפ"י הערך הבינארי: $\left(d_n d_{n-1} \ldots d_1 d_0\right)_2 = -2^n d_n + \sum_{i=0}^{n-1} 2^i d_i$
דרך נוספת לחישוב המשלים ל-2: ע"י חיבור 1 למשלים ל-1

- קיים יצוג יחיד למספר 0
- קל יחסית לממש ביצוע פעולות אריתמטיות
- טווח היצוג עבור n ספרות + ביט סימן (כנ"ל): $-2^n \le N < 2^n$
- סקירת ערכים עבור בייט אחד: $0,1,\ldots,126,127,-128,-127,\ldots,-2,-1$

חיבור/חיסור: באותו אופן, אבל מוחקים את הנשא ולא משתמשים בו

השוואה בין שני יצוגי המשלימים:

- פעולת ההשלמה מעט יותר מורכבת עבור 2
- חישוב החיבור פשוט יותר עבור 2 (נשא סופי אך לא מעגלי)
- טווח היצוג של 2 גדול במספר שלילי אחד מזה של 1
- יצוג יחיד לאפס ב-2, לעומת יצוג כפול לאפס ב-1

יצוג שברים בשיטת הנקודה הצפה (Floating Point)

דוגמא להבדל בין precision (רזולוציה) ל-accuracy (דיוק): המספר 3.14 הוא בעל רזולוציה של 10^{-2} (המרחק בינו לבין שכניו הקרובים ביותר), ואילו למספר 3.24159 יש רזולוציה טובה יותר של 10^{-5}, אך למרות זאת הדיוק של הראשון כיצוג של π טוב יותר $\left(|\pi - 3.14| < |\pi - 3.24159|\right)$.

טווח היצוג תלוי בבסיס (radix), בכמות הספרות, ובמיקום הנקודה. כך שניתן לראות בבירור שקיים יחס הפוך בין הרזולוציה לטווח היצוג.

- יצוג בשיטת הנקודה הצפה בבסיס הבינארי מחלק את הביטים ל-mantissa או significand מימין ו-exponent משמאל, כך שהערך הדצימלי של המספר מחושב ע"י: $x = M \times 2^E$.
- המנטיסה היא רציונלי מן הצורה $\pm d.ddd\ldots$, כלומר fixed-point ביצוג sign & magnitude. היא "מנורמלת" אם מזניחים את החלק השלם שלה (הספרה לפני הנקודה העשרונית) ומניחים שהשבר הנותר מחובר לערך יסוד כלשהו מוסכם מראש (לדוגמא: 1 או 0.5). כך שבכל מקרה המנטיסה חסומה בתחום ערכים מסויים, השאלה היא רק מהו בדיוק.
- האקספוננט הוא "מוטה" (biased), כלומר בעל הטיה B, אם מחסרים ביצוג העשרוני את B מ-E. לרוב משתמשים בערכי הטיה $B = 2^{|E|-1}$ או $B = 2^{|E|-1} - 1$ ($|E|$ מציין את מספר הביטים של E).
- לפי תקן IEEE: ביטים בסידור SEM מייצגים את $(-1)^S \times 2^{E-B} \times 1.M$, כאשר $B = 2^{|E|-1} - 1$. המספר 0 מיוצג ע"י $E = F = 0$, והערך המקסימלי של E מייצג NaN $\left(\frac{0}{0}\right)$ או Inf (∞).

אלגוריתם חיבור:

1. זהה את המספר בעל האקספוננט הקטן יותר
2. המר את יצוג המספר המזוהה, כך שהאקספוננט החדש יהיה שווה לזה של המספר השני (ע"י חלוקת המנטיסה בערך מתאים)
3. חבר את המנטיסות
4. נרמל את התוצאה, מחק ביטים מימין בעת הצורך

יתרונות השיטה: טווח דינמי ורחב, פעולות כפל/חילוק פשוטות
חסרונות השיטה: רזולוציה לא אחידה, פעולות חיבור/חיסור מסובכות, rounding errors, אין אסוציאטיביות ודיסטריבוטיביות

http://www.cs.huji.ac.il/course/2004/dlocs/ido_lectures/1+2.ppt
http://www.cs.huji.ac.il/course/2004/dlocs/handouts/01/cl01.pdf
http://www.cs.huji.ac.il/course/2004/dlocs/handouts/02/cl02a.pdf

http://www.cs.technion.ac.il/~cs234145/new_tutorials/t1.pdf
http://www.cs.technion.ac.il/~cs234145/new_tutorials/t2.pdf

http://en.wikipedia.org/wiki/Binary_numeral_system
http://en.wikipedia.org/wiki/Signed_number_representations
http://en.wikipedia.org/wiki/Two%27s_complement
http://en.wikipedia.org/wiki/Floating_point
http://en.wikipedia.org/wiki/IEEE_Floating_Point_Standard

אלגברה בוליאנית

סיכום זהויות (אקסיומות + משפטי יסוד):

איברי יחידה (אדישות):	$x \cdot 1 = x$	$x + 0 = x$
חילוף (קומוטטיביות):	$xy = yx$	$x + y = y + x$
פילוג (דיסטריבוטיביות):	$x + yz = (x + y)(x + z)$	$x(y + z) = xy + xz$
השלמה:	$x \cdot x' = 0$	$x + x' = 1$
עצמיות (אידמפוטנטיות):	$x \cdot x = x$	$x + x = x$
שליטה:	$x \cdot 0 = 0$	$x + 1 = 1$
קיבוץ (אסוציאטיביות):	$(xy)z = x(yz)$	$(x + y) + z = x + (y + z)$
בליעה 1 (צמצום):	$x(x + y) = x$	$x + xy = x$
בליעה 2:	$x(x' + y) = xy$	$x + x'y = x + y$
דה-מורגן:	$(xy)' = x' + y'$	$(x + y)' = x' \cdot y'$

עקרון הדואליות: $\cdot \leftrightarrow +$, $1 \leftrightarrow 0$
קדימות אופרטורים: ( )→Not→And→Or

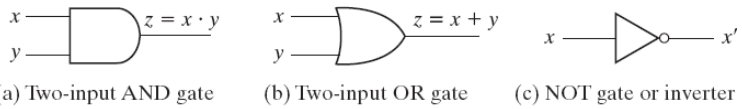
דוגמאות למערכות שלמות: $\{\text{NOR}\} = \{\neg, \vee\}, \{\text{NAND}\} = \{\neg, \wedge\}, \{\neg, , \wedge \vee\}$

http://www.cs.huji.ac.il/course/2004/dlocs/ido_lectures/3a.ppt
http://www.cs.huji.ac.il/course/2004/dlocs/handouts/02/cl02b.pdf

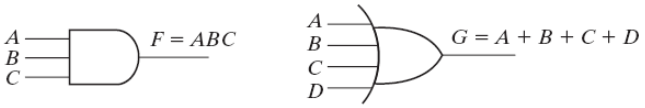
http://www.cs.technion.ac.il/~cs234145/lectures/fromeefor2004/lecture1.pdf
http://www.cs.technion.ac.il/~cs234145/new_tutorials/t4.pdf

http://en.wikibooks.org/wiki/Boolean_Algebra
http://en.wikipedia.org/wiki/Logic_gate

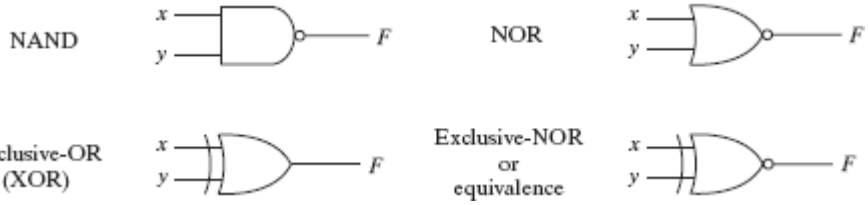
שערים לוגיים



(a) Two-input AND gate (b) Two-input OR gate (c) NOT gate or inverter



(a) Three-input AND gate (b) Four-input OR gate

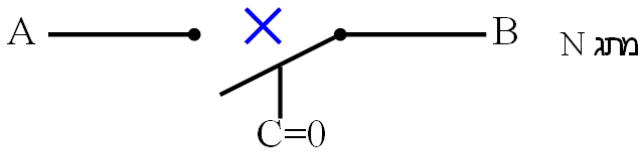


x	y	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
		0	$\cdot$	/	x	/	y	$\oplus$	+	$\downarrow$	$\odot$	$'$	$\subset$	$'$	$\supset$	$\uparrow$	1

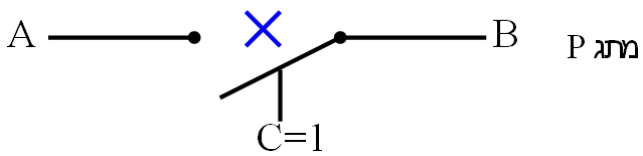
תיאור	שם	אופרטור	פונקציה
אפסיות	Null	0	$F_0 = 0$
גימום: x וגם y	AND	$x \cdot y$	$F_1 = xy$
x ולא y	Inhibition	x/y	$F_2 = xy'$
אין תלות ב-y	Transfer	x	$F_3 = x$
y ולא x	Inhibition	y/x	$F_4 = x'y$
אין תלות ב-x	Transfer	y	$F_5 = y$
איווי אקסקלוסיבי: x או y , אך לא שניהם ביחד	XOR	$x \oplus y$	$F_6 = xy' + x'y$
איווי: x או y	OR	$x + y$	$F_7 = x + y$
שלילת איווי	NOR	$x \downarrow y$	$F_8 = (x + y)'$
שקילות: x שווה ל-y	Equivalence	$x \odot y$	$F_9 = xy + x'y'$
השלמה: לא y	Complement	y'	$F_{10} = y'$
גרירה: אם x אז y	Implication	$x \subset y$	$F_{11} = x + y'$
השלמה: לא x	Complement	x'	$F_{12} = x'$
גרירה: אם y אז x	Implication	$x \supset y$	$F_{13} = x' + y$
שלילת גימום	NAND	$x \uparrow y$	$F_{14} = (xy)'$
זהות	Identity	1	$F_{15} = 1$

<u>מתגים ומימוש שערים לוגיים</u>

מתג N מנותק כאשר C = 0, ומתג P מנותק כאשר C = 1:

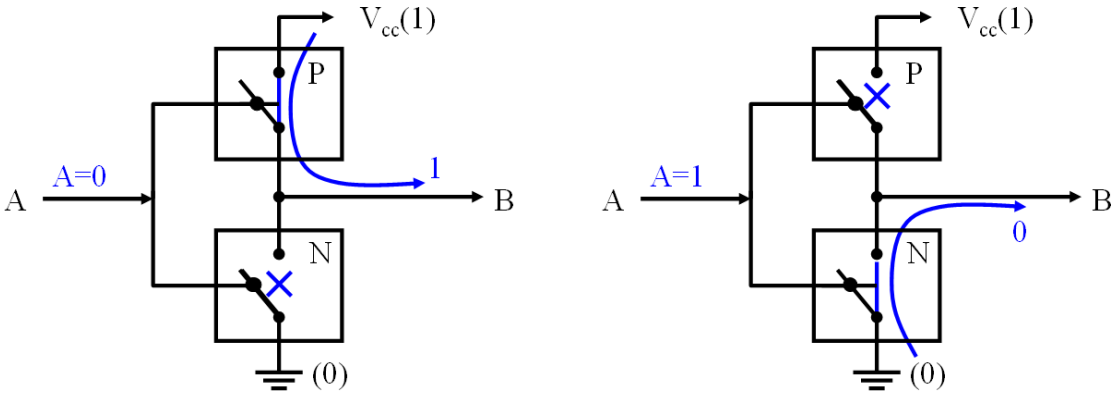


מתג N

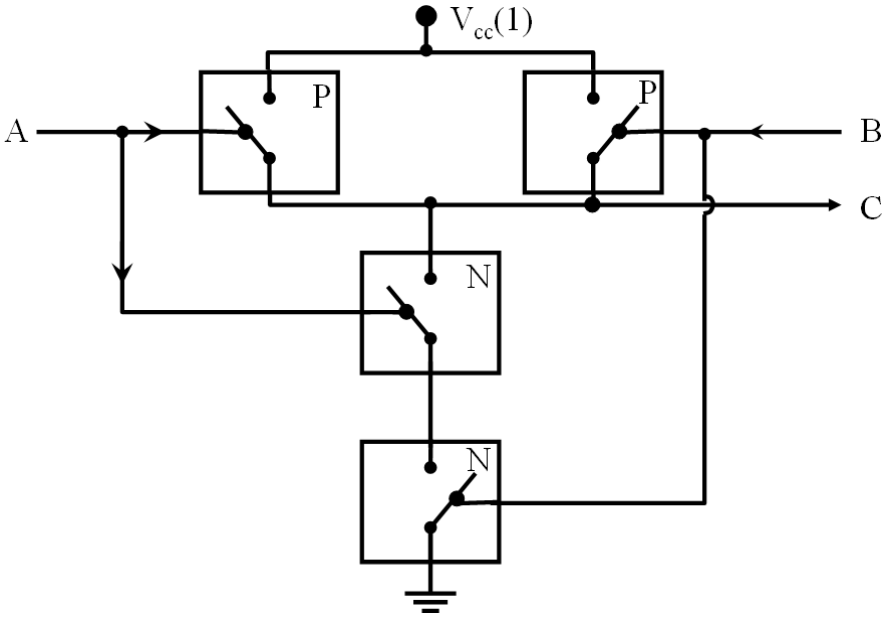


מתג P

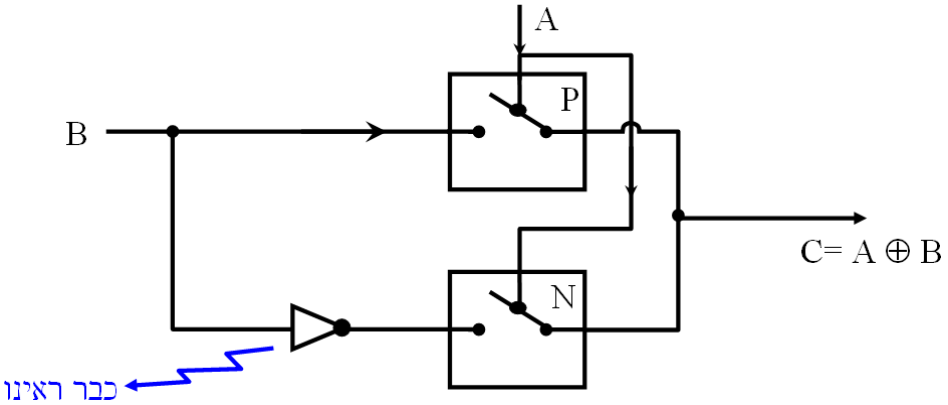
שער NOT ממומש באופן הבא:



שער NAND:



שער XOR:



צורה קנונית וסטנדרטית

סימון לסכום מכפלות: $\sum\left(M_i,\ldots,M_j\right)$

סימון למכפלת סכומים: $\prod\left(M_i,\ldots,M_j\right)$

הכתיב הנ"ל הוא "הצורה הקנונית" עבור פונקציות בוליאניות, וכאשר הוא מצומצם נקרא לו "הצורה הסטנדרטית".

n	*x*	*y*	*z*	minterm	Maxterm
0	0	0	0	$m_0 = x'y'z'$	$M_0 = x + y + z$
1	0	0	1	$m_1 = x'y'z$	$M_1 = x + y + z'$
2	0	1	0	$m_2 = x'yz'$	$M_2 = x + y' + z$
3	0	1	1	$m_3 = x'yz$	$M_3 = x + y' + z'$
4	1	0	0	$m_4 = xy'z'$	$M_4 = x' + y + z$
5	1	0	1	$m_5 = xy'z$	$M_5 = x' + y + z'$
6	1	1	0	$m_6 = xyz'$	$M_6 = x' + y' + z$
7	1	1	1	$m_7 = xyz$	$M_7 = x' + y' + z'$

דוגמא לשימוש בכללי דה-מורגן: $f\left(x,y,z\right) = \sum\left(0,2,3,6,7\right) = \prod\left(1,4,5\right)$

מפות קרנו (Karnaugh)

minterm מבוטאים ע"י 1, maxterm ע"י 0.

ניתן להשתמש ב"צירופים אדישים" (don't-care) ידועים של פונקציה בוליאנית נתונה (סימון: $\varnothing$) כדי לצמצם אף יותר את הצורה הסטנדרטית של הפונקציה

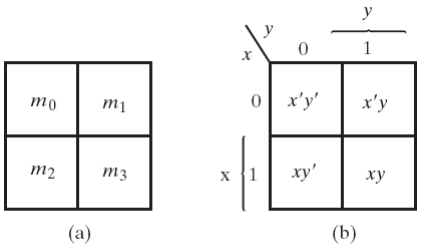


Fig. 3-1 Two-variable Map

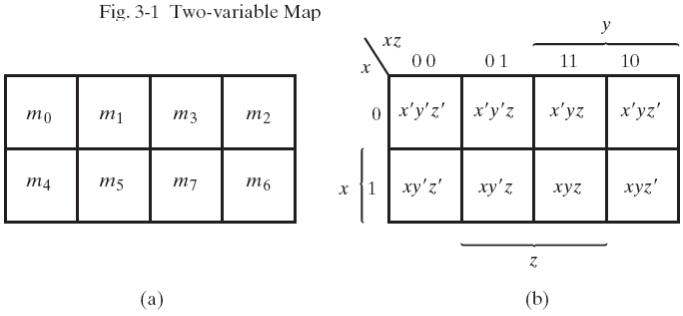


Fig. 3-3 Three-variable Map

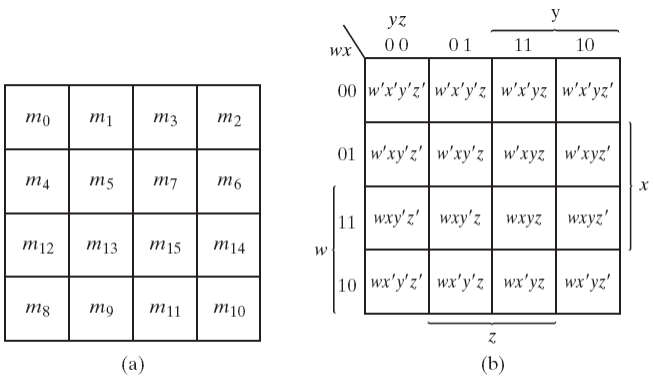


Fig. 3-8 Four-variable Map

סוגי מעגלים

במעגלים צירופיים (combinatorial) הפלט תלוי אך ורק בקלט, ואילו בסדרתיים (sequential) הוא תלוי גם במצב הזכרון שנאגר במעגל מקלטים קודמים.

תזמוני שערים-לוגיים במעגלים צירופיים

rise טווח הזמן שבו הפלט (ערך היציאה) של רכיב מסויים עולה מ-0 (low) ל-1 (high)
fall כנ"ל, אך בכיוון ההפוך – טווח הזמן שבו הפלט של רכיב יורד מ-1 (high) ל-0 (low)

הערה: 2 הערכים הנ"ל נמדדים מהרגע שבו מתח היציאה עבר 10% מן השינוי הכולל, ועד ל-90%.

PLH "זמן עליה ממוצע" - משך הזמן שלוקח ליציאת הפלט להשתנות מ-0 (low) ל-1 (high)
PHL "זמן ירידה ממוצע" - כנ"ל, אך בכיוון ההשתנות ההפוך

הערה: 2 הערכים הנ"ל נמדדים מהרגע שבו מתח הכניסה V_{in} עבר 50% מן השינוי הכולל בקלט, ועד לרגע שבו מתח היציאה V_{out} עבר 50% מן השינוי הכולל בפלט. להלן תרשים להמחשה:

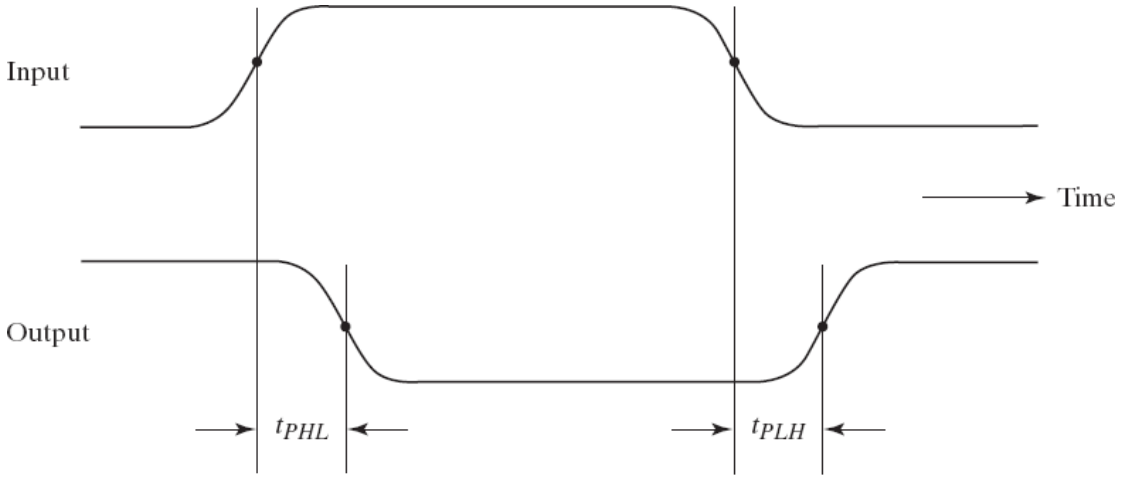
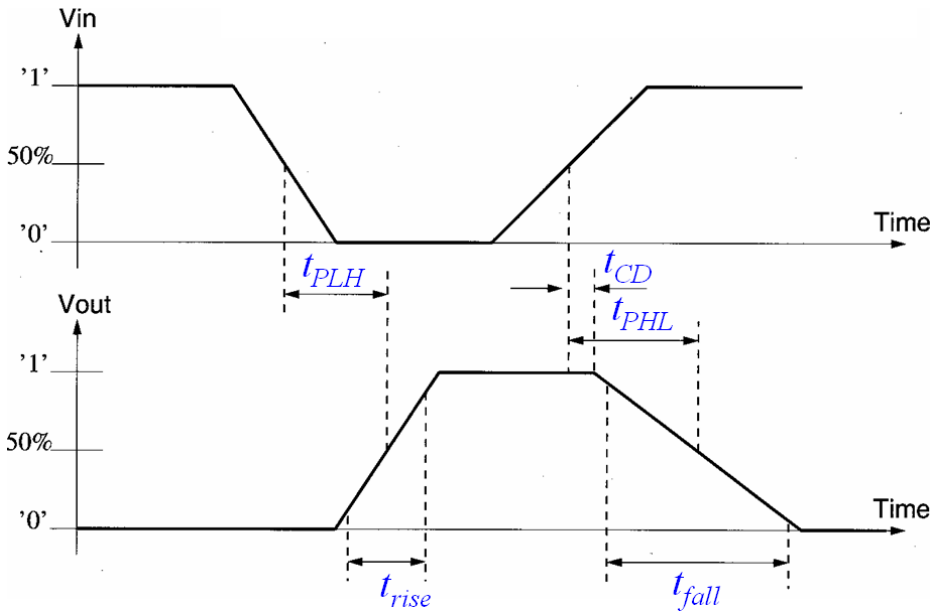


Fig. 10-4 Measurement of Propagation Delay

PD "השהיית הזרימה" (propagation delay) - משך הזמן המקסימלי שלוקח ליציאת הפלט של הרכיב להשתנות מערך אחד למשנהו, כלומר $t_{PD} = \max\{t_{PHL}, t_{PLH}\}$

CD "השהיית הזיהום" (contamination delay) - משך הזמן המינימלי שבו מובטח לנו שערך יציאת הפלט טרם החל להשתנות (קטן מאוד ביחס לשאר הזמנים!)

להלן דוגמא להמחשה - שער NOT:



תזמוני דלגלגים במעגלים סדרתיים

זמן-0 טווח הזמן שבו ערך היציאה של השעון במעגל הינו 0 (low)
זמן-1 כנ"ל, עבור הערך ההפוך 1 (high)

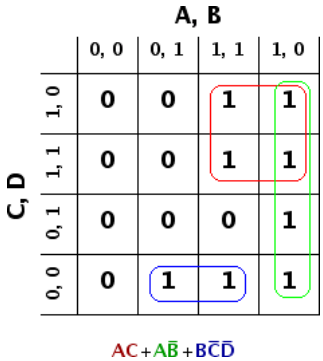
setup משך הזמן שבו קלט הכניסה צריך להיות ערך יציב, לפני תחילת שינוי מצב הדלגלג
hold כנ"ל, אך לאחר תחילת שינוי מצבו של הדלגלג (כלומר אחרי נעילת השעון)

דוגמא: ב-positive-edge flip-flops, t_{setup} נמדד בסוף "זמן 0", לפני שהשעון עובר מ-0 ל-1, ו-t_{hold} נמדד מתחילת "זמן 1", לאחר המעבר במצב השעון. לעומת זאת, ב-negative-edge FF, המצב הוא בדיוק הפוך משום שהם משנים את מצבם בהתאם לקלט רק בירידת השעון מ-1 ל-0.

pC-Q הגירסא הסדרתית ל-*PD* הצירופי (מגבלת המינימום לזמן-1 ב-PEFF / זמן-0 ב-NEFF)
cC-Q הגירסא הסדרתית ל-*CD* הצירופי

הערה: בדר"כ מניחים ש-t_{setup} ו-t_{hold} הם זניחים; גם t_{cC-Q} קטן יחסית, אך הוא חייב לקיים את התנאי $t_{hold} < t_{cC-Q}$ כדי שנוכל לחבר 2 דלגלגים בשרשרת.

הבהובים (Hazards)



ניתן לגלות הבהובים בעזרת מפת קרנו: ההבהובים יתכנו כאשר קיימים מעברים בין מצבים צמודים שאינם מקושרים אחד עם השני, כלומר בין שני minterms בפונקציה הבוליאנית, כאשר לכל אחד מהם משתמש בקלט כלשהו שהשני לא משתמש בו.

בדוגמא משמאל, יתכנו הבהובים ב-2 מקרים: במעבר מקלט $\langle 1,1,0,0 \rangle$ לקלט $\langle 1,0,0,0 \rangle$ או לקלט $\langle 1,1,1,0 \rangle$, ולהיפך (חשוב לזכור שגם הקצוות הנגדיים במפה נחשבים כצמודים).

הפתרון להבהובים סטטיים הוא הוספת minterm משותף בין המצבים.

נוהל ניתוח (אנליזה) למעגל צירופי

1. קביעת משתנים/סמלים ליציאות (בתוך המעגל) התלויות ישירות בקלט
2. מציאת פונקציה בוליאנית לכל אחת מהיציאות הנ"ל
3. חזרה על השלבים הנ"ל "בגלים" עד ליציאות של המעגל השלם
4. ניסוח הפונקציה הבוליאנית הסופית בעזרת הצבה

נוהל ניתוח (אנליזה) למעגל סינכרוני

1. פונקציות בוליאניות לכל כניסות הדלגלגים ולפלט המעגל (כפונקציה של t)
2. רישום טבלאות העירור האופייניות של כל הדלגלגים במעגל (בתור רפרנס)
3. טבלת מצבים: מצב נוכחי, המצב הבא + פלט (כפונקציות של הקלט), ערכים נדרשים לכניסות FF
4. תרשים מצבים (מבוסס על שורות הטבלה)
5. מה המעגל עושה? (בעזרת משוואות מצב, רקורסיה, סדרת בוחן, וכיו"ב)

כללי אצבע לניתוח זמני שעון:

- השעון חייב להיות במצב-1 לזמן של לפחות $T_{pC\text{-}Q}$
- השעון חייב להיות במצב-0 לזמן של לפחות $T_{setup}+T_{PD}$
- זמן הזיהום הקצר ביותר במעגל עם דלגלג חייב להיות קטן מה-T_{hold} שלו

נוהל עיצוב (סינתיזה) למעגל צירופי

1. קביעה וסימון של משתני הכניסה והיציאה
2. טבלת אמת של הפלט כפונקציה של הקלט
3. צמצום הפונקציות הבוליאניות של כל היציאות תחת אילוצי הבעיה
   (הגבלת סוגי רכיבים / מניעת הבהובים / מינימום רכיבים / מינימום זמן)
4. אינטגרציה של המעגל, "קיבוץ" של מכפלות משותפות
5. שרטוט המעגל

נוהל עיצוב (סינתיזה) למעגל סינכרוני

1. בניית תרשים מצבים מצומצם (Mealy/Moore)
2. טבלת מצבים מצומצמת: מצב נוכחי, המצב הבא + פלט (כפונקציה של הקלט)
3. בחירת סוג הדלגלגים המתאימים לפתרון הבעיה (בכמות log של מספר המצבים)
4. שילוב 2 הסעיפים הקודמים עם טבלאות המעברים של הדלגלגים הנבחרים
   (הוספת עמודות של ערכי הכניסות הנדרשים כדי לקיים את הטבלה)
6. חישוב פונקציות בוליאניות לכל כניסות הדלגלגים ולפלט המעגל (כפונקציה של t)
5. צמצום הפונקציות הבוליאניות תחת אילוצי הבעיה (מפות קרנו, וכיו"ב)
6. שרטוט המעגל

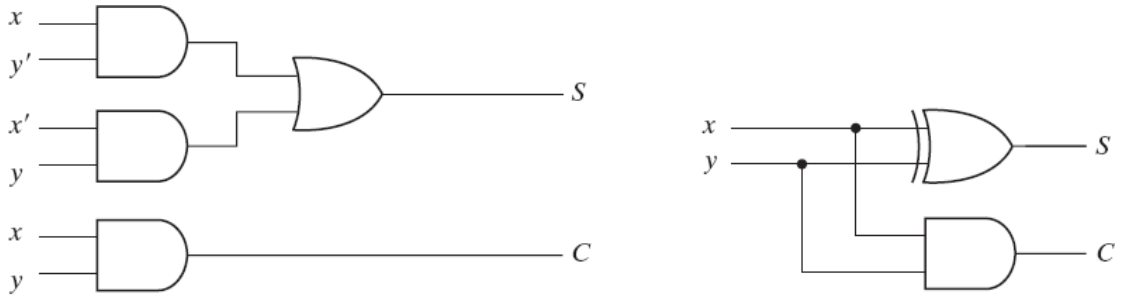
הערה: צמצום מצבים יעשה עפ"י אלגוריתם בשלבים המחלק את המצבים למחלקות שקילות.

תרשימי אוטומטים

מעברי Moore תלויים במצב בלבד, ב-Mealy תלוי במצב ובקלט, פלטי Moore רשומים על המצב, ב-Mealy רשומים על המעבר, Moore הוא צמצום של Mealy (ללא דלגלגים, כלומר ללא זכרון).

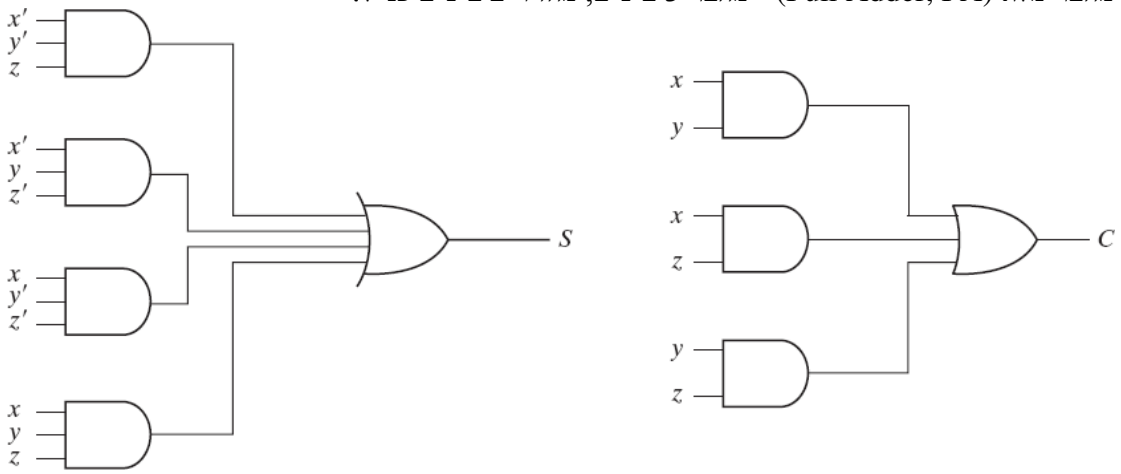
רכיבים סטנדרטיים

חצי מחבר (Half Adder, HA) - מחבר 2 ביטים, מחזיר ביט סכום (sum) וביט נשא (carry):

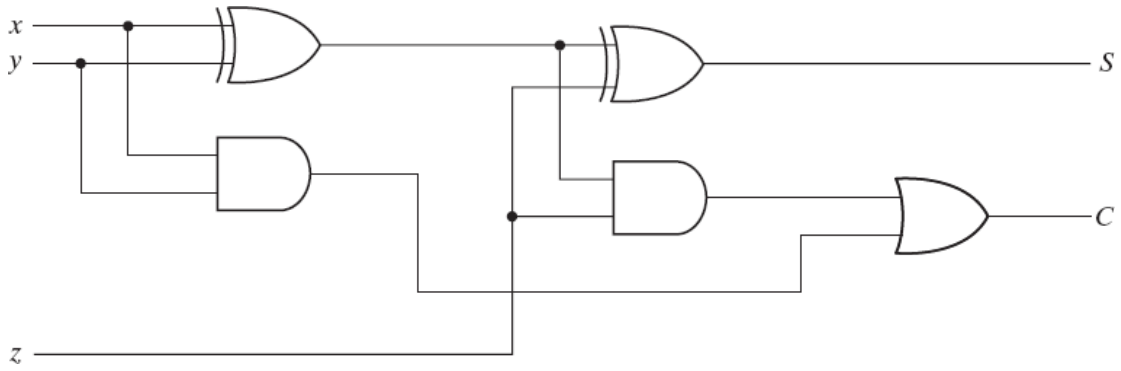


(a) $S = xy' + x'y$, $C = xy$ (b) $S = x \oplus y$, $C = xy$

מחבר מלא (Full Adder, FA) - מחבר 3 ביטים, מחזיר 2 ביטים כנ"ל:



מימוש FA כצירוף של שני HA:



ניתן לשרשר רכיבי FA עבור מחברים רבי-ביטים, ליישם מחסרים (FS) (מוסיפים ביט פעולה שערכו הוא 0 לחיבור ו-1 לחיסור, ב-XOR עם כניסות y במחברים), משווים (comparators), וכו'...

מפענחים (Decoders)

n כניסות ו- 2^n יציאות, כאשר (רק) ביט היציאה i דלוק אם הקלט הכולל הוא המספר i ביצוג בינארי, או בכתיב אחר לגבי כל הביטים של הקלט וכל הביטים של הפלט ביחד, $\text{Decoder}(n) = 2^n$. טבלת אמת:

e	x (in)			d (out)							
	2	**1**	**0**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	0	0	0								1
0	0	0	1							1	
0	0	1	0			0			1		
0	0	1	1					1			
0	1	0	0				1				
0	1	0	1			1					
0	1	1	0		1				0		
0	1	1	1	1							
1	∅	∅	∅	0	0	0	0	0	0	0	0

היציאה i הינה הפלט של המכפלה (minterm) ה-i-ית:

- מימוש: פיצול של כל כניסה ל-2 (הקלט + NOT על הקלט), וכל יציאה היא AND רלוונטי
- שימוש: מימוש של פונקציות בוליאניות כסכום (OR) של מכפלות minterm

הרחבות אפשריות:

- ביט enable (0 = כרגיל, 1 = כל הפלט מאופס) - NOT על קלט ביט זה, וחיבור לכל ה-AND-ים
- שערי NAND במקום AND (היפוך כל הביטים בפלט)

ניתן לממש מפענח $n \to 2^n$ ע"י שני $(n-1) \to 2^{n-1}$, כאשר אחד מביטי הקלט מחובר ל-enable של תתי-המפענחים, עם ובלי NOT. בכתיב בוליאני: $f(x_2, x_1, x_0) = x_2' f(x_1, x_0) + x_2 f(x_1, x_0)$. אך יש לשים לב שבחירות מסוימות יכולות לחסוך את הצורך בתת-מפענח שני משום שהפלט שלו תמיד יהיה תלוי בביט ה-enable.

מקודדים (Encoders)

פונקציה הפוכה לזו של מפענחים. קלט לא חוקי (ביט אחד בלבד דלוק וכל השאר כבויים) יביא לפלט שגוי או לא מוגדר. מימוש: ביט הקלט 0 (lsb) לא מחובר לאף יציאה, והשאר מחוברים לשערי OR.

מקודד עדיפויות (Priority Encoders)

ניתן להוסיף ביט פלט valid לגבי תקינות הקלט (true/false) - מימוש:

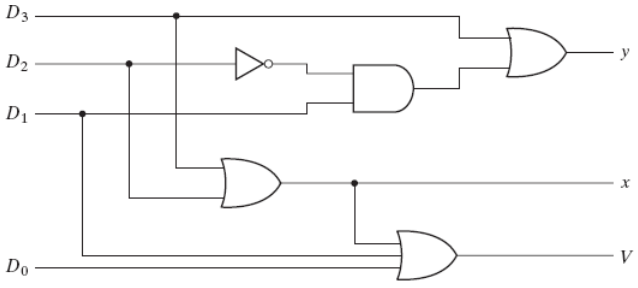
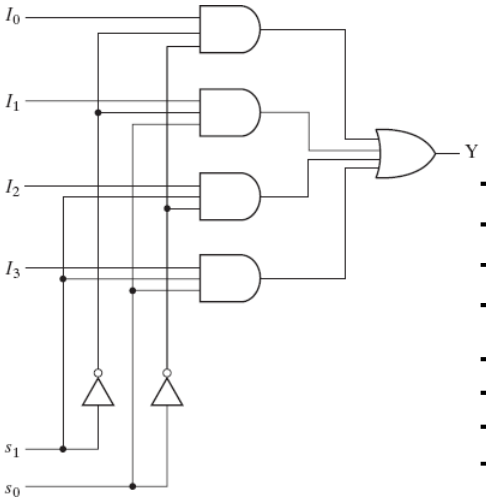


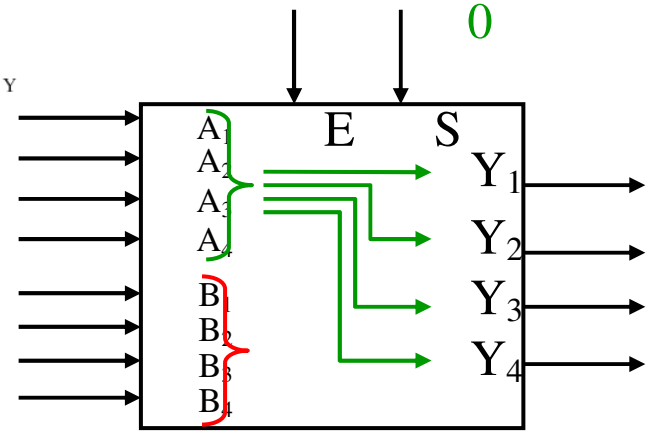
Fig. 4-23 4-Input Priority Encoder

x (in)								e (out)			v
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**	**2**	**1**	**0**	
0	0	0	0	0	0	0	0	0	0	0	0
							1	0	0	0	1
		0				1		0	0	1	1
					1			0	1	0	1
				1				0	1	1	1
			1					1	0	0	1
		1				∅		1	0	1	1
	1							1	1	0	1
1								1	1	1	1

מרכבים (Multiplexers)

קלט: 2^n כניסות + n קוי בחירה (sel), פלט: יציאה אחת, שדרכה מנותב הקלט מן הכניסה הנבחרת.
מימוש: לכל כניסה יש שער AND בצירוף עם פלטי הבחירה שמאפסים כל AND לקלט שלא נבחר, ולפני היציאה שער OR שמסכם את כולם:



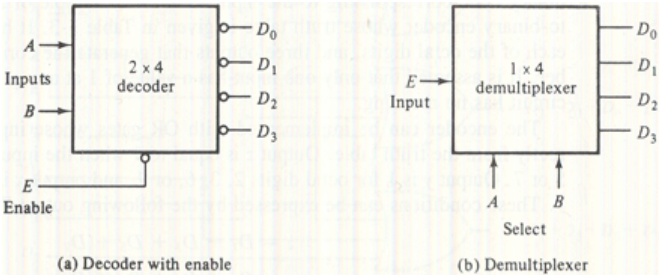


(התרשים מימין מתאר הרחבות אפשרויות: ביט enable ובחירת קבוצות קלטים ולא קו קלט אחד בלבד).

מימוש פונקציות בוליאניות: בדרך הישירה, הקלטים מחוברים לכניסות הבחירה, וה-minterm-ים מזוהים ע"י הדלקת/כיבוי כניסות הקלט. אופציות חסכוניות יותר: משתמשים בטבלת אמת מותנית ומחלקים את הקלטים בין כניסות הבחירה וכניסות הקלט...

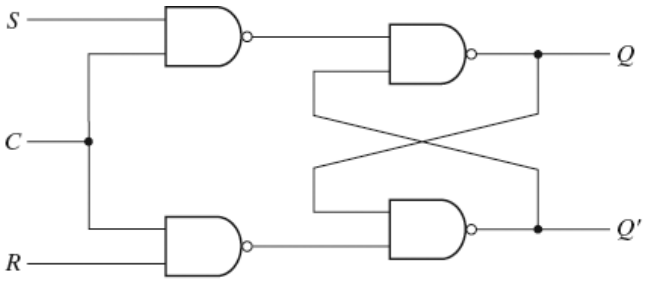
מפלגים (Demultiplexer)

פונקציה הפוכה לזו של מרכבים...
הערה: מפלג (Demux) $1 \leftarrow 2^n$ הוא למעשה מפענח (Decoder) $n \leftarrow 2^n$ (עם ביט איפשור)...



SR-Latch

Set-Reset Latchהבסיסי הוא ללא כניסת שעון (C), אך הוא מאפשר מצב לא חוקי (קלט 1,1).
כדי להתגבר על-כך ניתן להוסיף לו "שכבת הגנה" בצורת שעון:



(a) Logic diagram

C	S	R	Next state of Q
0	X	X	No change
1	0	0	No change
1	0	1	Q = 0; Reset state
1	1	0	Q = 1; set state
1	1	1	Indeterminate

(b) Function table

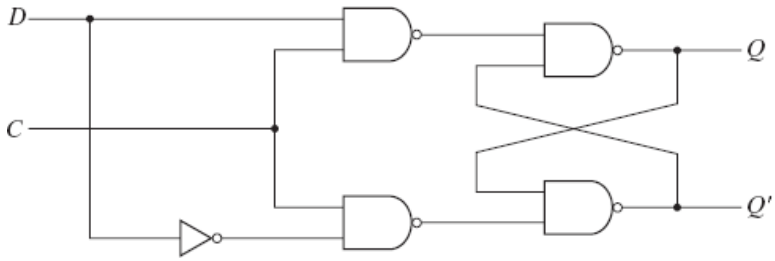
Fig. 5-5 SR Latch with Control Input

טבלת מעברים (עבור הדלגלג למטה, לא המנעול כאן):

Q(t)	Q(t+1)	S	R
0	0	0	∅
0	1	1	0
1	0	0	1
1	1	∅	0

D-Latch

Data Latch הוא יחידת אגירת המידע היסודית שעליה מבוססים אוגרים (Registers).
הוא ממומש על-גבי SR-Latch כך שהקלט הבודד של D-Latch מפוצל באופן הבא:



(a) Logic diagram

C	D	Next state of Q
0	X	No change
1	0	Q = 0; Reset state
1	1	Q = 1; Set state

(b) Function table

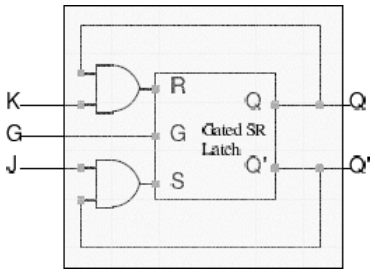
Fig. 5-6 D Latch

טבלת מעברים (עבור הדלגלג למטה, לא המנעול כאן):

Q(t)	Q(t+1)	D
0	0	0
0	1	1
1	0	0
1	1	1

JK-Latch

הרחבה על-גבי SR-Latch:



טבלת עירור אופיינית

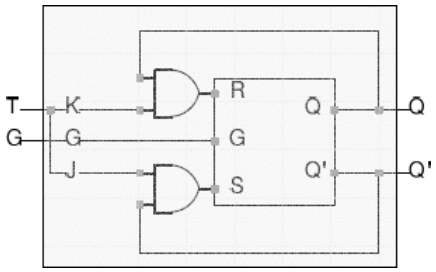
J=S	K=R	Q(t+1)
0	0	Q(t)
0	1	0
1	0	1
1	1	Q'(t)

טבלת מעברים

Q(t)	Q(t+1)	J	K
0	0	0	∅
0	1	1	∅
1	0	∅	1
1	1	∅	0

T-Latch

Trigger Latch הוא JK-Latch בעל כניסה אחת בלבד:



טבלת עירור אופיינית

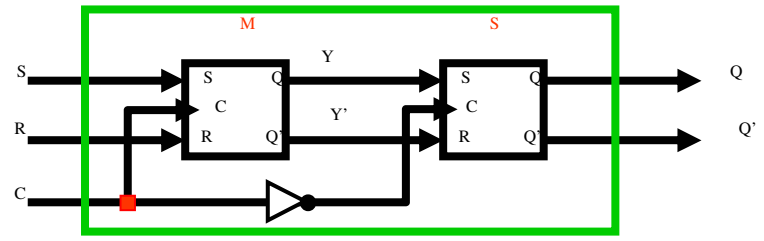
T	Q(t+1)
0	Q(t)
1	Q'(t)

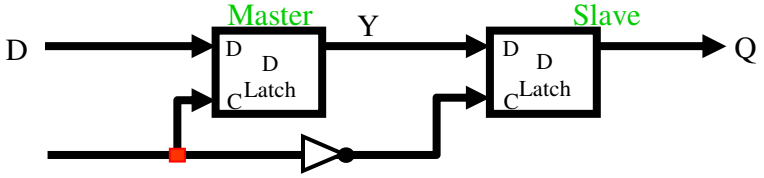
טבלת מעברים

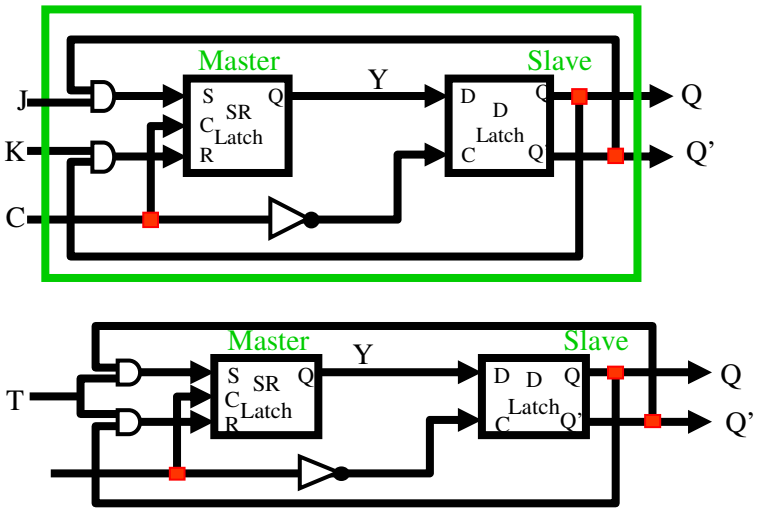
Q(t)	Q(t+1)	T
0	0	0
0	1	1
1	0	1
1	1	0

דלגלגים מתוזמני-שעון (Synchronized Flip-Flops)

יש בעיה עם latches: אם הם מחוברים לשעון, עלינו להבטיח שהוא יפעל זמן קצר מספיק כדי שהם יתכדכו רק פעם אחת בלבד. כדי לפתור בעיה זו משתמשים בדלגלגים הממומשים בשיטת master-slave או "דירבון-שפה" (edge-triggering), כלומר רגישים רק לזמן קצר בעת שינוי הפולס של השעון:







ברכיב positive edge-triggered השינויים בערך-הפנימי וביציאה נקבעים עם עליית השעון,
לעומת-זאת ברכיב negative edge-triggered השינויים נערכים עם ירידת השעון.
איתחול: ע"י preset/clear direct input הקובע ישירות את המצב הפנימי ואת Q(t), עבור Q(t+1).

חישוב הפיך

שער לוגי הוא הפיך (reversible) אם הפונקציה הבוליאנית שהוא מבטא היא חד-חד-ערכית ועל, או במילים אחרות אם לכל פלט y קיים קלט יחודי x כך ש- $L(x)=y$.

הערה: עפ"י "עקרון שובך היונים", מספר הביטים של הקלט חייב להיות שווה למספר הביטים של הפלט.
דוגמאות: שער NOT הוא הפיך, אך שער AND הוא בלתי-הפיך $\left(\mathrm{AND}(0,1)=\mathrm{AND}(1,0)=0\right)$.

כאמור, שער הפיך אינו מאבד אינפורמציה (כלומר אינו מתעל מטעני קלט להארקה, אלא רק מזיז אותם ליציאת הפלט) - ולכן בעקרון שער הפיך אינו מאבד אנרגיה. שער בלתי-הפיך בהכרח מאבד אנרגיה (בצורה של פליטת חום לסביבה) בגלל עקרון האנטרופיה בתרמודינמיקה.

מוטיבציה: בגלל שאין בזבוז אנרגיה, מחשב המתבסס על חישובים הפיכים יכול תיאורטית לשבור את "גבול Landuaer" (שהומצא לראשונה ע"י von-Neuman) - כל פעולה משחררת אנרגיית חום של $k_BT\ln 2$ (k_B קבוע Boltzmann, T טמפרטורת הסביבה).

מעגל בנוי רק משערים הפיכים $\Leftrightarrow$ מאפשר חישוב הפיך $\Leftrightarrow$ ניתן לחישוב במחשב קוונטי.

מבין השערים הלוגיים המוכרים עד-כה, רק NOT ו-XOR הם הפיכים, אך הם אינם מערכת בוליאנית שלמה (אוניברסלית). מערכות שלמות מוכרות: {NOT,OR}, {NOT,AND}, {NOT,AND,OR}, {NOR}, {NAND}.

להלן 2 שערים הפיכים חדשים המהווים מערכות שלמות בפני עצמם:

- שער Toffoli: $n\geq 3$ ביטי קלט, לא משנה את ה-$(n-1)$ הראשונים, וביט הפלט האחרון שווה ל-$\mathrm{XOR}\left(\mathrm{AND}\left(x_1,\ldots,x_{n-1}\right),x_n\right)$.
- שער Fredkin: שלא כמו שער Toffoli, הוא לא מסוגל לשנות את היחס בין כמות ה-0 וה-1. עבור $n\geq 3$ ביטי קלט, ה-$(n-2)$ הראשונים אינם משתנים, אך ה-2 האחרונים משוחלפים אם כל ה-$(n-2)$ הראשונים הם 1.