מדינת ישראל
משרד החינוך

סוג הבחינה: בגרות לבתי־ספר על־יסודיים
מועד הבחינה: קיץ תשע"א, 2011
נספח לשאלון: 815201

אין להעביר את הנוסחאון לנבחן אחר

נוסחאון באלקטרוניקה ומחשבים

(32 עמודים)

1. נוסחאות באלקטרוניקה תקבילית

חישובי הגבר

$$A_V = \frac{V_o}{V_i}$$

A_V – הגבר מתח
V_o [V] – מתח מוצא
V_i [V] – מתח מבוא

$$A_V = 20\log\frac{V_o}{V_i}$$

A_V [dB] – הגבר מתח בדציבלים

$$A_I = \frac{I_o}{I_i}$$

A_I – הגבר זרם
I_o [A] – זרם מוצא
I_i [A] – זרם מבוא

$$A_I = 20\log\frac{I_o}{I_i}$$

A_I [dB] – הגבר זרם בדציבלים

$$A_P = \frac{P_o}{P_i} = A_V \cdot A_I = A_I^2 \cdot \frac{R_L}{R_i} = A_V^2 \cdot \frac{R_i}{R_L}$$

A_P – הגבר הספק
P_o [W] – הספק מוצא
P_i [W] – הספק מבוא
R_L [Ω] – התנגדות נגד העומס
R_i [Ω] – התנגדות מבוא

$$A_P = 10\log\frac{P_o}{P_i}$$

A_P [dB] – הגבר הספק בדציבלים

המשך בעמוד 2

$$A_{VT} = A_{V1} \cdot A_{V2} \cdot A_{V3} \ ..... \ A_{VN}$$

A_{VT} – הגבר כולל של N דרגות המחוברות בשרשרת (קסקדה)

$$A_{VT}\,(\mathrm{dB}) = A_{VI}\,(\mathrm{dB}) + A_{V2}\,(\mathrm{dB}) + A_{V3}\,(\mathrm{dB}) + ... + A_{VN}\,(\mathrm{dB})$$

A_{VT} [dB] – הגבר כולל בדציבלים של N דרגות המחוברות בשרשרת (קסקדה)

<u>מאזן הספקים</u>

$$P_I + P_{CC} = P_L + P_{diss}$$

P_I [W] – הספק מבוא

P_{CC} [W] – הספק נצרך מהספקים

P_L [W] – הספק העומס

P_{diss} [W] – הספק מבוזבז

<u>משוב שלילי</u>

$$A_f = \frac{A}{1+\beta A}$$

A_f [A] – הגבר עם משוב (בחוג סגור)

A – הגבר ללא משוב (הגבר חוג פתוח)

β – מקדם משוב

<u>משוב מתח טורי</u>

$$R_{if} = R_i\,(1+\beta A)$$

R_{if} [Ω] – התנגדות מבוא עם משוב

R_i [Ω] – התנגדות מבוא ללא משוב

$$R_{of} = \frac{R_o}{1+\beta A}$$

R_{of} [Ω] – התנגדות מוצא עם משוב

R_o [Ω] – התנגדות מוצא ללא משוב

המשך בעמוד 3

טרנזיסטור דו־נושאי (בתחום הפעיל)

בהזנחת זרם הזליגה I_{CBO} :

$$I_C = \beta I_B \ , \quad I_E = (\beta + 1)\, I_B \ , \quad I_E = I_C + I_B$$

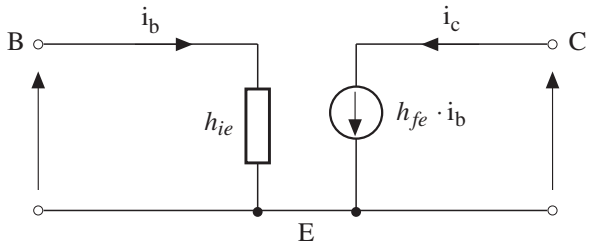
$$\alpha = \frac{I_C}{I_E} = \frac{\beta}{\beta + 1} \ , \quad \beta = \frac{\alpha}{1 - \alpha}$$

I_C [A] – זרם הקולט

I_E [A] – זרם הפולט

I_B [A] – זרם הבסיס

תרשים תמורה מקורב מסוג h של טרנזיסטור דו־נושאי



טרנזיסטור בחיבור פולט (אמיטר) משותף

	ללא נגד R_E	**עם נגד R_E**
A_I	h_{fe}	h_{fe}
R_i	h_{ie}	$h_{ie} + (1 + h_{fe})R_E$
A_V	$-\frac{h_{fe} \cdot R_L}{h_{ie}}$	$-\frac{h_{fe} \cdot R_L}{R_i}$
R_o	∞	∞

המשך בעמוד 4

מגברי שרת

מגבר מהפך

$$A_V = -\frac{R_f}{R_1}$$

- R_f [Ω] – נגד המשוב
- R_1 [Ω] – הנגד המחובר לכניסה המהפכת

מגבר עוקב

$$A_V = 1 + \frac{R_f}{R_1}$$

- R_f [Ω] – נגד המשוב
- R_1 [Ω] – הנגד היוצא מהכניסה המהפכת לאדמה

2. נוסחאות באלקטרוניקה ספרתית

$$v(t) = V_\infty - \left(V_\infty - V_{0+}\right)e^{-\frac{t}{\tau}}$$

$$t = -\tau \cdot \ell n\left(\frac{V_\infty - v(t)}{V_\infty - V_{0+}}\right)$$

- $v(t)$ [V] – הערך הרגעי של המתח
- V_∞ [V] – הערך שאליו המתח שואף להגיע כאשר $t \to \infty$
- V_{0+} [V] – הערך ההתחלתי של המתח

$$\tau = RC$$

- τ [sec] – קבוע זמן
- R [Ω] – התנגדות
- C [F] – קיבול

$$f_H = \frac{1}{2\pi\tau}$$

- f_H [Hz] – תדר חצי הספק עליון של רשת מעבירת נמוכים

$$f_L = \frac{1}{2\pi\tau}$$

- f_L [Hz] – תדר חצי הספק תחתון של רשת מעבירת גבוהים

$$t_r = 2.2\tau$$

- t_r [sec] – זמן עלייה של רשת מעבירת נמוכים

המשך בעמוד 5

3. אוסף פקודות למיקרו־מעבדים 8086/88

מקרא לאוגר הדגלים:

X – מושפע מהפעולה (Modified) 0 – מתאפס

U – לא מוגדר אחרי הפעולה (Undefined) 1 – מקבל '1'

R – מוחזר מהמחסנית (Returned)

ADC	ADC destination, source Add with carry			Flags O D I T S Z A P C X X X X X X
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
register, register	3(3)	—	2	ADC AX, SI
register, memory	9(10)+EA	1	2-4	ADC CX, BETA [SI]
memory, register	16(10)+EA	2	2-4	ADC ALPHA [BX] [SI], DI
register, immediate	4(4)	—	3-4	ADC BX, 256
memory, immediate	17(16)+EA	2	3-6	ADC GAMMA, 30H
accumulator, immediate	4(3-4)	—	2-3	ADC AL, 5

ADD	ADD destination, source Addition			Flags O D I T S Z A P C X X X X X X
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
register, register	3(3)	—	2	ADD CX, DX
register, memory	9(10)+EA	1	2-4	ADD DI, [BX].ALPHA
memory, register	16(10)+EA	2	2-4	ADD TEMP, CL
register, immediate	4(4)	—	3-4	ADD CL, 2
memory, immediate	17(16)+EA	2	3-6	ADD ALPHA, 2
accumulator, immediate	4(3-4)	—	2-3	ADD AX, 200

AND	AND destination, source Logical and			Flags O D I T S Z A P C X X X U X X
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
register, register	3(3)	—	2	AND AL, BL
register, memory	9(10)+EA	1	2-4	AND CX, FLAG_WORD
memory, register	16(10)+EA	2	2-4	AND ASCII [DI], AL
register, immediate	4(4)	—	3-4	AND CX, 0F0H
memory, immediate	17(16)+EA	2	3-6	AND BETA, 01H
accumulator, immediate	4(3-4)	—	2-3	AND AX, 01010000B

המשך בעמוד 6

CALL	CALL target Call a procedure			Flags O D I T S Z A P C
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
near-proc	19(14)	1	3	CALL NEAR__PROC
far-proc	28(23)	2	5	CALL FAR__PROC
memptr 16	21(19)+EA	2	2-4	CALL PROC_TABLE [SI]
regptr 16	16(13)	1	2	CALL AX
memptr 32	37(38)+EA	4	2-4	CALL [BX], TASK [SI]

CLC	CLC (no operands) Clear carry flag			Flags O D I T S Z A P C 0
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
(no operands)	2(2)	—	1	CLC

CLI	CLI (no operands) Clear interrupt flag			Flags O D I T S Z A P C 0
Operands	Clocks	Transfers*	Bytes	Coding Example
(no operands)	2(2)	—	1	CLI

CMP	CMP destination, source Compare destination to source			Flags O D I T S Z A P C X X X X X X
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
register, register	3(3)	—	2	CMP BX, CX
register, memory	9(10)+EA	1	2-4	CMP DH, [ALPHA]
memory, register	9(10)+EA	1	2-4	CMP [BP+2], SI
register, immediate	4(3)+EA	—	3-4	CMP BL, 02H
memory, immediate	10(10)+EA	1	3-6	CMP RADAR [BX], [DI],3420H
accumulator, immediate	4(3-4)	—	2-3	CMP AL, 00010000B

המשך בעמוד 7

CMPS	CMPS des–string, source–string Compare string			Flags O D I T S Z A P C X X X X X X
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
dest–string, source–string (repeat)	22(22)	2	1	CMPS BUFF1, BUFF2
dest–string, source–string	9+22/rep (5+22/rep)	2/rep	1	REPE CMPS ID, KEY

DAA	DAA (no operands) Decimal adjust for addition			Flags O D I T S Z A P C U X X X X X
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
(no operands)	4(4)	—	1	DAA

DAS	DAS (no operands) Decimal adjust for subtraction			Flags O D I T S Z A P C U X X X X X
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
(no operands)	4(4)	—	1	DAS

DEC	DEC destination Decrement by 1			Flags O D I T S Z A P C X X X X X
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
reg 16	3(3)	—	1	DEC AX
reg 8	3(3)	—	2	DEC AL
memory	15(15)+EA	2	2-4	DEC ARRAY [SI]

המשך בעמוד 8

DIV	DIV source Division, unsigned			Flags O D I T S Z A P C U U U U U U U
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
reg 8	80-90(29)	—	2	DIV CL
reg 16	144-162(38)	—	2	DIV BX
mem 8	86-96+EA (35)	1	2-4	DIV [ALPHA]
mem 16	150-168+ EA(94)	1	2-4	DIV TABLE [SI] / DIV [TABLE + SI]

IN	IN accumulator, port Input byte or word			Flags O D I T S Z A P C
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
accumulator, immed 8	10(10)	1	2	IN AL, 0FEH / IN AX, 0FEH
accumulator, DX	8(8)	1	1	IN AL, DX / IN AX, DX

INC	INC destination Increment by 1			Flags O D I T S Z A P C X X X X X
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
reg 16	3(3)	—	1	INC CX
reg 8	3(3)	—	2	INC BL
memory 8	15(15)+EA	2	2-4	INC ALPHA [DI] [BX]

INT	INT interrupt-type Interrupt			Flags O D I T S Z A P C X X
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
immed 8 (type=3)	52(45)	5	1	INT 3
immed 8 (type≠3)	52(47)	5	2	INT 67

IRET	IRET (no operands) Interrupt Return			Flags O D I T S Z A P C R R R R R R R R R
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
(no operands)	32(28)	3	1	IRET

המשך בעמוד 9

JC	JC short-label Jump if carry			Flags O D I T S Z A P C
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
short-label	16 or 4 (13 or 4)	—	2	JC CARRY_SET

JE/JZ	JE/JZ short-label Jump if equal/Jump if zero			Flags O D I T S Z A P C
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
short-label	16 or 4 (13 or 4)	—	2	JZ ZERO

JMP	JMP target Jump			Flags O D I T S Z A P C
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
short-label	15(13)	—	2	JMP SHORT
near-label	15(13)	—	3	JMP WITHIN_SEGMENT
far-label	15(13)	—	5	JMP FAR_LABEL
memptr 16	18(17)+EA	1	2-4	JMP [BX] TARGET
regptr 16	11(11)	—	2	JMP CX
memptr 32	24(26)+EA	2	2-4	JMP OTHER_SEG

JNC	JNC short-label Jump if not carry			Flags O D I T S Z A P C
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
short-label	16 or 4 (13 or 4)	—	2	JNC NOT_CARRY

JNE/JNZ	JNE/JNZ short-label Jump if not equal/Jump if not zero			Flags O D I T S Z A P C
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
short-label	16 or 4 (13 or 4)	—	2	JNE NOT_EQUAL

המשך בעמוד 10

LEA	LEA destination, source Load effective address			Flags O D I T S Z A P C
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
reg 16, mem 16	2(6)+EA	—	2-4	LEA BX, [BP] [DI]

LOOP	LOOP short – label Decrement cx and jump if not zero			Flags O D I T S Z A P C
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
short label	17/5(15/5)	—	2	LOOP AGAIN

MOV	MOV destination, source Move			Flags O D I T S Z A P C
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
memory, accumulator	10(9)	1	3	MOV ARRAY [SI], AL
accumulator, memory	10(8)	1	3	MOV AX, TEMP_RESULT
register, register	2(2)	—	2	MOV AX, CX
register, memory	8(12)+EA	1	2-4	MOV BP, STACK_TOP
memory, register	9(9)+EA	1	2-4	MOV COUNT [DI], CX
register, immediate	4(3-4)	—	2-3	MOV CL, 2
memory, immediate	10(12-13) +EA	1	3-6	MOV MASK [BX] [SI], 2CH
seg-reg, reg 16	2(2)	—	2	MOV ES, CX
seg-reg, mem 16	8(9)+EA	1	2-4	MOV DS, SEGMENT_BASE
reg 16, seg-reg	2(2)	—	2	MOV BP, SS
memory, seg-reg	9(11)+EA	1	2-4	MOV [BX] SEG_SAVE, CS

MOVSB/MOVSW	MOVSB/MOVSW (no operands) Move string (byte/word)			Flags O D I T S Z A P C
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
(no operands)	18(9)	2	1	MOVSB
(repeat) (no operands)	9+17/rep (8+8/rep)	2/rep	1	REP MOVSW

המשך בעמוד 11

MUL	MUL source Multiplication, unsigned			Flags O D I T S Z A P C X U U U U X
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
reg 8	70-77 (26-28)	—	2	MUL BL
reg 16	118-133 (35-37)	—	2	MUL CX
mem 8	76-83+ EA(32-34)	1	2-4	MUL MONTH [SI]
mem 16	124-139+ EA(41-43)	1	2-4	MUL [BAUD_RATE]

NOP	NOP (no operands) No Operation			Flags O D I T S Z A P C
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
(no operands)	3(3)	—	1	NOP

NOT	NOT destination Logical not			Flags O D I T S Z A P C
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
register	3(3)	—	2	NOT AX
memory	16(3)+EA	2	2-4	NOT [CHARACTER]

OR	OR destination, source Logical inclusive or			Flags O D I T S Z A P C X X X U X X
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
register, register	3(3)	—	2	OR AL, BL
register, memory	9(10)+EA	1	2-4	OR DX, PORT_ID [DI]
memory, register	16(10)+EA	2	2-4	OR FLAG_BYTE, CL
accumulator, immediate	4(3-4)	—	2-3	OR AL, 01101100B
register, immediate	4(4)	—	3-4	OR CX, 01H
memory, immediate	17(16)+EA	2	3-6	OR [BX], CMD_WORD

המשך בעמוד 12

OUT	OUT port, accumulator Output byte or word			Flags O D I T S Z A P C
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
immed 8, accumulator	10(9)	1	2	OUT 44, AX
DX, accumulator	8(7)	1	1	OUT DX, AL

POP	POP destination Pop word off stack			Flags O D I T S Z A P C
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
register	8(10)	1	1	POP DX
seg-reg (CS illegal)	8(8)	1	1	POP DS
memory	17(20)+EA	2	2-4	POP [PARAMETER]

PUSH	PUSH source Push word onto stack			Flags O D I T S Z A P C
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
register	11(10)	1	1	PUSH SI
seg-reg (CS legal)	10(9)	1	1	PUSH ES
memory	16(16)+EA	2	2-4	PUSH RETURN_CODE [SI]

RCL	RCL destination, count Rotate left through carry			Flags O D I T S Z A P C X X
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
register, 1	2(2)	—	2	RCL CX, 1
register, CL	8+4/bit (5+1/bit)	—	2	RCL AL, CL
memory, 1	15(15)+EA	2	2-4	RCL ALPHA, 1
memory, CL	20+4/bit (17+1/bit) +EA	2	2-4	RCL [BP].PARAM, CL

המשך בעמוד 13

RCR	RCR destination, count Rotate right through carry			Flags O D I T S Z A P C X X
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
register, 1	2(2)	—	2	RCR BX, 1
register, CL	8+4/bit (5+1/bit)	—	2	RCR BL, CL
memory, 1	15(15)+EA	2	2-4	RCR [BX].STATUS, 1
memory, CL	20+4/bit (17+1/bit) +EA	2	2-4	RCR ARRAY [DI], CL

REP	REP (no operands) Repeat string operation			Flags O D I T S Z A P C
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
(no operands)	2(2)	—	1	REP MOVS DEST, SRCE

RET	RET optional-pop-value Return from procedure			Flags O D I T S Z A P C
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
(intra-segment, no pop)	16(16)	1	1	RET
(intra-segment, pop)	20(18)	1	3	RET 4
(inter-segment, no pop)	26(22)	2	1	RET
(inter-segment, pop)	25(25)	2	3	RET 2

המשך בעמוד 14

ROL	ROL destination, count Rotate left			Flags O D I T S Z A P C X X
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
register, 1	2(2)	—	2	ROL BX, 1
register, CL	8+4/bit (5+1/bit)	—	2	ROL DI, CL
memory, 1	15(15)+EA	2	2-4	ROL FLAG_BYTE [DI], 1
memory, CL	20+4/bit (17+1/bit) +EA	2	2-4	ROL ALPHA, CL

ROR	ROR destination, count Rotate right			Flags O D I T S Z A P C X X
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
register, 1	2(2)	—	2	ROR BX, 1
register, CL	8+4/bit (5+1/bit)	—	2	ROR BX, CL
memory, 1	15(15)+EA	2	2-4	ROR PORT_STATUS, 1
memory, CL	20+4/bit (17+1/bit) +EA	2	2-4	ROR CMD_WORD, CL

SBB	SBB destination, source Subtract with borrow			Flags O D I T S Z A P C X X X X X X X
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
register, register	3(3)	—	2	SBB BX, CX
register, memory	9(10)+EA	1	2-4	SBB DI, [BX].PAYMENT
memory, register	16(10)+EA	2	2-4	SBB BALANCE, AX
accumulator, immediate	4(3-4)	—	2-3	SBB AX, 2
register, immediate	4(4)	—	3-4	SBB CL, 1
memory, immediate	17(16)+EA	2	3-6	SBB COUNT [SI], 10

המשך בעמוד 15

STC	STC (no operands) Set carry flag			Flags O D I T S Z A P C 1
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
(no operands)	2(2)	—	1	STC

STI	STI (no operands) Set interrupt enable flag			Flags O D I T S Z A P C 1
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
(no operands)	2(2)	—	1	STI

SUB	SUB destination, source Subtraction			Flags O D I T S Z A P C X X X X X X X
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
register, register	3(3)	—	2	SUB CX, BX
register, memory	9(10)+EA	1	2-4	SUB DX, MATH_TOTAL [SI]
memory, register	16(10)+EA	2	2-4	SUB [BP+2], CL
accumulator, immediate	4(3-4)	—	2-3	SUB AL, 10
register, immediate	4(4)	—	3-4	SUB SI, 5280

TEST	TEST destination, source Non-destructive logical and			Flags O D I T S Z A P C X X X U X X
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
register, register	3(3)	—	2	TEST SI, DI
register, memory	9(10)+EA	1	2-4	TEST SI, END_COUNT
accumulator, immediate	4(3-4)	—	2-3	TEST AL, 00100000B
register, immediate	5(4)	—	3-4	TEST BX, 0CC4H
memory, immediate	11(10)+EA	—	3-6	TEST [RETURN_COUNT],01H

המשך בעמוד 16

XCHG	XCHG destination, source Exchange registers			Flags O D I T S Z A P C
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
accumulator, reg 16	3(3)	—	1	XCHG AX, BX
memory, register	17(17)+EA	2	2-4	XCHG SEMAPHORE, AX
register, register	4(4)	—	2	XCHG AL, BL

XLAT	XLAT source-table Translate			Flags O D I T S Z A P C
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
source-table	11(11)	1	1	XLAT ASCII_TAB

XOR	XOR destination, source Logical exclusive or			Flags O D I T S Z A P C 0 X X U X X
Operands	**Clocks**	**Transfers***	**Bytes**	**Coding Example**
register, register	3(3)	—	2	XOR CX, BX
register, memory	9(10)+EA	1	2-4	XOR CL, MASK_BYTE
memory, register	16(10)+EA	2	2-4	XOR ALPHA [SI], DX
accumulator, immediate	4(3-4)	—	2-3	XOR AL, 01000010B
register, immediate	4(4)	—	3-4	XOR SI, 00C2H
memory, immediate	17(16)+EA	2	3-6	XOR RETURN_CODE, 0D2H

המשך בעמוד 17

4. נוסחאון בשפת C

נוסחאון זה מתאים למהדר Microsoft Visual C++ 2010 Express Edition .
חלקים ממנו מתאימים גם למהדרים אחרים.

Data Types (טיפוסי נתונים)

Name	Description	תאור	Size*	Range*
char	Character or small integer	תו בודד	1 byte	-128 to 127
unsigned char	Unsigned small integer	תו בודד ללא סימן	1 byte	0 to 255
short	Short Integer	מספר שלם קטן	2 bytes	-32768 to 32767
unsigned short	Unsigned short integer	מספר שלם קטן ללא סימן	2 bytes	0 to 65535
int	Integer	מספר שלם	4 bytes	-2147483648 to 2147483647
unsigned int	Unsigned integer	מספר שלם ללא סימן	4 bytes	0 to 4294967295
float	Floating point number	מספר ממשי	4 bytes	+/- 3.4e +/- 38 (~7 digits)
double	Double floating point number	מספר ממשי ארוך	8 bytes	+/- 1.7e +/- 308 (~15 digits)

*הערכים של עמודות אלו תלויים במבנה המחשב שבו נעשה הידור התוכנית.

דוגמאות:

```
char a;
float number;
int b, c;
unsigned short NewNumber;
```

המשך בעמוד 18

Preprocessor directives (הנחיות לקדם-מהדר)

Description	Syntax	Example
macro definitions	#define identifier replacement	#define ArrSize 100

identifier – מזהה ; replacement – תחליף

Operators (אופרטורים)

Description	תאור	Operator
Assignment	השמה	=

Initialization of variables (אתחול משתנים)

```
int d = 0;

d=75;              // decimal number

d=0x4b;            // hexadecimal number
```

Arithmetic operators (אופרטורים חשבוניים)

Description	תאור	Operator
Addition	חיבור	+
subtraction	חיסור	-
multiplication	כפל	*
division	חילוק	/
modulo	שארית	%

המשך בעמוד 19

Relational and equality operators (אופרטורים להשוואה ויחסים)

Description	תאור	Operator
Equal to	שווה	==
Not equal to	שונה	!=
Greater than	גדול מ.	>
Less than	קטן מ.	<
Greater than or equal to	גדול שווה מ.	>=
Less than or equal to	קטן שווה מ.	<=

Logical operators (אופרטורים לוגיים בין ביטויים)

Description	תאור	Operator
NOT	היפוך	!
AND	וגם	&&
OR	או	||

Bitwise Operators (אופרטורים על סיביות)

Description	תאור	ASM equivalent	Operator
AND	וגם	AND	&
Inclusive OR	או כולל	OR	|
Exclusive OR	או מוציא	XOR	^
Bit inversion	היפוך	NOT	~
Shift Left	הזזה שמאלה	SHL	<<
Shift Right	הזזה ימינה	SHR	>>

המשך בעמוד 20

Basic Input/Output (קלט/פלט בסיסי)

Description	Syntax	Example
Standard Output	int putchar (int character);	`int a='G';` `putchar(a);`
Standard Input	int getchar (void);	`int c;` `c=getchar();`

Formatted Input/Output (פלט לפי תבנית)

Description	Syntax	Example
Formatted output	printf(format[,arg1,arg2,...]);	`int num=10;` `printf("num=%d\n",num);`
Formatted Input	scanf(format [,arg1,arg2,...]);	`int num;` `scanf("%d",&num);`

Specifier	Operator	פלט	Example
%c	Character	תו בודד	a
%d	Signed decimal integer	עשרוני שלם	133
%e	Scientific notation	עשרוני כולל נקודה וחזקה של 10	3.012e+4
%f	Decimal floating point	עשרוני כולל נקודה עשרונית	123.45
%s	String of characters	מחרוזת תווים	Hello
%x	Unsigned hexadecimal integer	הקסדצימלי ללא סימן	3fe

המשך בעמוד 21

Conditional Structures (מבני בקרה – משפטי תנאי)

Description	Syntax	Example

if	if (condition)
	{
	statements ;
	}
		`if (d == 100)`
		`{`
		`printf("d is 100");`
		`}`

if .. else	scanf (condition)
	statement1;
	else
	statement2 ;
		`if (d == 100)`
		`printf("d is 100");`
		`else`
		`printf("d is not 100");`

if .. else if .. else	if (condition)
	statement1 ;
	else if (condition)
	statement2 ;
	else
	statement3 ;
		`if (d > 0)`
		`printf("d is positive");`
		`else if (d < 0)`
		`printf("d is negative");`
		`else`
		`printf("d is 0");`

condition – תנאי ; statement – הצהרה

המשך בעמוד 22

Iteration Structures (מבני בקרה - לולאות)

Description	Syntax	Example

while loop	while (expression)
	{
	statements ;
	}
	
		while (n>0)
		{
		printf(" %d \n",n);
		n--;
		}

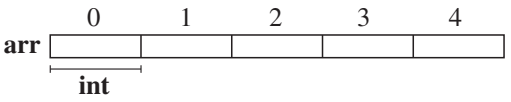
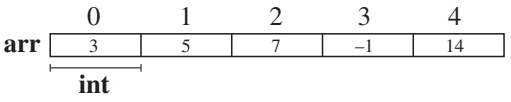
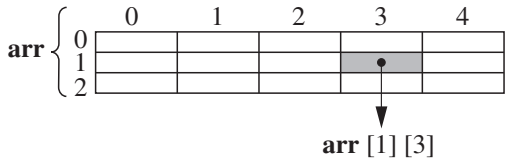
do-while loop	do
	{
	statements ;
	} while (condition);
		do
		{
		printf("Enter 0 to end: ");
		scanf("%d",&n);
		}while (n != 0);

for loop	for (initialization; condition; increase)
	{
	statements ;
	}
		for (i=0; i<10; i++)
		{
		printf(" %d \n",i);
		}

condition – תנאי ; statement – הצהרה

המשך בעמוד 23

Arrays (מערכים)

Description	Syntax	Example
הגדרת מערך חד מימדי 	type name [elements];	`int arr[5];`
אתחול והצבת ערכים במערך 	type name [elements] = {value1,..valueN};	`int arr[5] = {3,5,7,-1,14};`
הגדרת מערך דו מימדי 	type name [elements, elements];	`int arr[3][5];`

elements – פרטים ; value – ערך

Structure of a program (מבנה כללי של תוכנית)

```
#include <stdio.h>

void main(void)

{


}
```

המשך בעמוד 24

Hardware Input/Output (קלט/פלט בסיסי מחומרה)

Description	Syntax	Example
Hardware Output	Out32(hardware address, value);	`Out32(0x378,0xAA);`
Hardware Input	Inp32(hardware address);	`int dataIN;` `dataIN=Inp32(0x379);`

hardware address – כתובת חומרה ; value – ערך

```
#include <stdio.h>
short _stdcall Inp32(short PortAddress);
void _stdcall Out32(short PortAddress, short data);
void main(void)
{
      int dataIN;
      Out32(0x378,0xAA);
      dataIN=Inp32(0x379);
}
```

Sleep Function (פונקציית השהיה)

Description	Syntax	Example
Suspends the execution of the current thread until the time-out interval elapses	void Sleep (dword dwMilliseconds);	Sleep(2000);

*For windows 32-bit registry a DWORD is a 4-bytes unsigned int.

```
#include <windows.h>
void main(void)
{
      Sleep(2000);
}
```

המשך בעמוד 25

5. נוסחאון בשפת VB

נוסחאון זה מתאים למהדר Microsoft Visual Basic 2010 Express Edition .
חלקים ממנו מתאימים גם למהדרים אחרים.

Data Types (טיפוסי נתונים)

Data Type	Size in Bytes	Description	תאור	Type
Byte	1	8-bit unsigned integer	8 ביט ללא סימן	System.Byte
Char	2	16-bit Unicode characters	16 ביט ללא סימן	System.Char
Integer	4	32-bit signed integer	מספר שלם 32 ביט	System.Int32
Double	8	64-bit floating point variable	מספר ממשי 64 ביט	System.Double
Long	8	64-bit signed integer	מספר שלם 64 ביט	System.Int64
Short	2	16-bit signed integer	מספר שלם 16 ביט	System.Int16
Single	4	32-bit floating point variable	מספר ממשי 32 ביט	System.Single
String	Varies	Non-Numeric Type	מחרוזת תווים	System.String
Date	8	Date	תאריך	System.Date
Boolean	2	Non-Numeric Type	ערך בוליאני אמת או שקר	System.Boolean
Decimal	16	128-bit floating point variable	מספר ממשי ארוך 128 ביט	System.Decimal

דוגמאות:

```
Dim a, b As Byte

Dim x As Integer = -20

Dim s As String = "Hello"
```

המשך בעמוד 26

Operators (אופרטורים)

Description	תאור	Operator
Assignment	השמה	=

Initialization of variables (אתחול משתנים)

```
Dim d As Integer

d = 75          ' decimal

d = &H4B        ' hexadecimal
```

Arithmetic operators (אופרטורים חשבוניים)

Description	תאור	Operator
Addition	חיבור	+
Subtraction	חיסור	-
Multiplication	כפל	*
Division	חילוק	/
Modulo	שארית	Mod
Integer Division	חילוק שלמים	\
Exponential	חזקה	^

המשך בעמוד 27

Relational and equality operators (אופרטורים להשוואה ויחסים)

Description	תאור	Operator
Equal to	שווה	=
Not equal to	שונה	<>
Greater than	גדול מ-	>
Less than	קטן מ-	<
Greater than or equal to	גדול שווה מ-	>=
Less than or equal to	קטן שווה מ-	<=

Logical operators (אופרטורים לוגיים בין ביטויים)

Description	תאור	Operator
NOT	היפוך	Not
AND	וגם	And
OR	או	Or

Bitwise Operators (אופרטורים על סיביות)

Description	תאור	ASM equivalent	Operator
AND	וגם	AND	And
Inclusive OR	או כולל	OR	Or
Exclusive OR	או מוציא	XOR	Xor
Bit inversion	היפוך	NOT	Not
Shift Left	הזזה שמאלה	SHL	<<
Shift Right	הזזה ימינה	SHR	>>

המשך בעמוד 28

Basic Input/Output in Console Application (קלט/פלט בסיסי)

Description	Syntax	Example
Standard Output	Console.WriteLine(String)	`Dim value As String = "Hello"` `Console.WriteLine(value)`
Standard Input	String = Console.ReadLine()	`Dim returnValue As String` `returnValue = Console.ReadLine()`

Formatted Output In Console Application (פלט לפי תבנית)

Description	Syntax	Example
Formatted output	Console.WriteLine _ ("{ N [: formatCharacter]}", arg0, ... argN)	`Dim i As Integer = 123456` `Console.WriteLine ("{0:D}", i)`

Format Character	Output	פלט	Example
D or d	Signed decimal integer	עשרוני שלם	`133`
E or e	Scientific notation	עשרוני, כולל נקודה וחזקה של 10	`3.012e+4`
F or f	Decimal floating point	עשרוני, כולל נקודה עשרונית	`123.45`
X or x	Unsigned hexadecimal integer	הקסדצימלי ללא סימן	`3fe`

המשך בעמוד 29

Basic Input/Output in Windows Application (קלט/פלט בסיסי)

Description	Syntax	Example

Standard Output	MsgBox(String)
	or
	MessageBox.Show(String)
		`Dim str As String = "Hello"`
		`MsgBox(str)`
		`MessageBox.Show(str)`

Standard Input	String = InputBox(String)	`Dim s As String`
		`Dim num As Integer`
		`s = InputBox("Enter some text")`
		`num = InputBox("Enter Number") *`

* Automatic casting – המרת טיפוסים אוטומטית

Iteration Structures (מבני בקרה - לולאות)

Description	Syntax	Example

while loop	While expression
	statement
	End While
		`While n > 0`
		`MsgBox(n)`
		`n = n - 1`
		`End While`

do-while loop	Do
	statement
	Loop While condition
		`Do`
		`n = InputBox("Enter 0 to end: ")`
		`Loop While n <> 0`

for loop	for initialization To condition Step increase
	statement
	Next
		`For i = 0 To 6 Step 2`
		`MsgBox("i=" & i)`
		`Next`

condition – תנאי ; statement – הצהרה

המשך בעמוד 30

Conditional Structures (מבני בקרה - משפטי תנאי)

Description	Syntax	Example

if	If condition Then
	[statements]
	End If
		`If d = 100 Then`
		`  MsgBox("d is 100")`
		`End If`

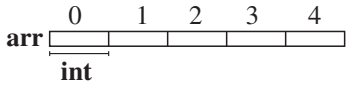
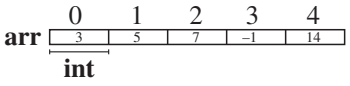
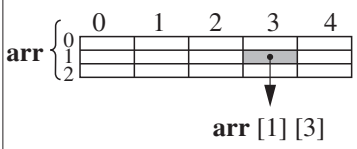
if .. else	If condition Then
	[statements]
	Else
	[statements]
	End If
		`If d = 100 Then`
		`  MsgBox("d is 100")`
		`Else`
		`  MsgBox("d is not 100")`
		`End If`

if .. else if .. else	If condition Then
	[statements]
	Else If condition Then
	[statements]
	…
	Else
	[statements]
	End If
		`If d > 0 Then`
		`  MsgBox("d is positive")`
		`ElseIf d < 0 Then`
		`  MsgBox("d is negative")`
		`Else`
		`  MsgBox("d is 0")`
		`End If`

condition – תנאי ; statement – הצהרה

המשך בעמוד 31

Arrays (מערכים)

Description	Syntax	Example
הגדרת מערך חד מימדי 	Dim name (elements) As type	`Dim arr(5) As Integer`
אתחול והצבת ערכים במערך 	Dim name (elements) As type = {value1,..valueN}	`Dim arr1() As Integer = {3, 5, 7, -1, 14}`
הגדרת מערך דו מימדי 	Dim name (elements, elements) type	`Dim arr2(3, 5) As Integer`

elements – פרטים ; value – ערך

Structure of a program in Console Application (מבנה כללי של תוכנית):

```
Module Module1

    Sub Main()


    End Sub

End Module
```

Structure of a program in Windows Application (מבנה כללי של תוכנית):

```
Public Class Form1

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e
    As System.EventArgs) Handles MyBase.Load

    End Sub

End Class
```

המשך בעמוד 32

Hardware Input/Output (קלט/פלט בסיסי מחומרה)

Description	Syntax	Example
Hardware Output	Out32(hardware address, value);	`Out32(&H378,&HAA);`
Hardware Input	Inp32(hardware address);	`int dataIN;` `dataIN=Inp32(&H379);`

hardware address – כתובת חומרה ; value – ערך

```
Public Class Form1

    Private Declare Function Inp32 Lib "inpout32.dll" Alias
    "Inp32" (ByVal PortAddress As Integer) As Integer

    Private Declare Sub Out32 Lib "inpout32.dll" Alias "Out32"
    (ByVal PortAddress As Integer, ByVal Value As Integer)

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal
    e As System.EventArgs) Handles MyBase.Load

        Dim dataIN As Integer

        Out32(&H378, &HAA)

        dataIN = Inp32(&H379)

    End Sub

End Class
```

Sleep Function (פונקציית השהיה)

Description	Syntax	Example
Suspends the execution of the current thread until the time-out interval elapses.	Public Shared Sub Sleep (_ Milliseconds Timeout As Integer _)	System.Threading.Thread.Sleep(2000)

בהצלחה!