

אלקטרוניקה ומחשבים – מוגבר

הוראות לנבחן

א. משך הבחינה: שעותיים.

ב. מבנה השאלון ומפתח ההערכה: בשאלון זה שבע שאלות בשני פרקים. יש לענות על שלוש שאלות, שאלה אחת לפחות מכל פרק.

לכל שאלה – 33.3 נקודות. סך הכול – 100 נקודות.

ג. חומר עזר מותר לשימוש: מחשבון.

ד. הוראות מיוחדות:

- ענה על מספר השאלות הנדרש בשאלון. המעריך יקרא ויעריך את מספר התשובות הנדרש בלבד, לפי סדר כתיבתן במחברתך, ולא יתייחס לתשובות נוספות.
- התחל כל תשובה לשאלה בעמוד חדש.
- כתוב את כל תשובותיך **אך ורק בעט**.
- הקפד לנסח את תשובותיך כהלכה, ולסרטט את התרשימים בבהירות.
- כתוב את תשובותיך בכתב-יד ברור, כדי לאפשר הערכה נאותה שלהן.
- אם לדעתך חסרים נתונים הדרושים לפתרון שאלה, אתה רשאי להוסיף אותם, אך עליך להסביר מדוע הוספת אותם.
- בכתיבת פתרונות חישוביים, קבלת מִגְרַב הנקודות מותנית בהשלמת כל המהלכים שלהלן, בסדר שהם רשומים בו:
 - * כתיבת הנוסחה המתאימה.
 - * הצבה של כל הערכים ביחידות המתאימות.
 - * חישוב (אפשר באמצעות מחשבון).
 - * כתיבת התוצאה המתקבלת, ולצידה יחידות המידה המתאימות.
 - * ליווי הפתרון החישובי בהסבר קצר.

כתוב במחברת הבחינה בלבד, בעמודים נפרדים, כל מה שברצונך לכתוב כ**טיוטה** (ראשי פרקים, חישובים וכדומה).

כתוב "טיוטה" בראש כל עמוד טיוטה. כיבת טיוטות כלשהן על דפים שמחוץ למחברת הבחינה עלול לגרום לפסילת הבחינה!

בשאלון זה 9 עמודים ו-20 עמודי נספחים.

ההנחיות בשאלון זה מנוסחות בלשון זכר,

אך מכוונות הן לנבחנות והן לנבחנים.

בהצלחה!

המשך מעבר לדף

השאלות

בשאלון זה שבע שאלות בשני פרקים. יש לענות על שלוש שאלות, שאלה אחת לפחות מכל פרק.

פרק ראשון: מערכות ספרתיות

ענה על שאלה אחת לפחות מבין השאלות 1–3 (לכל שאלה – 33.3 נקודות).

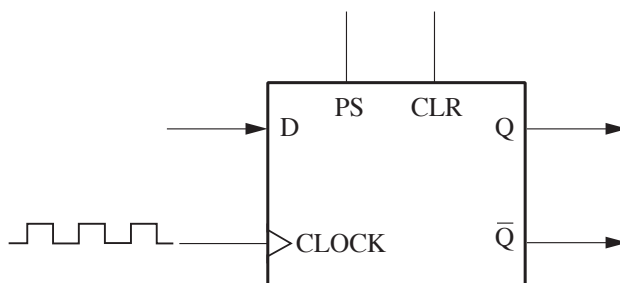
שאלה 1

נתונה הפונקצייה: $F(A, B, C, D) = \sum(1, 2, 5, 8, 10, 13, 15) + \sum_0(0, 6, 7)$

- 13.3 נק' א. הצג את הפונקצייה $F(A, B, C, D)$ באמצעות מפת קרנו.
- 10 נק' ב. בטא את הפונקצייה F כסכום של מכפלות במינימום ליטרלים.
- 10 נק' ג. ממש את הפונקצייה המצומצמת באמצעות שערים לוגיים.

שאלה 2

- 16 נק' א. באיור א' לשאלה 2 מתואר דלגלג מסוג D, המגיב לעליית דופק-השעון ($\uparrow$). לדלגלג מבואות ישירים CLR ו-PS, הפעילים ברמה גבוהה ('1').



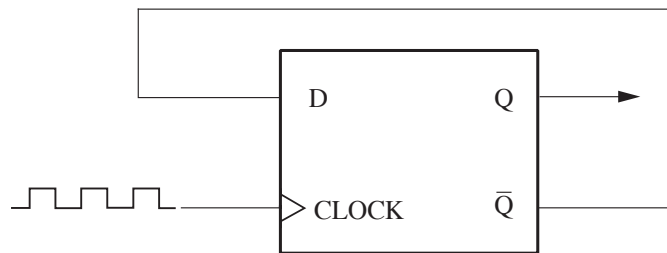
איור א' לשאלה 2

להלן טבלת מצבים לא מלאה של הדלגלג:

CLOCK	D	PS	CLR	Q
'1'	'1'		'0'	'1'
'0'	'1'	'0'		'0'
	'1'	'0'	'0'	
		'0'	'0'	'0'

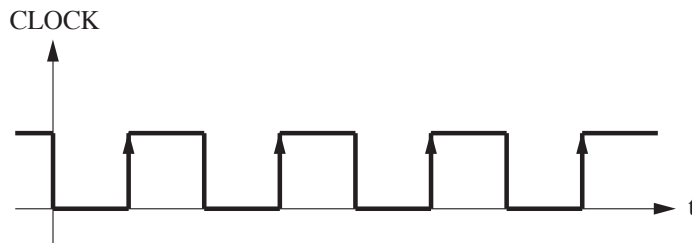
העתק את הטבלה למחברתך והשלם אותה. לווה את תשובותיך בהסבר מתאים.

17.3 נק') ב. מחברים את מוצא הדלגלג $\bar{Q}$ למבוא D, כמתואר באיור ב' לשאלה. נתון כי בזמן $t = 0$ המבואות הישירים CLR ו-PS נמצאים במצב '0', וכן כי $Q = '0'$.



איור ב' לשאלה 2

באיור ג' לשאלה מתואר אות השעון בהדק CLOCK.

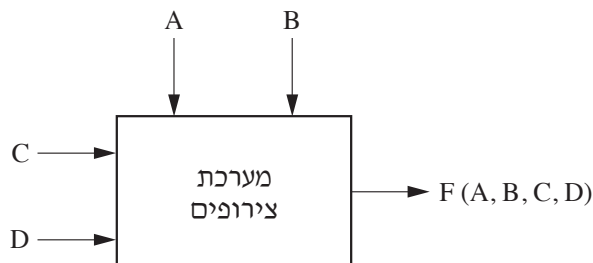


איור ג' לשאלה 2

העתק את איור ג' למחברתך, וסרטט מתחתיו, בהתאמה, ארבעה מחזורים של צורת הגל במוצא Q ושל צורת הגל במוצא $\bar{Q}$ כפונקצייה של הזמן. לווה את סרטוטך בטבלת אמת מתאימה.

שאלה 3

באיור לשאלה 3 נתונה מערכת צירופים בעלת ארבעה מבואות (A, B, C, D) ומוצא F.



איור לשאלה 3

A ו-B הם מבואות בקרה. ערכי המבואות A ו-B קובעים את הפעולה הלוגית שתבצע המערכת בין המבואות C ו-D, כמפורט בטבלה שלהלן:

A	B	F
0	0	$C + D$
0	1	$\overline{C + D}$
1	0	$C \oplus D$
1	1	$\overline{C \oplus D}$

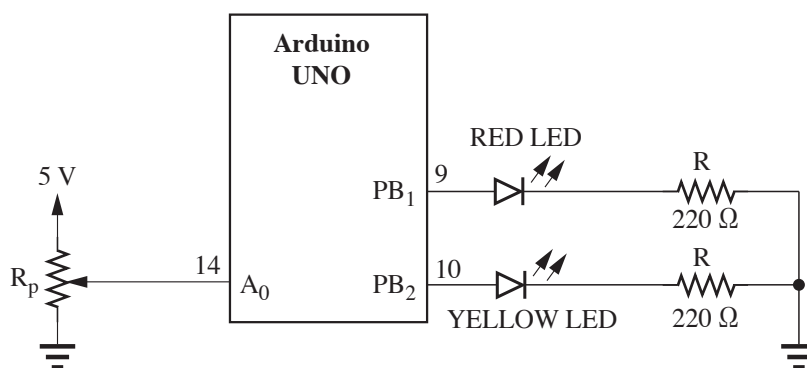
- (10 נק') א. כתוב את טבלת האמת של הפונקצייה $F(A, B, C, D)$.
- (13.3 נק') ב. בטא את הפונקצייה F כסכום של מכפלות במינימום ליטרלים.
- (10 נק') ג. ממש את הפונקצייה F באמצעות שערים לוגיים.

פרק שני: מבוא להנדסת מחשבים

ענה על שאלה אחת לפחות מבין השאלות 4–7 (לכל שאלה – 33.3 נקודות).

שאלה 4

באיור לשאלה 4 נתונה ערכת Arduino UNO. ההדקים PB_1 ו- PB_2 של הערכה מחוברים, בהתאמה, לנורית LED אדומה ולנורית LED צהובה. להדק A_0 של הערכה מחובר פוטנציומטר. לממיר ה-A/D הפנימי במעבד של ה-Arduino יש 10 סיביות.



איור לשאלה 4

להלן תוכנית בשפת C הכתובה לערכת Arduino UNO :

```
1. #define analogPin 0
2. #define LED_R 9
3. #define LED_Y 10
4. int val = 0;
5. void setup()
6. {
7.     pinMode(LED_R, OUTPUT);
8.     pinMode(LED_Y, OUTPUT);
9. }
10. void loop()
11. {
12.     val=analogRead(analogPin);
13.     if (val<512)
14.     {
15.         digitalWrite(LED_R, HIGH);
16.         digitalWrite(LED_Y, LOW);
17.     }
18.     else
19.     {
20.         digitalWrite(LED_R, LOW);
21.         digitalWrite(LED_Y, HIGH);
22.     }
23. }
```

10 נק') א. הסבר את ההוראות שבשורות 1, 4, 12 ו-15.

10 נק') ב.

5 נק') 1. אילו נוריות ידלקו כאשר הזחלן נמצא בקצהו העליון של הפוטנציומטר? נמק את תשובתך.

5 נק') 2. אילו נוריות ידלקו כאשר הזחלן נמצא בקצהו התחתון של הפוטנציומטר? נמק את תשובתך.

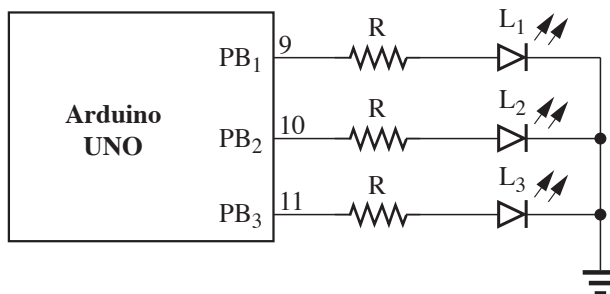
6 נק') ג. הסבר מה מבצע קטע התוכנית שבין השורות 5 ל-9.

7.3 נק') ד. שנה את קוד התוכנית, כך ששתי הנוריות ידלקו כאשר המתח בהדק A_0 יהיה גדול מ-4 V ,

ויהיו כבויות כאשר המתח בהדק A_0 יהיה 4 V או פחות.

שאלה 5

באיור לשאלה 5 נתונות שלוש נוריות LED, המחוברות למפתח B של ערכת Arduino UNO.



איור לשאלה 5

כתוב תוכנית בשפת C לערכת Arduino UNO, שתבצע את הפעולות שלהלן:

1. תקלוט ממקלדת המחשב תו בודד לתוך המשתנה N, תוך שימוש בפקודה Serial.read.
2. אם נקלט התו 'A' – על התוכנית להדליק את הנורית L₁ למשך שנייה אחת.
3. אם נקלט התו 'B' – על התוכנית להדליק את הנורית L₂ למשך שתי שניות.
4. אם נקלט התו 'C' – על התוכנית להדליק את הנורית L₃ למשך ארבע שניות.
5. אם נקלט כל תו אחר – כל הנוריות תהיינה כבויים.

שאלה 6

להלן תת־שגרה הכתובה בשפת־הסף של המיקרו־מעבד 8086/88 :

```

1.  MAIN:      MOV     SI, 200H
2.                MOV     DI, 209H
3.                MOV     BX, 3
4.  NEXT2:     MOV     CX, 3
5.                MOV     AL, 0
6.  NEXT1:     ADD     AL, [SI]
7.                INC     SI
8.                DEC     CX
9.                JNZ     NEXT1
10.             MOV     [DI], AL
11.             INC     DI
12.             DEC     BX
13.             JNZ     NEXT2
14.             RET

```

בטבלה שלהלן נתונים תוכני התאים $200H \div 20BH$ לפני ביצוע התת־שגרה:

כתובת התא	20BH	20AH	209H	208H	207H	206H	205H	204H	203H	202H	201H	200H
תוכן התא	30H	20H	10H	05H	04H	A0H	22H	1AH	18H	07H	04H	03H

(12 נק') א. הסבר את ההוראות 1, 6, 9 ו־10.

(12 נק') ב. כתוב את הערך של כל אחד מן האוגרים AL, CX, BX ו־SI לאחר ביצוע שמונה השורות הראשונות של התת־שגרה.

(9.3 נק') ג. כתוב את תוכני התאים $200H \div 20BH$ לאחר ביצוע התת־שגרה.

שאלה 7

להלן תוכנית בשפת C :

```
1. #include <stdio.h>
2. void main ()
3. {
4.     int arr[12]={3,4,7,1,-1,2,-4,7,4,7,0,-9};
5.     int i,c=0;
6.     printf("arr[12]=");
7.     for (i=0;i<12;i++)
8.     {
9.         printf("%d,",arr[i]);
10.        if(arr[i]>1 && arr[i]<7)
11.            c++;
12.    }
13.    printf("\nc=%d",c);
14. }
```

- 10 נק') א. הסבר את ההוראות שבשורות 4, 6, 9 ו-10 .
- 10 נק') ב. הסבר מה מבצעת התוכנית.
- 6 נק') ג. כתוב את הפלט של התוכנית.
- 7.3 נק') ד. הוסף קטע־קוד בתחילת התוכנית, כך שתוכן המערך arr ייקבע על-ידי נתונים שהמשתמש יקליד תוך כדי הרצת התוכנית.

בהצלחה!

אין להעביר את הנוסחאון
לנבחן אחר

נוסחאון במיקרו-מעבד 8086/88

(12 עמודים)

מקרא לאוגר הדגלים:

0 – מתאפס

X – מושפע מהפעולה (Modified)

1 – מקבל '1'

U – לא מוגדר אחרי הפעולה (Undefined)

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
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
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

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
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

CMPS	CMPS des-string, source-string Compare string			Flags O D I T S Z A P C X X X X X
Operands	Clocks	Transfers*	Bytes	Coding Example
dest-string,	22(22)	2	1	CMPS BUFF1, BUFF2
source-string (repeat)				
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
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
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]

DIV	DIV source Division, unsigned			Flags O D I T S Z A P C 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	1	2-4	DIV [ALPHA]
	(35)			
mem 16	150-168+	1	2-4	DIV TABLE [SI] /
	EA(94)			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

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

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

MOVS/MOVSW	MOVS/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	MOVS
(repeat) (no operands)	9+17/rep (8+8/rep)	2/rep	1	REP MOVSW

MUL	MUL source Multiplication, unsigned			Flags O D I T S Z A P C X 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

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

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 MOVSB 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

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
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

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
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

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

אין להעביר את הנוסחאון
לנבחן אחר

נוסחאון בשפת C לערכת Arduino UNO (8 עמודים)

הנוסחאון מתאים לבקר Arduino UNO. חלקים מנוסחאון זה מתאימים גם לבקרים אחרים ממשפחת Arduino.

Data Types (טיפוסי נתונים)

Name	Description	תאור	Size	Range
Boolean	holds one of two values, true or false	בייט אחד	1 byte	true / false
char	Character or small integer.	תו בודד או בייט אחד	1 byte	-128 to 127
unsigned char	Unsigned small integer.	בייט אחד ללא סימן	1 byte	0 to 255
byte	8-bit unsigned number	בייט אחד ללא סימן	1 byte	0 to 255
int	Integer	מספר שלם	2 bytes	-32768 to 32767
unsigned int	Unsigned integer	מספר שלם ללא סימן	2 bytes	0 to 65535
long	64-bit integer	מספר שלם ארוך	4 bytes	-2147483648 to 2147483647
unsigned long	64-bit unsigned integer	מספר שלם ארוך ללא סימן	4 bytes	0 to 4294967295
Float / double	Floating point number	מספר ממשי	4 bytes	-3.4028235E+38 to 3.4028235E+38
String	String object	מחרוזת	--	--

דוגמאות:

unsigned char num1;	float pi = 3.1416;
int num2,num3;	String NyString = "Hello String";

Initialization of variables (אתחול משתנים):

```
byte num1=75;           // decimal
int num2=0x45f;         // hexadecimal
byte num3=B10010;      // binary
```

Constants (קבועים)

Description	Example
HIGH, 1	digitalWrite (ledPin, HIGH);
LOW, 0	digitalWrite (ledPin, LOW);
INPUT	pinMode (inPin, INPUT);
OUTPUT	pinMode (ledPin, OUTPUT);
INPUT_PULLUP	pinMode (2, INPUT_PULLUP);

Preprocessor directives (הנחיות לקדם-מהדר)

Description	Syntax	Example
macro definitions	#define identifier replacement	#define LED 7

Operators (אופרטורים)

Description	תאור	Operator
Assignment	השמה	=

Arithmetic operators (אופרטורים חשבוניים)

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

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)

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

Bitwise operators (אופרטורים על סיביות)

Description	תאור	Operator
AND	וגם	&
Inclusive OR	או כולל	
Exclusive OR	או מוציא	^
Byte inversion	היפוך בית	~
Shift Left	הזזה שמאלה	<<
Shift Right	הזזה ימינה	>>

Port registers (קלט/פלט)

register	תאור	Example
DDRD	The Port D Data Direction Register – read / write	DDRD = B11111111; //All pins in PORTD are outputs DDRD = B00000000; //All pins in PORTD are inputs
PORTD	The Port D Data Register - read / write	PORTD = B11111111; //All pins in PORTD are high
PIND	The Port D Input Pins Register - read only	char my_var = 0; my_var = PIND; //Read the PORTD

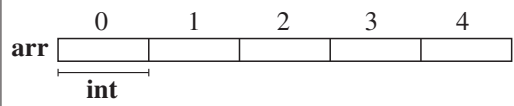
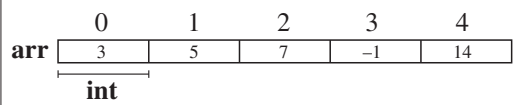
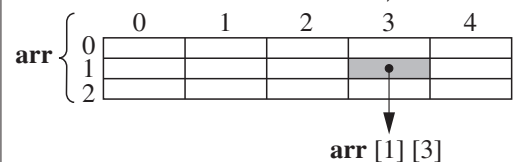
Conditional Structures (מבני בקרה – משפטי תנאי)

Description	Syntax	Example
if	if (condition) statement	if (d == 100) { //..... }
if .. else	if (condition) statement1 else statement2	if (d == 100) //..... else //.....
if .. else if .. else	if (condition) statement1 else if (condition) statement2 else statement3	if (d > 0) //..... else if (d < 0) //..... else //.....

Iteration Structures (מבני בקרה – לולאות)

Description	Syntax	Example
while loop	while (expression) statement;	while (n>0) { n--; }
do-while loop	do statement while (condition);	do { //..... } while (n != 0);
for loop	for (initialization; condition; increase) statement;	for (i=0; i<10; i++) { //..... }

Arrays (מערכים)

Description	Syntax	Example
הגדרת מערך חד מימדי 	type name [elements];	byte arr[5];
אתחול והצבת ערכים במערך 	type name [elements] = {value1,..valueN};	char arr[5] = {3,5,7,-1, 14};
הגדרת מערך דו מימדי 	type name [elements] [elements];	byte arr[3][5];
הגדרת מערך תווים (מחרוזת)	char name [elements] = "string";	char message[6] = "hello";

Structure of a program (מבנה כללי של תכנית)

```
void setup()
{
    // ...
}
void loop()
{
    // ...
}
```

Serial Functions (פונקציות קלט/פלט לתקשורת טורית)

Description	Syntax	Example
Sets the data rate in bits per second (baud) for serial data transmission.	<code>Serial.begin(speed)</code>	<code>Serial.begin(9600);</code>
Get the number of bytes (characters) available for reading from the serial port.	<code>Serial.available()</code>	<code>if (Serial.available()) int inByte = Serial.read();</code>
Reads incoming serial data.	<code>Serial.read()</code>	<code>int inByte = Serial.read();</code>
Prints data to the serial port as human-readable ASCII text.	<code>Serial.print(val)</code> <code>Serial.print(val, format)</code>	<code>int x = 10;</code> <code>// print as an ASCII encoded decimal</code> <code>Serial.print(x);</code> <code>//print as an ASCII encoded hexadecimal</code> <code>Serial.print(x, HEX);</code>
Prints data to the serial port as human-readable ASCII text followed by a carriage return character (ASCII 13, or '\r') and a newline character (ASCII 10, or '\n')	<code>Serial.println(val)</code> <code>Serial.println(val, format)</code>	<code>int x = 10;</code> <code>// print as an ASCII encoded decimal</code> <code>Serial.println(x);</code> <code>//print as an ASCII encoded hexadecimal</code> <code>Serial.println(x, HEX);</code>

Analog And Digital I/O Functions (פונקציות קלט/פלט)

Description	Syntax	Example
Configures the specified pin to behave either as an input or an output.	<code>pinMode (pin, mode)</code>	<code>pinMode (ledPin, OUTPUT);</code>
Write a HIGH or a LOW value to a digital pin.	<code>digitalWrite(pin, value)</code>	<code>digitalWrite(ledPin, HIGH);</code>
Reads the value from a specified digital pin, either HIGH or LOW.	<code>digitalRead(pin)</code>	<code>int val = digitalRead(7);</code>
Configures the reference voltage used for analog input	<code>analogReference(type)</code> type: DEFAULT INTERNAL EXTERNAL	<code>analogReference(INTERNAL);</code> INTERNAL: an built-in reference, equal to 1.1 volts on the ATmega168 or ATmega328 and 2.56 volts on the ATmega8
Reads the value from the specified analog pin(10-bit analog to digital converter)	<code>analogRead(pin)</code>	<code>int val = analogRead(3);</code>
Writes an analog value (PWM wave) to a pin	<code>analogWrite(pin, value)</code>	<code>pinMode(9, OUTPUT);</code> <code>analogWrite(9, 128);</code>
Generates a square wave of the specified frequency (and 50% duty cycle) on a pin	<code>tone(pin, frequency)</code> OR <code>tone(pin, frequency, duration)</code>	<code>tone(12, 261);</code> <code>delay(2000);</code> <code>noTone(12);</code> OR <code>tone(12, 261, 2000);</code>

Servo Functions (פונקציות להפעלת מנוע סרוו)

Description	Syntax	Example
Creates a variable of type Servo	<code>Servo name;</code>	<code>Servo myservo;</code>
Attach the Servo variable to a pin	<code>servo.attach(pin)</code> <code>servo.attach(pin, min, max)</code>	<code>myservo.attach(9);</code>
Read the current angle of the servo	<code>servo.read()</code>	<code>int angle = myservo.read();</code>
Writes a value to the servo	<code>servo.write(angle)</code>	<code>myservo.write(90);</code>

Time Functions (פונקציות שעות)

Description	Syntax	Example
Pauses the program for the amount of time in milliseconds	<code>delay(ms)</code>	<code>delay(1000);</code>
Pauses the program for the amount of time in microseconds	<code>delayMicroseconds(us)</code>	<code>delayMicroseconds(50);</code>
Reads a pulse (either HIGH or LOW) on a pin.	<code>pulseIn(pin, value)</code> <code>pulseIn(pin, value, timeout)</code>	unsigned long duration; <code>pinMode(pin, INPUT);</code> <code>duration = pulseIn(7, HIGH);</code>

Liquid Crystal Functions (פונקציות להפעלת צג מבוסס טקסט)

Description	Syntax	Example
Creates a variable of type LiquidCrystal.	<code>LiquidCrystal(rs, enable, d4, d5, d6, d7)</code> <code>LiquidCrystal(rs, rw, enable, d4, d5, d6, d7)</code>	<code>LiquidCrystal lcd</code> <code>(12, 11, 10, 5, 4, 3, 2);</code>
Initializes the interface to the LCD screen, and specifies the dimensions (width and height) of the display.	<code>lcd.begin(cols, rows)</code>	<code>lcd.begin(16,1);</code>
Prints text to the LCD.	<code>lcd.print(data)</code> <code>lcd.print(data, BASE)</code> base: BIN,OCT,HEX	<code>lcd.print("hello, world!");</code>
Position the LCD cursor	<code>lcd.setCursor(col, row)</code>	<code>lcd.setCursor(0, 1);</code>
Clears the LCD screen and positions the cursor in the upper-left corner.	<code>lcd.clear()</code>	<code>lcd.clear();</code>