|  |  |  |
| --- | --- | --- |
| دستور العمل | مد آدرس دهی اپرند اول(مد آدرس دهی در صورت نداشتن اپرند) | مدرس آدرس دهی **اُپرند** دوم(در صورت وجود) |
| SUB R7, R0 | مستقیم توسط رجیستر | مستقیم توسط رجیستر |
| ANDI R0, 0x40 | مستقیم توسط رجیستر | مستقیم توسط حافظه |
| RJMP 0xFF | آدرس دهی حافظه برنامه نسبی | --- |
| IJMP | غیر مستقیم حافظه | --- |
| RCALL 0x1000 | آدرس دهی حافظه برنامه نسبی | --- |
| JMP 0x1000 | مستقیم داده |  |
| CPC R10, R7 | مستقیم توسط رجیستر | مستقیم توسط رجیستر |
| BRTC 0x400 | آدرس دهی حافظه برنامه نسبی | --- |
| ST –X, R0 | غیر مستقیم با پیش کاهش | مستقیم توسط رجیستر |
| LDI R12, 0x40 | مستقیم توسط رجیستر | مستقیم توسط حافظه |
| STS 0x100, R16 | غیر مستقیم توسط حافظه | مستقیم توسط رجیستر |
| STD X+0x15, R4 | غیر مستقیم توسط حافظه با جا به جایی | مستقیم توسط رجیستر |
| Mov Rd, Rr | مستقیم توسط رجیستر | مستقیم توسط رجیستر |
| ELPM R0, Z | مستقیم توسط رجیستر | آدرس دهی حافظه ی برنامه با آدرس ثابت |
| IN R0, EEDR | مستقیم توسط رجیستر | مستقیم توسط رجیستر |

1.در هر یک از دستورات زیر از چه مدهای آدرس دهی استفاده شده است؟

2.برنامه ای بنویسید که معادل اسکی نام فامیل شما را در حافظه EEPROM میکروکنترلر بنویسید.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Letter** | **ASCII Code** | **Binary** | **Letter** | **ASCII Code** | **Binary** |
| a | 097 | 01100001 | A | 065 | 01000001 |
| b | 098 | 01100010 | B | 066 | 01000010 |
| c | 099 | 01100011 | C | 067 | 01000011 |
| d | 100 | 01100100 | D | 068 | 01000100 |
| e | 101 | 01100101 | E | 069 | 01000101 |
| f | 102 | 01100110 | F | 070 | 01000110 |
| g | 103 | 01100111 | G | 071 | 01000111 |
| h | 104 | 01101000 | H | 072 | 01001000 |
| i | 105 | 01101001 | I | 073 | 01001001 |
| j | 106 | 01101010 | J | 074 | 01001010 |
| k | 107 | 01101011 | K | 075 | 01001011 |
| l | 108 | 01101100 | L | 076 | 01001100 |
| m | 109 | 01101101 | M | 077 | 01001101 |
| n | 110 | 01101110 | N | 078 | 01001110 |
| o | 111 | 01101111 | O | 079 | 01001111 |
| p | 112 | 01110000 | P | 080 | 01010000 |
| q | 113 | 01110001 | Q | 081 | 01010001 |
| r | 114 | 01110010 | R | 082 | 01010010 |
| s | 115 | 01110011 | S | 083 | 01010011 |
| t | 116 | 01110100 | T | 084 | 01010100 |
| u | 117 | 01110101 | U | 085 | 01010101 |
| v | 118 | 01110110 | V | 086 | 01010110 |
| w | 119 | 01110111 | W | 087 | 01010111 |
| x | 120 | 01111000 | X | 088 | 01011000 |
| y | 121 | 01111001 | Y | 089 | 01011001 |
| z | 122 | 01111010 | Z | 090 | 01011010 |

start:

ldi R17,0x00

ldi R18,0x00

ldi R16,0x01

EEPROM\_WRITE:

sbic EECR,EEWE

rjmp EEPROM\_WRITE :

out EEARL,R17

out EEARH,R18

sbi EECR,EEMWE

sbi EECR,EEWE

inc R18

ldi R16, 100 1101 ;M

EEPROM\_write:

sbic EECR,EEWE

rjmp EEPROM\_WRITE

out EEARL,R18

out EEARH,R19

out EEDR,R17

sbi EECR,EEMWE

sbi EECR,EEWE

inc R18

ldi R16,0011 1011 ;I

EEPROM\_write:

sbic EECR,EEWE

rjmp EEPROM\_WRITE

out EEARL,R18

out EEARH,R19

out EEDR,R16

sbi EECR,EEMWE

sbi EECR,EEWE

inc R18

ldi R16, 01010010 ; 82 = r

|  |
| --- |
|  |

EEPROM\_write:

sbic EECR,EEWE

rjmp EEPROM\_WRITE

out EEARL,R18

out EEARH,R19

out EEDR,R16

sbi EECR,EEMWE

sbi EECR,EEWE

inc R18,

ldi R16, 1001101 ;M

EEPROM\_write:

sbic EECR,EEWE

rjmp EEPROM\_WRITE

out EEARL,R18

out EEARH,R19

out EEDR,R16

sbi EECR,EEMWE

sbi EECR,EEWE

inc R18,

ldi R16, 01001111 ;O

|  |
| --- |
|  |

EEPROM\_write:

sbic EECR,EEWE

rjmp EEPROM\_WRITE

out EEARL,R18

out EEARH,R19

out EEDR,R16

sbi EECR,EEMWE

sbi EECR,EEWE

inc R18,

ldi R16,0011 1010 ;H

EEPROM\_write:

sbic EECR,EEWE

rjmp EEPROM\_WRITE

out EEARL,R18

out EEARH,R19

out EEDR,R16

sbi EECR,EEMWE

sbi EECR,EEWE

inc R18,

ldi R16,0100 0100 ;a

EEPROM\_write:

sbic EECR,EEWE

rjmp EEPROM\_WRITE

out EEARL,R18

out EEARH,R19

out EEDR,R16

sbi EECR,EEMWE

sbi EECR,EEWE

inc R18,

ldi R16, 1001101 ;M

EEPROM\_write:

sbic EECR,EEWE

rjmp EEPROM\_WRITE

out EEARL,R18

out EEARH,R19

out EEDR,R16

sbi EECR,EEMWE

sbi EECR,EEWE

inc R18,

ldi R16,1001101;M

EEPROM\_write:

sbic EECR,EEWE

rjmp EEPROM\_WRITE

out EEARL,R18

out EEARH,R19

out EEDR,R16

sbi EECR,EEMWE

sbi EECR,EEWE

inc R18,

ldi R16,0100 0001 ;a

EEPROM\_write:

sbic EECR,EEWE

rjmp EEPROM\_WRITE

out EEARL,R18

out EEARH,R19

out EEDR,R16

sbi EECR,EEMWE

sbi EECR,EEWE

inc R18,

ldi R16,0011 0101 ;D

OUT EEDR, R16

SBI EECR, EEWE

3-وضعیت پرچمها را پس از اجرای هر یک از دستورالعملهای برنامه زیر مشخص نمائید. فرض کنید کلیه پرچمها پس از شروع برنامه 0 هستند.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| C | Z | N | H | S | V | I | T |  |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **LDI R0, 0x3F** |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | **NEG R0** |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0  (0 رو میریزه تو T) | **BST R0, 5** |
| 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | **ADD R0, 0x3F** |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **INC R0** |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | **SEI** |

4-در یک زیر روال، یک بایت داده را از ثبات EEDR از آدرس 0x100 حافظه EEPROM دریافت، آنرا به ثبات R10 منتقل، نیبلهای آنرا جابجا، بیت شماره 4 آنرا 1 و بیت پنجم آن را تست کنید و پیرو آن اقدامات زیر را انجام دهید :

الف- اگر نتیجه تست بیت پنجم 1 بود، مقدار نهایی R0 را در آدرس 0x05 نسبت به مقدار فعلی ثبات Z در حافظه داده ذخیره

نمائید ) Z=0x9A .)

ب- اگر نتیجه تست بیت پنجم 0 بود، محتوای R0 را پس از یک شیفت منطقی به چپ، در عدد 0x2 ضرب و نتیجه را در دو بایت متوالی در پشته ذخیره کنید (SP=0x300)

ج- پس از ذخیره مقدار R0 در پشته، مقدار نهایی SP چقدر است؟

1.ldi R17, 0x00

2.ldi R18, 0x01

3.EEPROM\_read:

4.SBIC EERC, EEWE ; wait

5.RJMP EEPROM\_read

6.OUT EEARH, R18

7.OUT EEARL, R17

8.SBI EECR, EERE

9.IN R10, EEDR ; Read from register

10. SWAP R10

11.SBR R10, 16

12.SBRC R10,4

13.rjump 21

14.rjump 15

15: LSL R0

16: LDI R16, 0x02

17: MULS R0,R16

18: PUSH R0

19: PUSH R1

20: JMP 22

21 : STD Z + 0x05 , R0

22 : FINISH

c)

چون نتیجه در دو بایت متوالی پشت سر هم ذخیره شده است، و دو بایت معادل یک ورد است، از مقدار اشاره گر پشته، یک واحد کم میشود

0x300 -00001 SP final :

5-برنامهای به زبان اسمبلی ATMega16 بنویسید که100 عدد که در آدرس ARRAY در حافظه برنامه قرار گرفته اند را به صورت صعودی مرتب کند )فرض کنید این حافظه از پیش تعریف و مقداردهی شده است(.

به زبان اسمبلی، Bubble sort را پیاده سازی میکنیم:

1. LDI RX , 00000000 #m
2. LDI RZ , 1100100#100 ;outer loop
3. LDI RY , 00000000 #n
4. CALL 9
5. INC RX
6. CPSE RX , RZ
7. JMP 4
8. JMP 23 #end
9. INC RY
10. LDI Z , ARRAY
11. LMP R0 , Z+
12. LMP R1 , Z+
13. CP R0,R1
14. BRLT -5 ; back to the first of the loop
15. MOV R2 , R0 ; swap
16. MOV R0 , R1
17. MOV R1 , R2
18. DEC Z
19. SPM Z+
20. CPST RY, RZ ;compare
21. JMP 9
22. JMP 3