

باسمه تعالی

## تکلیف سری اول ریزپردازنده

سارا برادران (شماره دانشجویی : ۹۶۲۴۱۹۳) – غزاله زمانی (شماره دانشجویی : ۹۷۲۸۰۴۳)

---

### Question 1)

مقادیر R20 و Carry به ترتیب به صورت زیر تغییر می کنند.

	R20	carry
LDI R20, 0x40	0100 0000	-
CLC	0100 0000	0
ROR R20	0010 0000	0
ROR R20	0001 0000	0
ROR R20	0000 1000	0
ROR R20	0000 0100	0
SWAP R20	0100 0000	0

### Question 2)

	R20	Carry
Ldi R20, 0X00	0000 0000	-
Sec	0000 0000	1
Rol R20	0000 0001	0
Cl c	0000 0001	0
Rol R20	0000 0010	0
Sec	0000 0010	1
Rol R20	0000 0101	0
Cl c	0000 0101	0
Rol R20	0000 1010	0
Sec	0000 1010	1
Rol R20	0001 0101	0
Cl c	0001 0101	0
Rol R20	0010 1010	0
Sec	0010 1010	1
Rol R20	0101 0101	0
Cl c	0101 0101	0
Rol R20	1010 1010	0

### Question 3)

```
.DEF INPUT = R20
CBI DDRB, 2
SBI DDRC, 2

START:
IN INPUT, PINB
SBRS INPUT, 2
RJMP IS_OFF
DELAY
DELAY
IN INPUT, PINB
SBRS INPUT, 2
RJMP START
SBI PORTC, 2
RJMP START

.MACRO DELAY
    LDI R30, 120
Delay1:
    LDI R31, 250
Delay2:
    DEC R31
    NOP
    BRNE Delay2
    DEC R30
    BRNE Delay1
.ENDMACRO
```

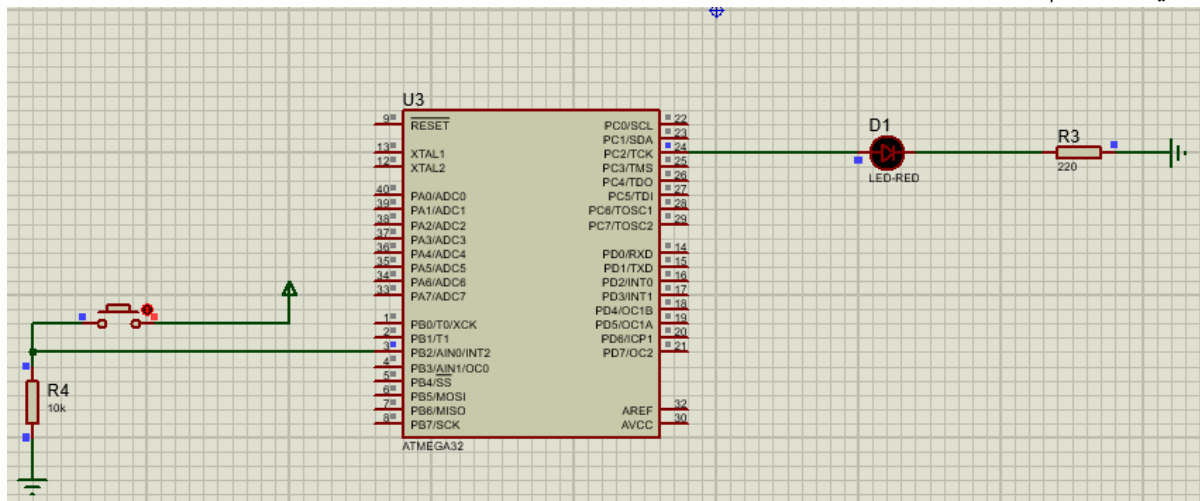
توضیحات :

```
IS_OFF :
DELAY
DELAY
IN INPUT, PINB
SBRC INPUT, 2
RJMP START
CBI PORTC, 2
RJMP START
```

اگر کلید در یک لحظه فشار داده شود و اندکی آن را نگه داریم چراغ روشن می شود در حالی که اگر صرفاً به کلیک سریع بر روی کلید صورت بگیرد مشابه نویز قلمداد شده و چراغ روشن نمی شود.

برای ۱ شدن استیت نیاز است به اندازه زمان  $2 * \text{delay}$  کلید نگه داشته شود.

تصویر مدار رسم شده در پروتئوس



فیلم از کارکرد مدار در پروتئوس

<https://iutbox.iut.ac.ir/index.php/s/LbFnkCd2J4mRnE9>

### Question 4)

The screenshot displays the MASM32 IDE interface. At the top, the 'Registers' window shows a list of registers from R00 to R31. Registers R12, R19, and R20 are highlighted in red. Below this, the 'Disassembly' window is active, showing the assembly code for 'main.asm' within 'AssemblerApplication16'. The code defines INPUT as R20 and CTR as R19. It then loads INPUT with 0X57 and CTR with 0X00. A loop is defined, starting with MOV R21, INPUT, followed by CPI R21, 0X00, BREQ END, SBRC R21, 0, INC CTR, LSR R21, and RJMP LOOP. The 'RJMP LOOP' instruction is highlighted in yellow. The status bar at the bottom indicates the current instruction address is 121.

```
.DEF INPUT = R20
.DEF CTR = R19

LDI INPUT, 0X57

LDI CTR, 0X00
MOV R21, INPUT
LOOP:
CPI R21, 0X00
BREQ END
SBRC R21, 0
INC CTR
LSR R21
RJMP LOOP
END: RJMP END
```

Registers

R00 = 0x00 R01 = 0x00 R02 = 0x00 R03 = 0x00 R04 = 0x00 R05 = 0x00 R06 = 0x00  
R07 = 0x00 R08 = 0x00 R09 = 0x00 R10 = 0x00 R11 = 0x00 R12 = 0x00 R13 = 0x00  
R14 = 0x00 R15 = 0x00 R16 = 0x00 R17 = 0x00 R18 = 0x00 R19 = 0x05 R20 = 0x57  
R21 = 0x00 R22 = 0x00 R23 = 0x00 R24 = 0x00 R25 = 0x00 R26 = 0x00 R27 = 0x00  
R28 = 0x00 R29 = 0x00 R30 = 0x00 R31 = 0x00

Disassembly main.asm AssemblerApplication16

```
SBRC INPUT, 2
INC CTR
SBRC INPUT, 3
INC CTR
SBRC INPUT, 4
INC CTR
SBRC INPUT, 5
INC CTR
SBRC INPUT, 6
INC CTR
SBRC INPUT, 7
INC CTR
END: RJMP END
```

```
.DEF INPUT = R20
.DEF CTR = R19

LDI INPUT, 0X57

LDI CTR, 0X00
SBRC INPUT, 0
INC CTR
SBRC INPUT, 1
INC CTR
SBRC INPUT, 2
INC CTR
SBRC INPUT, 3
INC CTR
SBRC INPUT, 4
INC CTR
SBRC INPUT, 5
INC CTR
SBRC INPUT, 6
INC CTR
SBRC INPUT, 7
INC CTR
END: RJMP END
```

Registers

R00 = 0x00 R01 = 0x00 R02 = 0x00 R03 = 0x00 R04 = 0x00 R05 = 0x00 R06 = 0x00  
R07 = 0x00 R08 = 0x00 R09 = 0x00 R10 = 0x00 R11 = 0x00 R12 = 0x00 R13 = 0x00  
R14 = 0x00 R15 = 0x00 R16 = 0x00 R17 = 0x00 R18 = 0x01 R19 = 0x05 R20 = 0x57  
R21 = 0x01 R22 = 0x00 R23 = 0x00 R24 = 0x00 R25 = 0x00 R26 = 0x00 R27 = 0x00  
R28 = 0x00 R29 = 0x00 R30 = 0x00 R31 = 0x00

Disassembly main.asm AssemblerApplication16

```

LDI R18, 0b10000000
AGAIN:
MOV R21, INPUT
AND R21, R18
BREQ DECREASE
CHECK_END:
CPI R18, 0b00000001
BREQ END
LSR R18
RJMP AGAIN

DECREASE:
DEC CTR
RJMP CHECK_END

END: RJMP END

```

121 %

```

.DEF INPUT = R20
.DEF CTR = R19

LDI CTR, 0x08
LDI INPUT, 0x57
LDI R18, 0b10000000

AGAIN:
MOV R21, INPUT
AND R21, R18
BREQ DECREASE
CHECK_END:
CPI R18, 0b00000001
BREQ END
LSR R18
RJMP AGAIN

DECREASE:
DEC CTR
RJMP CHECK_END

END: RJMP END

```

## Question 5)

Registers

R00 = 0x00 R01 = 0x00 R02 = 0x00 R03 = 0x00 R04 = 0x00 R05 = 0x00 R06 = 0x00  
R07 = 0x00 R08 = 0x00 R09 = 0x00 R10 = 0x00 R11 = 0x00 R12 = 0x00 R13 = 0x00  
R14 = 0x00 R15 = 0x00 R16 = 0x00 R17 = 0x00 R18 = 0x00 R19 = 0x00 R20 = 0x98  
R21 = 0x90 R22 = 0x08 R23 = 0x00 R24 = 0x00 R25 = 0x00 R26 = 0x00 R27 = 0x00  
R28 = 0x00 R29 = 0x00 R30 = 0x00 R31 = 0x00

Disassembly main.asm AssemblerApplication16

```

LDI INPUT, 0xB9 //THE INPUT NUMBER

LDI R21, 0x0F
AND R21, INPUT
LDI R22, 0xF0
AND R22, INPUT
LDI INPUT, 0x00
LSR R22
LSR R22
LSR R22
LSR R22
LSL R21
LSL R21
LSL R21
LSL R21
OR INPUT, R22
OR INPUT, R21

END: RJMP END

```

121 %

```

.DEF INPUT = R20

LDI INPUT, 0xB9 //THE INPUT NUMBER

LDI R21, 0x0F
AND R21, INPUT
LDI R22, 0xF0
AND R22, INPUT
LDI INPUT, 0x00
LSR R22
LSR R22
LSR R22
LSR R22
LSL R21
LSL R21
LSL R21
LSL R21
OR INPUT, R22
OR INPUT, R21

END: RJMP END

```

## Question 6)

فایل حاوی کد با نام `asm.Q6` ضمیمه شده است. تصاویر زیر نتیجه اجرای کد برای ۴ حالت گوناگون تقسیم می باشد.

۱- مقسوم و مقسوم علیه هر دو مثبت

The screenshot shows the AssemblerApplication18 interface. The top window, titled "Registers", displays the initial values of registers R00 through R31. Registers R24 and R25 are highlighted with a red box, showing values 0x01 and 0x03 respectively. The bottom window, titled "Disassembly", shows the assembly code for the program. The code includes instructions to load the low and high bytes of RAMEND into R16, output them to the serial port (SPL and SPH), define the dividend (R16) and divisor (R17), and then load the values 10 and 7 into R24 and R25 respectively.

```
Registers
R00 = 0x00 R01 = 0x00 R02 = 0x00 R03 = 0x00 R04 = 0x00 R05 = 0x00 R06 = 0x00
R07 = 0x00 R08 = 0x00 R09 = 0x00 R10 = 0x00 R11 = 0x00 R12 = 0x00 R13 = 0x00
R14 = 0x00 R15 = 0x00 R16 = 0x03 R17 = 0x07 R18 = 0x00 R19 = 0x00 R20 = 0x00
R21 = 0x00 R22 = 0x00 R23 = 0x00 R24 = 0x01 R25 = 0x03 R26 = 0x00 R27 = 0x00
R28 = 0x00 R29 = 0x00 R30 = 0x00 R31 = 0x00

Disassembly
AssemblerApplication18
main.asm
LDI R16, LOW(RAMEND)
OUT SPL, R16
LDI R16, HIGH(RAMEND)
OUT SPH, R16

.DEF DIVIDEND = R16
.DEF DIVISOR = R17
.DEF QUOTIENT = R24
.DEF REMINDER = R25

LDI DIVIDEND, 10
LDI DIVISOR, 7
```

۲- مقسوم منفی و مقسوم علیه مثبت

The screenshot shows the AssemblerApplication18 interface. The top window, titled "Registers", displays the state of registers R00 through R31. Registers R24 and R25 are highlighted with a red box, showing values 0xFF and 0xFD respectively. The bottom window, titled "Disassembly", shows the assembly code for the program. The code is identical to the first screenshot, but the dividend is loaded as -10 instead of 10.

```
Registers
R00 = 0x00 R01 = 0x00 R02 = 0x00 R03 = 0x00 R04 = 0x00 R05 = 0x00 R06 = 0x00
R07 = 0x00 R08 = 0x00 R09 = 0x00 R10 = 0x00 R11 = 0x00 R12 = 0x00 R13 = 0x00
R14 = 0x00 R15 = 0x00 R16 = 0xFD R17 = 0x07 R18 = 0x00 R19 = 0x00 R20 = 0x00
R21 = 0x00 R22 = 0x00 R23 = 0x00 R24 = 0xFF R25 = 0xFD R26 = 0x00 R27 = 0x00
R28 = 0x00 R29 = 0x00 R30 = 0x00 R31 = 0x00

Disassembly
AssemblerApplication18
main.asm
LDI R16, LOW(RAMEND)
OUT SPL, R16
LDI R16, HIGH(RAMEND)
OUT SPH, R16

.DEF DIVIDEND = R16
.DEF DIVISOR = R17
.DEF QUOTIENT = R24
.DEF REMINDER = R25

LDI DIVIDEND, -10
LDI DIVISOR, 7
```

### ۳- مقسوم مثبت و مقسوم علیه منفی

Registers

R00 = 0x00 R01 = 0x00 R02 = 0x00 R03 = 0x00 R04 = 0x00 R05 = 0x00 R06 = 0x00  
 R07 = 0x00 R08 = 0x00 R09 = 0x00 R10 = 0x00 R11 = 0x00 R12 = 0x00 R13 = 0x00  
 R14 = 0x00 R15 = 0x00 R16 = 0x03 R17 = 0x07 R18 = 0x80 R19 = 0x00 R20 = 0x00  
 R21 = 0x00 R22 = 0x00 R23 = 0x00 R24 = 0xFF R25 = 0x03 R26 = 0x00 R27 = 0x00  
 R28 = 0x00 R29 = 0x00 R30 = 0x00 R31 = 0x00

Disassembly      AssemblerApplication18      main.asm

```
LDI R16, LOW(RAMEND)
OUT SPL, R16
LDI R16, HIGH(RAMEND)
OUT SPH, R16

.DEF DIVIDEND = R16
.DEF DIVISOR = R17
.DEF QUOTIENT = R24
.DEF REMINDER = R25

LDI DIVIDEND, 10
LDI DIVISOR, -7
```

### ۴- مقسوم و مقسوم علیه هر دو منفی

Registers

R00 = 0x00 R01 = 0x00 R02 = 0x00 R03 = 0x00 R04 = 0x00 R05 = 0x00 R06 = 0x00  
 R07 = 0x00 R08 = 0x00 R09 = 0x00 R10 = 0x00 R11 = 0x00 R12 = 0x00 R13 = 0x00  
 R14 = 0x00 R15 = 0x00 R16 = 0xFD R17 = 0x07 R18 = 0x80 R19 = 0x00 R20 = 0x00  
 R21 = 0x00 R22 = 0x00 R23 = 0x00 R24 = 0x01 R25 = 0xFD R26 = 0x00 R27 = 0x00  
 R28 = 0x00 R29 = 0x00 R30 = 0x00 R31 = 0x00

Disassembly      AssemblerApplication18      main.asm

```
LDI R16, LOW(RAMEND)
OUT SPL, R16
LDI R16, HIGH(RAMEND)
OUT SPH, R16

.DEF DIVIDEND = R16
.DEF DIVISOR = R17
.DEF QUOTIENT = R24
.DEF REMINDER = R25

LDI DIVIDEND, -10
LDI DIVISOR, -7
```

109 %

## Question 7)

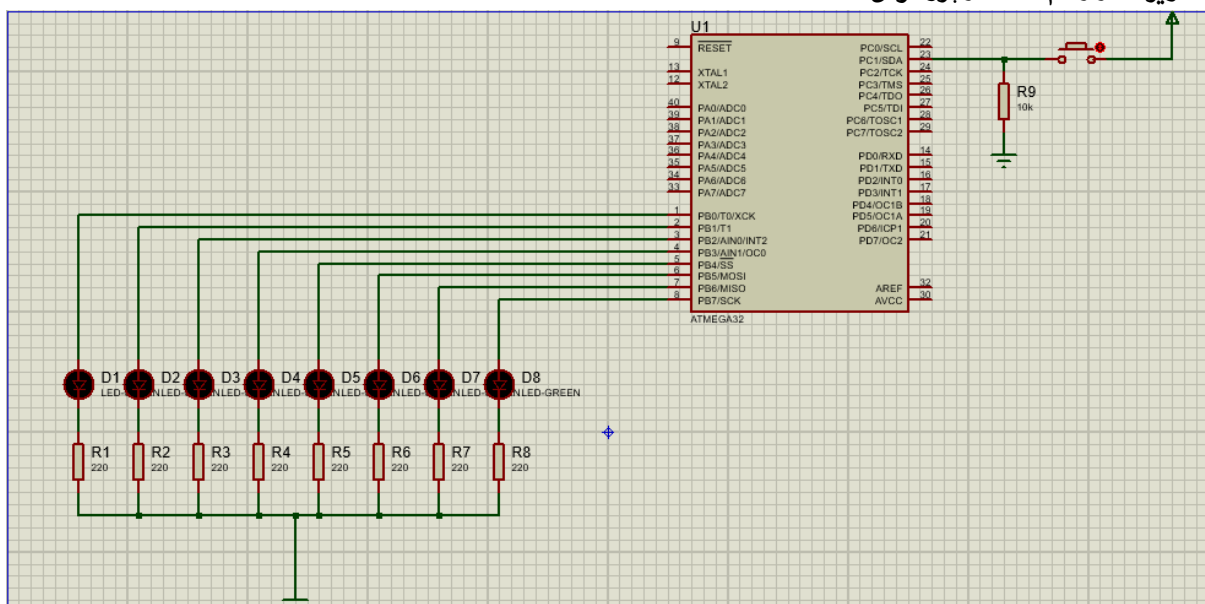
```
LDI R19,0x00
OUT DDRA,R19
IN R16,PINA
LDI R19,0x1B
LDI R21,0x00
LDI R22,0x00
LDI R20,0x00
LOOP: CP R19,R20
BREQ SUBIT
MUL10
ADD R21,R17
ADC R22,R18
DEC R19
JMP LOOP
SUBIT:
CLC
ROL R16
ROL R19
CLC
ROL R16
ROL R19
COM R19
NEG R16
ADC R19,R20
ADD R21,R16
ADC R22,R19
```

چون ملزم به استفاده از دستور MUL10 بودیم، ابتدا حاصل  $PINA * 27 * 10$  را می یابیم سپس حاصل را منهای  $PINA * 4$  می کنیم تا به حقوق سالیانه  $(PINA * 266)$  برسیم که آن را در ثبات های R21 و R22 نمایش می دهیم.

## Question 8)

فایل حاوی کد با نام Q8.c ضمیمه شده است.

تصویر مدار رسم شده در پروتئوس



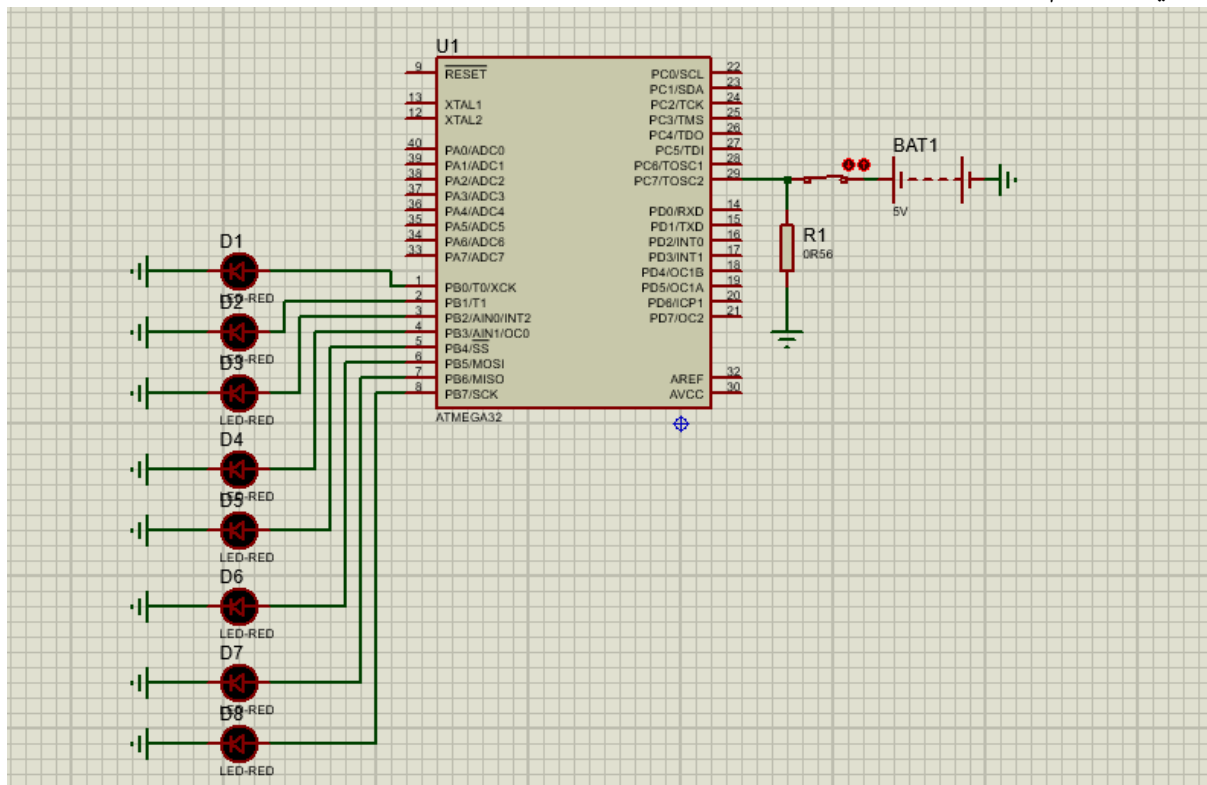
فیلم از کارکرد مدار در پروتئوس

<https://iutbox.iut.ac.ir/index.php/s/FGFpm9YANxSznnz>

## Question 9)

فایل حاوی کد با نام c. Q9 ضمیمه شده است.

تصویر مدار رسم شده در پروتئوس



فیلم از کارکرد مدار در پروتئوس

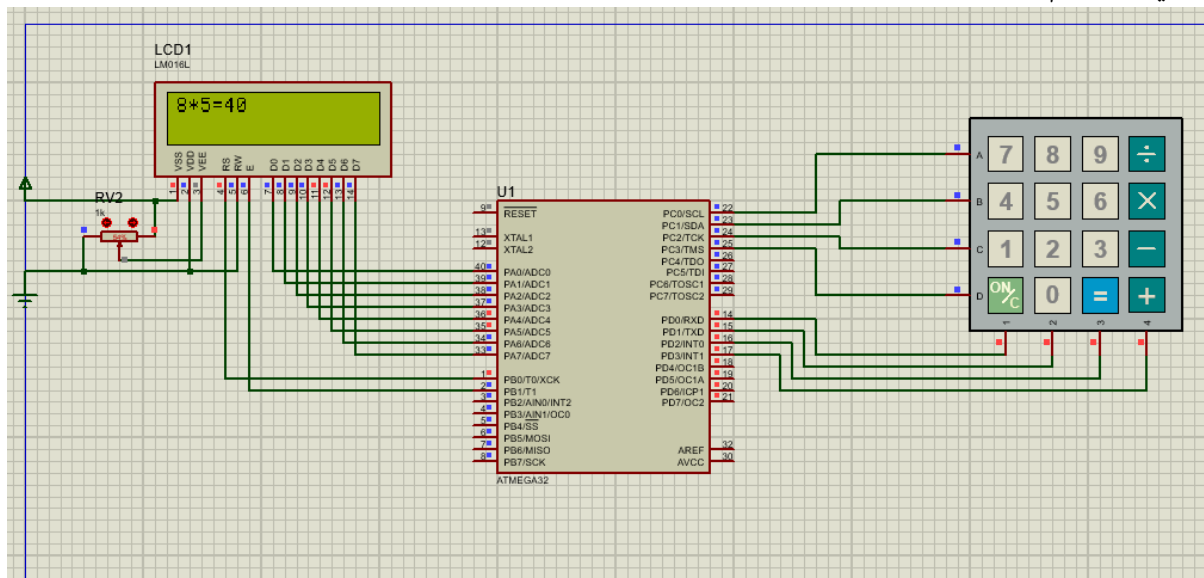
<https://iutbox.iut.ac.ir/index.php/s/emCaRpWnFo9qWNN>



## Question 10)

فایل حاوی کد با نام Q10.c ضمیمه شده است.

تصویر مدار رسم شده در پروتئوس



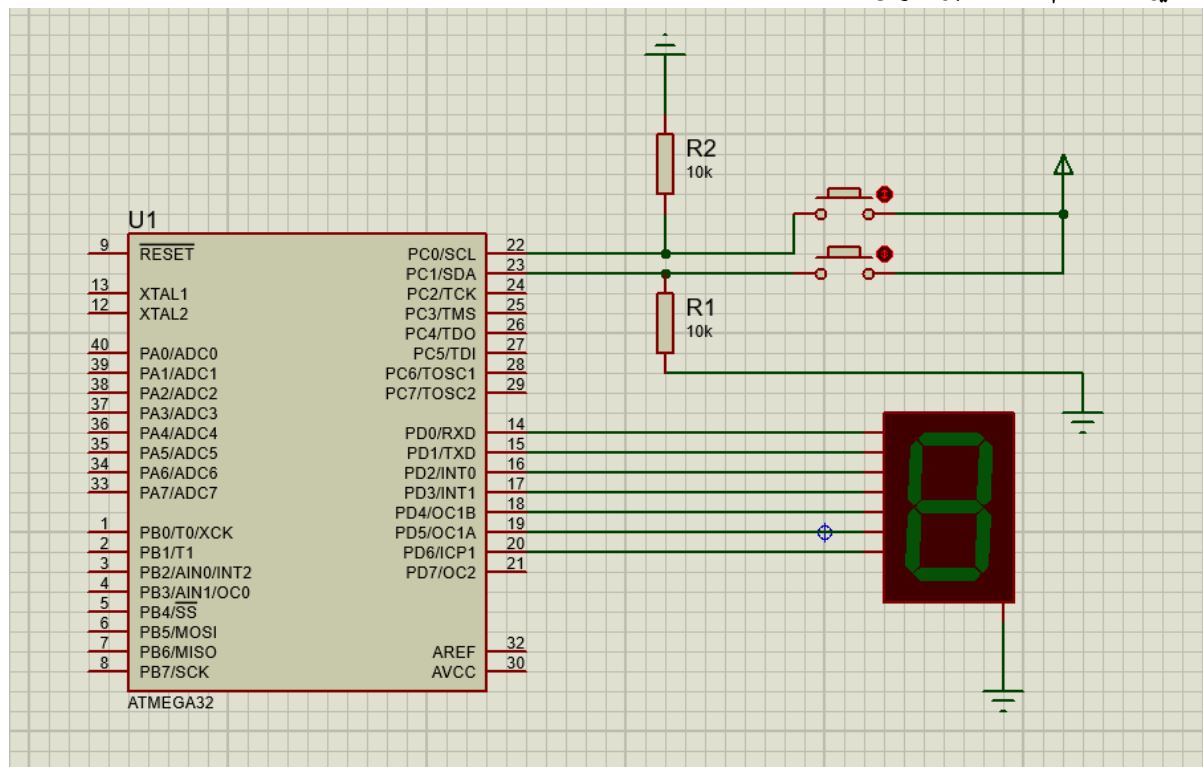
فیلم از کارکرد مدار در پروتئوس

<https://iutbox.iut.ac.ir/index.php/s/koa8gywW5AFA8we>

## Question 11)

فایل حاوی کد با نام Q11.c ضمیمه شده است.

تصویر مدار رسم شده در پروتئوس



فیلم از کارکرد مدار در پروتئوس

<https://iutbox.iut.ac.ir/index.php/s/Nd2qJ6FmEK9TWEE>