



تمرین شماره دو

روش های آدرس دهی

کار با پورت های ورودی و خروجی

درس میکروپروسسور

نویسنده: حمیدرضا ابوئی

شماره دانشجویی: ۹۷۳۳۰۰۲

استاد:

دکتر الماس گنج

تدریس یار:

علی سروشی

زمستان ۱۴۰۰

۱- دو رقم سمت راست شماره دانشجویی خود که در مبنا ۱۰ است را در نظر بگیرید با ۱۰ (۳۹) جمع BCD کنید و بیت های رجیستر وضعیت (status register) را مشخص کنید.
سه رقم آخر شماره دانشجویی: ۰۲

ابتدا اعداد را به فرمت باینری تبدیل می کنیم:

39 => 0011 1001

2 => 0000 0010

0011 1011

اما در حالت BCD در صورتی که رقمی بیشتر از ۹ باشد باید آن را با ۶ جمع کرد:

0011 1011

+0000 0110

0100 0001

4 1

بنابراین از بین تمام flag ها فقط H روشن می شود.

```

;
; AssemblerApplication2.asm
;
; Created: 3/9/2022 9:46:22 PM
; Author : Hamidreza
;

; Replace with your application code
start:

    LDI R16,$39
    LDI R17,$02
    MOV R18,R17
    ADD R18,R16
    LDI R19,6
    ADD R18,R19

    rjmp start

```

Status Register **I T H S V N Z C**

۲- برنامه داخل حلقه چند بار تکرار میشود؟(صحت پاسخ را با ATMEAL STUDIO نشان دهید) .

برای تحلیل این سوال ابتدا روند این برنامه را بررسی می کنیم تا ببینیم برای محاسبه تعداد دفعات باید چه محاسباتی انجام دهیم.

پس از مقداردهی 0x10 و 250 به رجیسترهای R16 و R17، به سراغ قسمت اول می‌رویم. در این قسمت هر بار R17 افزایش می‌یابد تا به عدد ۰ برسد (پرچم Z روشن شود) سپس به قسمت دوم می‌رویم. در این قسمت مقدار R16 کاهش می‌یابد و حلقه به سراغ قسمت قبل می‌رود تا از ابتدا R17 شروع به افزایش کند تا به ۲۵۵ برسد و سپس سرریز کند و Z مجدداً یک شود.

با این روند، تعداد حلقه‌ها را محاسبه می‌کنیم:

قسمت اول حلقه اول: مقدار اولیه $R17 = 250$ مقدار ثانویه $R17 = 0$ تعداد حلقه: 6

قسمت دوم: $R16 = 0x0F$ و R17 مجدداً از ۰ تا ۰ بعدی می‌شمرد: ۲۵۶

قسمت سوم: $R16 = 0x0E$ و...

بنابراین جمع تعداد دفعات برابر است با:

$$6 + 15 \times 256 = 3846$$

برای محاسبه و تست این عدد، برنامه زیر نوشته شد که یک شمارنده ۱۶ بیتی برای این منظور اختصاص داده شده:

```
start:
    LDI R16,0x10
    LDI R17,250
    CLR R18
    CLR R19
    CLR R2
LOOP:
    INC R18
    BRNE ifs
    INC R19
ifs:
    INC R17
    BRNE LOOP
    DEC R16
    BRNE LOOP
    JMP start
```

پس از تست یک دور کامل این برنامه، نتیجه زیر حاصل شد:

R18	0x06
R19	0x0F

که برابر است با عدد $0x0F06 = 3846$

۳ - پرچمهای (Flags) رجیستر وضعیت (Status Register) را پس از اجرای هر خط از برنامه زیر با ذکر علت مشخص کنید (پرچمهای تحت تاثیر هر دستور، در دیتاشیت مشخص شده اند). (صحت جواب خود را با Atmel Studio نشان دهید).

#	ASM	R16	R17	R18	R19	Status Flags
0	initialization	0b11111111	0b11111111	0b00010000	0b01110000	
1	COM R16	0b00000000				Z=1, c=1
2	NEG R16	0b00000000				Z=1, c=0
3	SUB R16,R17	0b00000001				H=1, C=1
4	ADD R16,R17	0b00000000				C=1, H=1, Z=1
5	ADD R19,R18				0b10000000	N=1, V=1

۱ - دستور COM عملیات complement یا مکمل ۱ را اعمال می‌کند. بنابراین تمام بیت‌ها قرینه می‌شوند. به دلیل ۰ شدن

نتیجه نهایی پرچم Z روشن می‌شود همچنین در توضیحات دستورات آمده که این دستور، پرچم C را Set می‌کند

۲ - دستور NEG مکمل دو می‌گیرد و مکمل ۲ عدد صفر است. C فقط در صورتی ۰ می‌شود که محتویات رجیستر ۰ شود یعنی دقیقاً در همین حالت که داریم بررسی می‌کنیم.

۳ - دستور SUB تفریق بدون کرای است R16-R17 را داخل R16 می‌ریزد. در هنگام مکمل ۲ کردن R17، هم کرای و هم Half carry ایجاد می‌شود بنابراین این دو فلگ روشن می‌شوند.

۴ - ۱ با ff جمع می‌شود و هم کرای تولید می‌شود هم Half carry و هم نتیجه نهایی برابر صفر می‌شود.

۵ - با روشن شدن آخرین رقم R19، نشان داده می‌شود که این مقدار این رجیستر منفی است. بنابراین پرچم N روشن می‌شود. همچنین به دلیل Carry این رقم (کرای از رقم قبلی به این رقم) پرچم V روشن می‌شود.

۴- در ATME Studio برنامه‌ای بنویسید که حاصل جمع زیر را در آدرسهای ۱۰۰H به بعد حافظه SRAM ذخیره کند. اعداد را در آدرس 0X60 به بعد حافظه فلش ذخیره کنید و از حافظه فلش بخوانید:

CA30+FADA

start:

```

CLR R5
LDI ZL,0X60
CLR ZH
LPM R0,Z+
LPM R1,Z+
LPM R2,Z+
LPM R3,Z+
ADC R1,R3
ADC R0,R2
ADC R4,R5
LDI ZH,0X01
LDI ZL,0X00
ST Z+,R1
ST Z+,R0
ST Z+,R4
rjmp start

```

```
.ORG 0x0030
.DB $CA,$30
.DB $FA,$DA
```

Memory 4	
Memory: prog FLASH	Address: 0x0060,prog
prog 0x0060	ca 30 fa da ff
prog 0x0079	ff ff

Memory 4	
Memory: prog FLASH	Address: 0x0100,data
data 0x0100	0a c5 01 00
data 0x0119	00 00

۵- برنامه ای بنویسید که ابتدا خانه های H ۲۰۰ تا H ۲۶۴ حافظه SRAM را با اعداد ۱ تا ۱۰۰ پر کند (باحلقه). سپس محتویات این ۱۰۰ سلول را به آدرس H ۴۰۰ به بعد SRAM منتقل کند (باحلقه).

```
start:
    CLR R16
    LDI ZL,0X00
    LDI ZH,0X02
    LDI R17,100
LOOP:
    INC R16
    ST Z+,R16
    CP R16,R17
    BRNE LOOP
    LDI YH,0X02
    CLR YL
    LDI ZH,0X04
    CLR ZL
LOOP2:
    LD R16,Y+
    ST Z+,R16
    CP R16,R17
    BRNE LOOP2
    rjmp start
```

Memory 4			
Memory:	prog FLASH	Address:	0x0200,data
data 0x0200	01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19		
data 0x0219	1a 1b 1c 1d 1e 1f 20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32		
data 0x0232	33 34 35 36 37 38 39 3a 3b 3c 3d 3e 3f 40 41 42 43 44 45 46 47 48 49 4a 4b		
data 0x0248	4c 4d 4e 4f 50 51 52 53 54 55 56 57 58 59 5a 5b 5c 5d 5e 5f 60 61 62 63 64		
data 0x0264	00 00		
Memory 4			
Memory:	prog FLASH	Address:	0x0400,data
data 0x0400	01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19		
data 0x0419	1a 1b 1c 1d 1e 1f 20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32		
data 0x0432	33 34 35 36 37 38 39 3a 3b 3c 3d 3e 3f 40 41 42 43 44 45 46 47 48 49 4a 4b		
data 0x0448	4c 4d 4e 4f 50 51 52 53 54 55 56 57 58 59 5a 5b 5c 5d 5e 5f 60 61 62 63 64		
data 0x0464	00 00		

۶- برنامه ای بنویسید که اعداد ۸ بیتی ۰۲۵۵، ۰۲۰۰، ۱۳۵، ۱۳۴، ۷۲، ۲۰، ۱۵، ۱۳، ۱۱، ۱۰ را در آدرس های H۶۰ الی H69 از حافظه SRAM قرار بدهد. سپس آنهایی که زوج هستند را به فرم مکمل ۲ آنها تبدیل کند و به ترتیب در آدرس H ۱۰۰ به بعد حافظه SRAM قرار دهد و از آنهایی که فرد هستند یک واحد کم کرده و به ترتیب در آدرس H ۱۲۰ به بعد حافظه SRAM قرار دهد.

start:

```
LDI R16,HIGH(RAMEND)
OUT SPH,R16
LDI R16,LOW(RAMEND)
OUT SPL,R16
LDI ZH,0X00
LDI ZL,0X60
LDI R16,10
ST Z+,R16
LDI R16,11
ST Z+,R16
LDI R16,13
ST Z+,R16
LDI R16,15
ST Z+,R16
LDI R16,20
ST Z+,R16
LDI R16,72
ST Z+,R16
LDI R16,134
ST Z+,R16
LDI R16,135
ST Z+,R16
LDI R16,200
ST Z+,R16
LDI R16,255
ST Z+,R16
LDI ZL,0X60
CLR ZH
LDI R17,10
```

LOOP:

ODD:




























[illegible][illegible]

```
LDI R16,0B10111111
OUT DDRA,R16
COM R16
OUT PORTA,R16
LDI R16,0B00001000
start:
    CLR R17
    IN R16,PINA
    BST R16,6
    BLD R17,3
    OUT PORTB,R17
    rjmp start
```




























Name	Address	Value	Bits
IO PINA	0x39	0x40	<input type="checkbox"/> <input checked="" type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
IO DDRA	0x3A	0xBF	<input checked="" type="checkbox"/> <input type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/>
IO PORTA	0x3B	0x40	<input type="checkbox"/> <input checked="" type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>

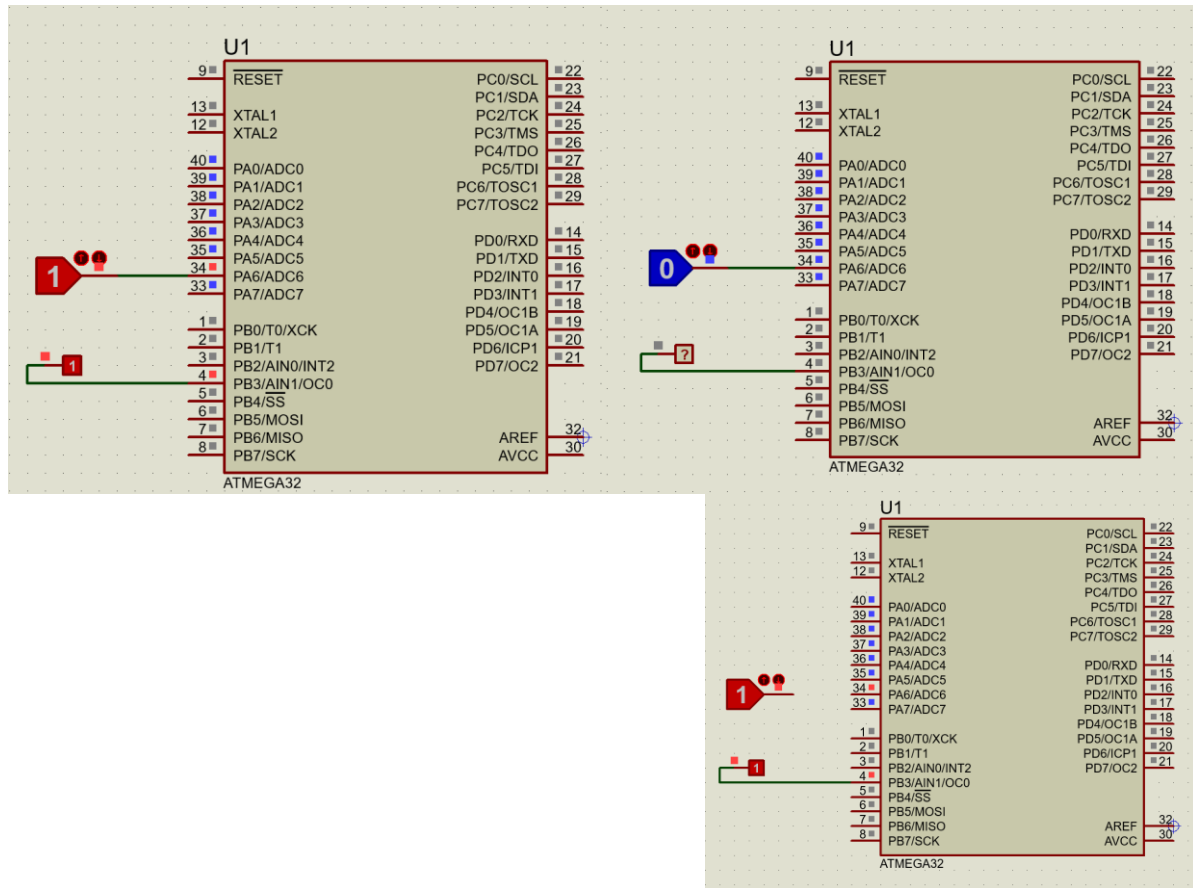
Status Register ☐ ☒ ☒ ☒ ☒ ☒ ☒ ☒

Name	Address	Value	Bits
IO PINB	0x36	0x00	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
IO DDRB	0x37	0x00	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
IO PORTB	0x38	0x08	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input checked="" type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>

Name	Address	Value	Bits
 PINA	0x39	0x00	       
 DDRA	0x3A	0xBF	       
 PORTA	0x3B	0x40	       

Status Register




Name	Address	Value	Bits
 PINB	0x36	0x00	       
 DDRB	0x37	0x00	       
 PORTB	0x38	0x00	       






۸- برنامه ای بنویسید که دائماً مقدار ورودی پورت C را معکوس و با ۵ جمع کند و در خانه ای با آدرسی به مقدار ورودی پورت B در SRAM ذخیره کند

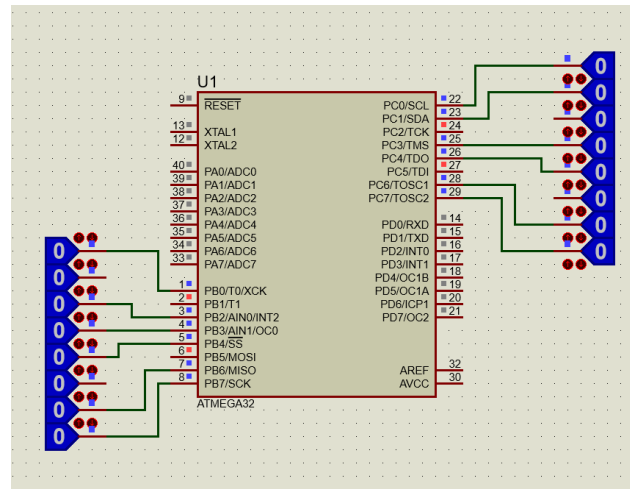
```
LDI R16,0X00
OUT DDRB,R16
OUT DDRC,R16
COM R16
OUT PORTB,R16 ;PULLUP
OUT PORTC,R16 ;PULLUP
start:
IN R16,PINC
COM R16
LDI R17,5
ADD R16,R17
IN R18,PINB
CLR ZL
ADD ZL,R18
ST Z,R16
rjmp start
```

R16 0x20
R17 0x05
R18 0xAA

Name	Address	Value	Bits
 PINB	0x36	0xAA	<input checked="" type="checkbox"/> <input type="checkbox"/> <input checked="" type="checkbox"/> <input type="checkbox"/> <input checked="" type="checkbox"/> <input type="checkbox"/> <input checked="" type="checkbox"/> <input type="checkbox"/>
 DDRB	0x37	0x00	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
 PORTB	0x38	0xFF	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/>

Name	Address	Value	Bits
 PINC	0x33	0xE4	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input checked="" type="checkbox"/> <input type="checkbox"/>
 DDRC	0x34	0x00	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
 PORTC	0x35	0xFF	<input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/> <input checked="" type="checkbox"/>

Memory 4			
Memory: prog FLASH			
data 0x00AA	20	00	00 00 00 00
data 0x00C3	00	00	00 00 00 00



۹- برنامه ای بنویسید که دائما اگر بیت ۶ ورودی پورت A صفر بود ورودی پورت B را مکمل ۲ کند و با ورودی پورت C جمع کند و حاصل را در پورت D را نشان دهد در غیر اینصورت اگر یک بود ورودی پورت B را از ورودی پورت C کم کند و حاصل را در پورت D را نشان دهد.(امتیازی)

```
LDI R16,0B10111111
OUT DDRA,R16
COM R16
OUT PORTA,R16
LDI R16,0X00
LDI R17,0XFF
OUT DDRB,R16
OUT PORTB,R17
OUT DDRC,R16
OUT PORTC,R17
```

```

OUT DDRD,R17
OUT PORTD,R16
start:
    IN R18,PINA
    IN R19,PINB
    IN R20,PINC
    BST R18,6
    BRTC FIRST
    BRTS SECOND

```

FIRST:

```

NEG R19
ADD R19,R20
OUT PORTD,R19
jmp start

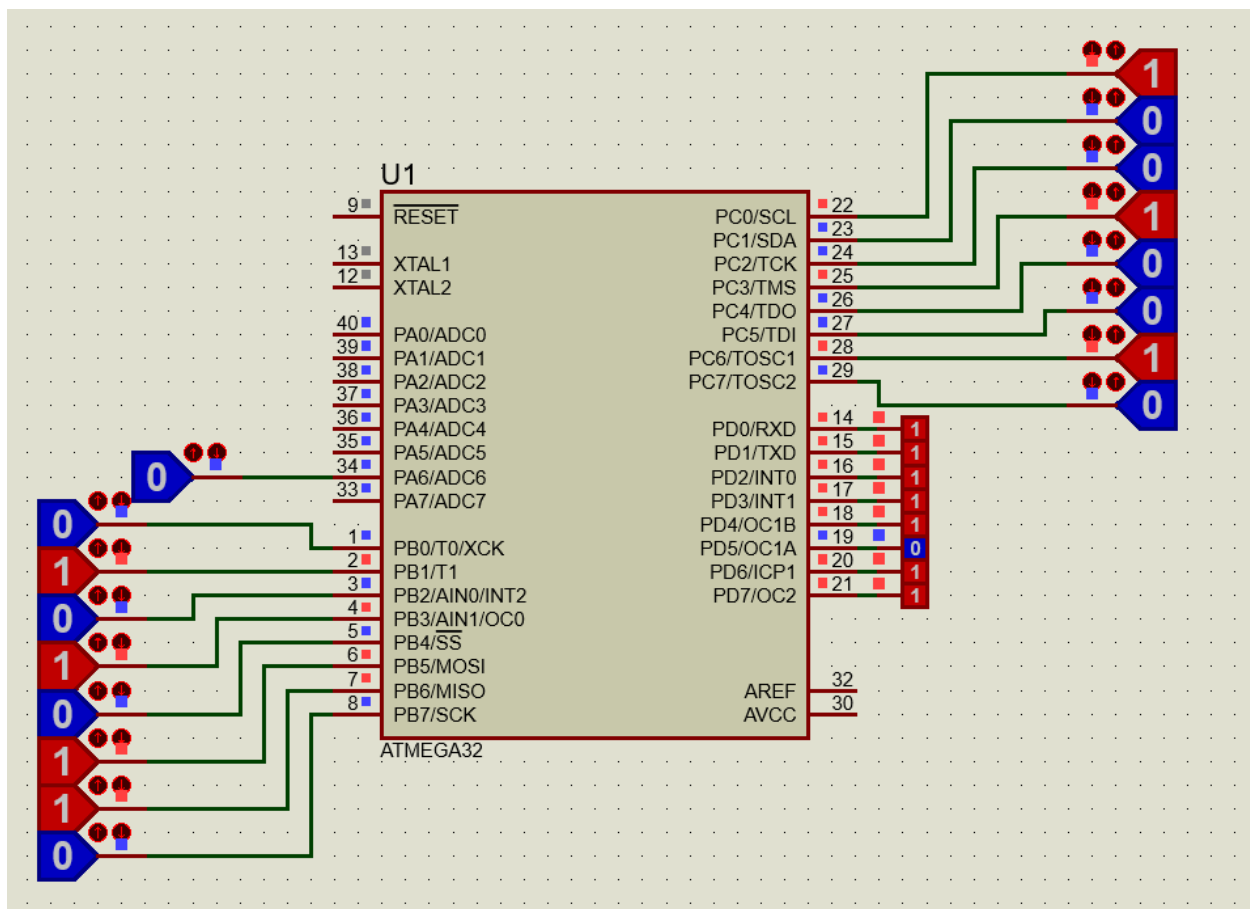
```

SECOND:

```

SUB R20,R19
OUT PORTD,R20
jmp start

```



با تشکر.