

실험제목 : ATmega328PB 어셈블리 언어 실습 (I)**실험목적**

ATmega328PB의 명령어의 동작을 이해하고 이들을 조합하여 간단한 프로그램을 작성하는 실습 과정을 통해 ATmega328PB의 내부 구조를 이해하고 어셈블리 언어로 프로그래밍하는 능력을 배양한다.

실험 준비물

Atmel Studio 7

실험에 필요한 예비지식

1. 실험에 필요한 자료
 - 1) 강의 노트 파일 참조
 - 2) [AVR-Instruction-Set-Manual](#)
 - 3) [AVR Assembler User Guide \(AVRASM2\)](#)

2. Assembly Language Program Format (AVRASM2)

1) 형식

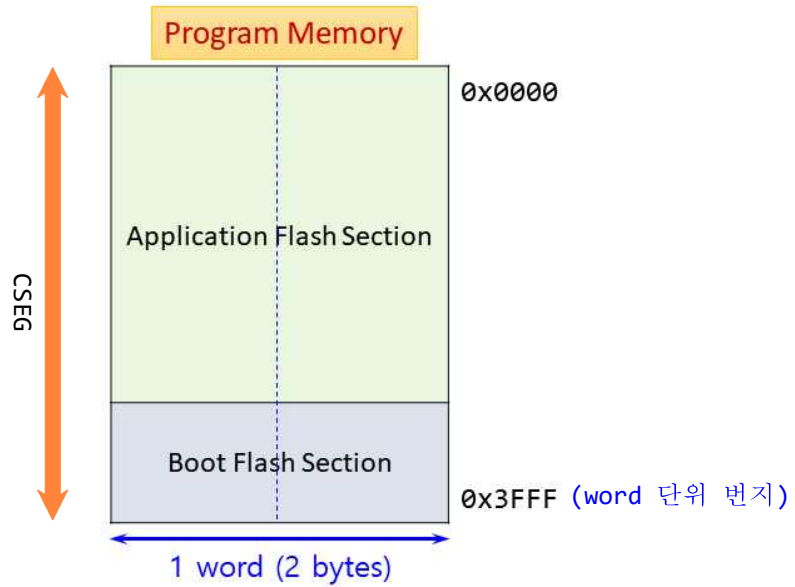
```
[label:] directive [operands] [Comment]
[label:] instruction [operands] [Comment]
Comment
Empty line
```

2) 설명

- 가) [] 부분은 생략 가능.
- 나) Label은 반드시 colon(:)으로 끝나야 함.
- 나) Comment는 반드시 semicolon(;)으로 시작해야 함.

3. Assembler Directives

(참고) ATmega328PB Memory Map



1) CSEG

가) 용도

Code segment의 시작을 알림.

Code segment만의 location counter를 가지며 byte counter가 아닌 word counter이다.

나) 형식

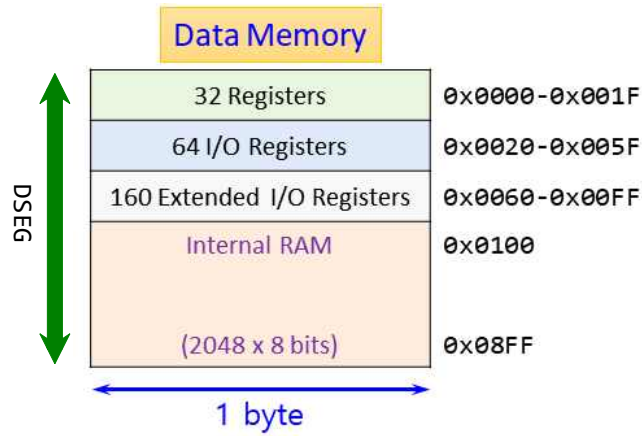
.CSEG

다) 예

```

.DSEG
.ORG    0x200 ; byte 단위 번지
BUF:   .BYTE  4
.CSEG
.ORG    0x100 ; word 단위 번지
START: LDI    R15,0x6F
...
    
```

2) DSEG



가) 용도

Data segment의 시작을 알림

Data segment만의 location counter를 가지며 byte counter이다.

나) 형식

.DSEG

다) 예

```

.DSEG
.ORG    0x200
BUF:   .BYTE  4
.CSEG
.ORG    0x100
START: LDI    R15, 0x6F
...
    
```

3) ORG

가) 용도

Location Counter를 지정된 값으로 초기화

나) 형식

`.ORG`

다) 예

```

.DSEG
.ORG    0x200 ; byte 단위 번지
BUF:   .BYTE  4
.CSEG
.ORG    0x100 ; word 단위 번지
START: LDI    R15,0x6F
...

```

4) BYTE

가) 용도

임시 데이터 저장용 공간을 확보하기 위해 Data Segment (SRAM) 영역에 지정된 바이트 수 만큼 메모리 공간을 확보

나) 형식

`.BYTE n`

다) 예

```

.DSEG
.ORG    0x200
BUF:   .BYTE  4
.CSEG
.ORG    0x100
START: LDI    R15,0x6F
...

```

5) DB, DW, DD, DQ

가) 용도

상수값을 저장하기 위해 Program Memory나 EEPROM 영역에 지정된 바이트(혹은 워드) 수 만큼 메모리 공간을 확보

나) 형식

.DB n

.DW n

다) 예

```
.DSEG
.ORG 0x200
BUF: .BYTE 4
.CSEG
.ORG 0x100
LED_TBL: .DB 0x7F, 48, 0b01010101, -128
WD_TBL: .DW 0x1234, 32764
START: LDI R15,0x6F
...
```

6) DEF

가) 용도

범용 레지스터에 다른 이름(symbolic name)을 부여할 때 사용.

나) 형식

```
.DEF Symbol = Register
```

다) 예

```
.DEF TEMP=R16
.DEF IOR=R0
LDI TEMP,0xF0 ; same as LDI R16,0xF0
IN IOR,0x3F ; Read SREG into IOR register
EOR TEMP,IOI ;
...
```

7) EQU

가) 용도

일반적으로 상수값을 정의하는데 사용

나) 형식

```
.EQU Symbol = expression
```

다) 예

```
.EQU LED=5
.EQU DDRB=0x04
.EQU PORTB=0x05
SBI DDRB,LED ; same as SBI DDRB,5
SBI PORTB,LED ; Set PB5 to HIGH
...
```

4. 실험에 필요한 명령어

1) Load Immediate

가) 형식

LDI Rd,K

나) 동작

$Rd \leftarrow K$ 단, $16 \leq d \leq 31$, $0 \leq K \leq 255$

다) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

라) 예

LDI R16,0xF5

2) Add without Carry

가) 형식

ADD Rd,Rr

나) 동작

$Rd \leftarrow Rd + Rr$ 단, $0 \leq d \leq 31$, $0 \leq r \leq 31$

다) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	↔	↔	↔	↔	↔	↔

라) 예

ADD R16,R17

3) Subtract without Carry

가) 형식

SUB Rd,Rr

나) 동작

$Rd \leftarrow Rd - Rr$ 단, $0 \leq d \leq 31,$ $0 \leq r \leq 31$

다) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	↔	↔	↔	↔	↔	↔

라) 예

SUB R16,R17

4) Multiply Unsigned

가) 형식

MUL Rd,Rr

나) 동작

$R1:R0 \leftarrow Rd \times Rr$ 단, $0 \leq d \leq 31,$ $0 \leq r \leq 31$

다) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	↔	↔

라) 예

MUL R16,R17

5) Store Direct to Data Space

가) 형식

STS addr16,Rr

나) 동작

$(addr16) \leftarrow Rr$ 단, $0 \leq r \leq 31, 0 \leq addr16 \leq 65535$

다) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

라) 예

STS 0x0102,R17

6) Relative Jump

가) 형식

RJMP k

나) 동작

$PC \leftarrow PC + k + 1$ 단, $-2,048 \leq k \leq +2,047$

다) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

라) 예

HERE: RJMP HERE

7) Add with Carry

가) 형식

ADC Rd,Rr

나) 동작

$Rd \leftarrow Rd + Rr + C$ 단, $0 \leq d \leq 31, \quad 0 \leq r \leq 31$

다) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	↔	↔	↔	↔	↔	↔

라) 예

ADC R16,R17

8) Subtract with Carry

가) 형식

SBC Rd,Rr

나) 동작

$Rd \leftarrow Rd - Rr - C$ 단, $0 \leq d \leq 31, \quad 0 \leq r \leq 31$

다) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	↔	↔	↔	↔	↔	↔

라) 예

SBC R16,R17

9) Load Program Memory

가) 형식 및 동작

```

LPM                R0 ← (Z)
LPM Rd,Z           Rd ← (Z)                단, 0 ≤ d ≤ 31
LPM Rd,Z+          Rd ← (Z), Z ← Z + 1     단, 0 ≤ d ≤ 31
    
```

나) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

다) 예

```

LDI    ZH,HIGH(TABLE << 1)    ; Initialize Z-pointer, R31
LDI    ZL,LOW(TABLE << 1)     ; R30
LPM    R16,Z+
HERE:  RJMP    HERE
.ORG   0x100
TABLE: .DB    0x13, 0x57, 0x9B, 0xDF
    
```

10) Store Indirect From Register to Data Space using Index X

가) 형식 및 동작

ST X,Rr	$(X) \leftarrow Rr$	단, $0 \leq r \leq 31$
ST X+,Rr	$(X) \leftarrow Rr, X \leftarrow X + 1$	단, $0 \leq r \leq 31$
ST -X,Rr	$X \leftarrow X - 1, (X) \leftarrow Rr,$	단, $0 \leq r \leq 31$

나) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

다) 예

```

.CSEG
LDI  XH,HIGH(ANS_4)      ; Initialize X-pointer, R27
LDI  XL,LOW(ANS_4)       ; R26
ST   X+,R16              ; Store data to SRAM pointed by X
HERE: Rjmp  HERE

.DSEG
.ORG  0x0100
ANS_4: .BYTE  4
    
```


12) Clear Register

가) 형식

CLR Rd

나) 동작

$Rd \leftarrow Rd \text{ XOR } Rd$ 단, $0 \leq d \leq 31$

다) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	-	0	0	0	1	-

라) 예

CLR R16

13) Decrement

가) 형식

DEC Rd

나) 동작

$Rd \leftarrow Rd - 1$ 단, $0 \leq d \leq 31$

나) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	-	↔	↔	↔	↔	-

라) 예

DEC R16

14) Increment

가) 형식

INC Rd

나) 동작

$Rd \leftarrow Rd + 1$ 단, $0 \leq d \leq 31$

나) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	-	↔	↔	↔	↔	-

라) 예

INC R16

15) Two's Complement

가) 형식

NEG Rd

나) 동작

$Rd \leftarrow 0x00 - Rd$ 단, $0 \leq d \leq 31$

나) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	↔	↔	↔	↔	↔	↔

라) 예

NEG R16

16) Copy Register

가) 형식

MOV Rd,Rr

나) 동작

$Rd \leftarrow Rr$ 단, $16 \leq d \leq 31,$ $16 \leq r \leq 31$

다) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

라) 예

LDI R16,R15

17) Bitwise OR

가) 형식

OR Rd,Rr

나) 동작

$Rd \leftarrow Rd \text{ OR } Rr$ 단, $0 \leq d \leq 31,$ $0 \leq r \leq 31$

다) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	-	↔	0	↔	↔	-

라) 예

OR R16,R17

18) Bitwise AND

가) 형식

AND Rd,Rr

나) 동작

$Rd \leftarrow Rd \text{ AND } Rr$ 단, $0 \leq d \leq 31, \quad 0 \leq r \leq 31$

다) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	-	↔	0	↔	↔	-

라) 예

AND R16,R17

19) Bitwise AND with Immediate

가) 형식

ANDI Rd,K

나) 동작

$Rd \leftarrow Rd \text{ AND } K$ 단, $16 \leq d \leq 31, \quad 0 \leq K \leq 255$

다) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	-	↔	0	↔	↔	-

라) 예

ANDI R16,0x0F

20) Set Bit in I/O Register

가) 형식

SBI A,b

나) 동작

$I/O(A,b) \leftarrow 1$ 단, $0 \leq A \leq 31,$ $0 \leq b \leq 7$

다) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

라) 예

SBI PORTB,5 ; Set PB5

21) Clear Bit in I/O Register

가) 형식

CBI A,b

나) 동작

$I/O(A,b) \leftarrow 0$ 단, $0 \leq A \leq 31,$ $0 \leq b \leq 7$

다) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

라) 예

CBI PORTB,5 ; Clear PB5

26) Subtract Immediate from Word

가) 형식

SBIW Rd+1,Rd,K

나) 동작

$R[d+1]:Rd \leftarrow R[d+1]:Rd - K$ $d \in \{24, 26, 28, 30\}$, $0 \leq K \leq 63$

다) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	-	↔	↔	↔	↔	↔

라) 예

SBIW R25:R24,1

SBIW R24,1 ; 위의 명령어와 같은 의미

27) Store Register to I/O Location

가) 형식

OUT A,Rr

나) 동작

$I/O(A) \leftarrow Rr$ 단, $0 \leq r \leq 31$, $0 \leq A \leq 63$

다) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

라) 예

OUT SPH,R16 ; Stack Pointer (SPH)

28) No Operation

가) 형식

NOP

나) 동작

없음

다) Status Register (SREG)

I	T	H	S	V	N	Z	C
-	-	-	-	-	-	-	-

라) 예

NOP

5. 상태 레지스터(Status Register, SREG)의 각 비트의 의미

Bit No.	7	6	5	4	3	2	1	0
Name	I	T	H	S	V	N	Z	C
Reset	0	0	0	0	0	0	0	0

Bit 7 - I: Global Interrupt Enable

Bit 6 - T: Copy Storage

Bit 5 - H: Half Carry Flag

Bit 4 - S: Sign Flag. $S=N \oplus V$.

Can be used to determine if the result of a previous operation is positive or negative.

Bit 3 - V: Two's Complement Overflow Flag

Bit 2 - N: Negative Flag. Same as bit 7 of previous operation.

Bit 1 - Z: Zero Flag

Bit 0 - C: Carry Flag

실습 내용

Microchip Studio의 **simulation** 기능을 사용하여 아래에 주어진 항목들을 실습하시오.

1. (8-비트 산술 연산) 아래에 주어진 상수들의 계산식을 수행하는 프로그램을 어셈블리 언어로 작성하시오. 연산 후 지정된 레지스터 및 SREG의 지정된 flag 값을 표시하시오.

1) (LDI, ADD 명령어 사용) $0xF5+0x0B$ 의 연산 결과를 R16에 저장.

R16 레지스터의 값: _____

SREG 값: H=__, S=__, V=__, N=__, Z=__, C=__

2) (LDI, SUB 명령어 사용) $0x82-0x7E$ 의 연산 결과를 R16에 저장.

R16 레지스터의 값: _____

SREG 값: H=__, S=__, V=__, N=__, Z=__, C=__

2. 16-비트 상수 $0x3CE7+0x3B8D$ 연산 결과의 하위 바이트는 내부 SRAM의 $0x100$ 번지에 저장하고, 상위 바이트는 내부 SRAM의 $0x101$ 번지에 저장하시오.

- .DSEG, .ORG, .BYTE, .CSEG directive 사용
- LDI, ADD, ADC, STS 명령어 사용

내부 SRAM의 $0x101$ 번지 내용: _____

내부 SRAM의 $0x100$ 번지 내용: _____

3. 16-비트 상수 0x4567 - 0x1289 연산 결과의 하위 바이트는 내부 SRAM의 0x102 번지에 저장하고, 상위 바이트는 내부 SRAM의 0x103 번지에 저장하시오.

- .DSEG, .ORG, .BYTE directive 사용
- LDI, SUB, SBC, STS 명령어 사용

내부 SRAM의 0x103 번지 내용: _____

내부 SRAM의 0x102 번지 내용: _____

과제

실습 내용 1, 2, 및 3의 실습 내용과 결과를 정리하여 보고서로 제출

-끝-