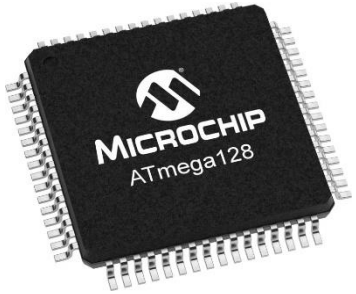
Lecture 07

EEPROM 제어

EEPROM

- Electrically Erasable Programmable Read-Only Memory
- 전기공급이 끊긴 상태에서도 장기간 기억하는 비휘발성 (non-volatile) 메모리
- 칩을 구성하는 소자의 전하를 전기적으로 변화시킴으로써 데이터를 기록하거나 지울 수 있음
- LCD패널, 전화기, 전자레인지, 자동세탁기 등 가전제품이나 민수용 통신기기에 많이 쓰임

단품
EEPROM



EEPROM은
MCU에 포함됨

ATmega128 EEPROM

- 4Kbyte (4096 bytes) 데이터 메모리
- 10만 번 이상 읽기/쓰기 가능
- EEPROM 레지스터
  - **어드레스 레지스터: EEARH 및 EEARL**
    - EEARH(하위 4비트), EEARL(8비트)를 합해서 12비트의 주소 영역 (2^{12} = 4096 bytes)에 걸쳐 8비트 데이터를 저장할 수 있음
  - **데이터 레지스터 : EEDR**
    - 읽기 : EEPROM에 읽혀진 데이터를 일시 보관함
    - 쓰기 : EEPROM에 쓸 데이터를 보관함
  - **컨트롤 레지스터 : EECR**
    - EERIE, EEMWE, EEWE, EERE의 4개의 제어비트를 통해 EEPROM을 제어하게 됨

EEPROM 레지스터

- EEARH 및 EEARL: EEPROM Address Register

Bit	15	14	13	12	11	10	9	8	
	–	–	–	–	EEAR11	EEAR10	EEAR9	EEAR8	EEARH
	EEAR7	EEAR6	EEAR5	EEAR4	EEAR3	EEAR2	EEAR1	EEAR0	EEARL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	X	X	X	X	
	X	X	X	X	X	X	X	X	

  - Bits 15..12 : reserved
  - Bits 11..0 : EEPROM 어드레스
    - EEARH(4비트) + EEARL(8비트) = 12비트 → 0~4095번지까지 4Kbyte EEPROM의 주소를 지정함
    - EEPROM을 접근하기 전에 **주소를 꼭 지정해 야 함**

EEPROM 레지스터

- EEDR: EEPROM Data Register

Bit	7	6	5	4	3	2	1	0	
	MSB							LSB	EEDR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

  - Bits 7..0 : EEPROM 데이터
    - 읽기 : EEAR에 지정된 주소의 EEPROM에 읽혀진 데이터를 일시 보관함
    - 쓰기 : EEAR에 지정된 주소의 EEPROM에 쓸 데이터를 보관함

EEPROM 레지스터

- EECR: EEPROM Control Register

Bit	7	6	5	4	3	2	1	0	
	–	–	–	–	EERIE	EEMWE	EEWE	EERE	EECR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	X	0	

  - Bits 7..4 : reserved
  - Bit 3 : **EERIE**(EEPROM Ready Interrupt Enable)
    - SREG의 7번 비트를 1로 설정한 다음 EERIE를 1로 설정하면 EEPROM Ready 인터럽트를 허용함
    - EEWE 비트가 0이면 안정된 인터럽트가 발생함
  - Bit 2 : EEMWE(EEPROM Master Write Enable)
  - Bit 1 : EEWE(EEPROM Write Enable)
  - Bit 0 : EERE(EEPROM Read Enable)

EEPROM 레지스터

- EECR: EEPROM Control Register

Bit	7	6	5	4	3	2	1	0	
	–	–	–	–	EERIE	EEMWE	EEWE	EERE	EECR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	X	0	

  - Bits 7..4 : reserved
  - Bit 3 : EERIE(EEPROM Ready Interrupt Enable)
  - Bit 2 : **EEMWE**(EEPROM Master Write Enable)
    - EEMWE를 1로 설정한 다음 4개의 clock cycle 이내 EEWE를 1로 설정하면 EEAR에 지정된 주소의 EEPROM에 데이터를 쓰게 됨
    - EEPROM에 데이터 쓰기가 완료되면 하드웨어가 EEMWE를 0로 설정해줌
  - Bit 1 : EEWE(EEPROM Write Enable)
  - Bit 0 : EERE(EEPROM Read Enable)

EEPROM 레지스터

- EECR: EEPROM Control Register

Bit	7	6	5	4	3	2	1	0	
	–	–	–	–	EERIE	EEMWE	EEWE	EERE	EECR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	X	0	

  - Bits 7..4 : reserved
  - Bit 3 : EERIE(EEPROM Ready Interrupt Enable)
  - Bit 2 : EEMWE(EEPROM Master Write Enable)
  - Bit 1 : **EEWE**(EEPROM Write Enable)
    - EEMWE를 1로 설정한 후 4개의 clock cycle 이내 EEWE를 1로 설정하면 EEAR에 지정된 주소의 EEPROM에 데이터를 쓰게 됨
  - Bit 0 : EERE(EEPROM Read Enable)

EEPROM 레지스터

- EECR: EEPROM Control Register

Bit	7	6	5	4	3	2	1	0	
	–	–	–	–	EERIE	EEMWE	EEWE	EERE	EECR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	X	0	

  - Bits 7..4 : reserved
  - Bit 3 : EERIE(EEPROM Ready Interrupt Enable)
  - Bit 2 : EEMWE(EEPROM Master Write Enable)
  - Bit 1 : EEWE(EEPROM Write Enable)
  - Bit 0 : **EERE**(EEPROM Read Enable)
    - EERE를 1로 설정하면 EEAR에 지정된 주소의 EEPROM에 데이터를 읽혀지며 읽혀진 데이터는 EEDR에 일시 보관됨
    - 읽기 명령을 시키기 전에 EEWE가 0이 될 때까지 기다려 야 함

EEPROM 레지스터

- SPMCSR: Store Program Memory Control and Status Register

Bit	7	6	5	4	3	2	1	0	
	SPMIE	RWWSB	–	RWWSRE	BLBSET	PGWRT	PGERS	SPMEN	SPMCSR
Read/Write	R/W	R	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

  - Bit 0: **SPMEN**(Store Program Memory Enable)
    - 프로그램 메모리 활성화 저장이 비트는 4개의 clock cycle 동안 SPM 명령을 활성화함
    - SPMEN 비트는 SPM 명령이 완료되면 자동으로 0이 됨

EEPROM에 쓰기 절차

1. EEWE 비트가 0이 될 때까지 기다림
2. SPMSCR 레지스터의 SPMEN 비트가 0이 될 때까지 기다림
3. EEAR에 EEPROM 주소를 씀 (선택사항)
4. EEDR에 EEPROM 데이터를 씀 (선택사항)
5. EEMWE 비트를 1로, EEWE 비트를 0로 설정함
6. 4개의 clock cycle 이내 EEWE 비트를 1로 설정함

EEPROM에 쓰기 함수

Bit	7	6	5	4	3	2	1	0	
	–	–	–	–	EERIE	EEMWE	EEWE	EERE	EECR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	X	0	

```
void EEPROM_write(unsigned int  addr,
                  unsigned char data
) {
    while (EECR & (1<<EEWE));
    EEAR = addr;
    EEDR = data;
    EECR |= (1<<EEMWE);
    EECR |= (1<<EEWE);
}
```

EEPROM 주소 및 데이터

EEWE 비트가 0이 될 때까지 기다림
1<<EEWE = 1<<1 = 0000_0010

EEAR에 EEPROM 주소를 씀
EEDR에 EEPROM 데이터를 씀

EEMWE 비트를 1로 설정함
1<<EEMWE = 1<<2 = 0000_0100

EEWE 비트를 1로 설정함

EEPROM에 읽기 함수

Bit	7	6	5	4	3	2	1	0	
	–	–	–	–	EERIE	EEMWE	EEWE	EERE	EECR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	X	0	

```
unsigned char EEPROM_read(unsigned int addr)
{
    while (EECR & (1<<EEWE));
    EEAR = addr;
    EECR |= (1<<EERE);
    return EEDR;
}
```

EEPROM 주소

EEWE 비트가 0이 될 때까지 기다림

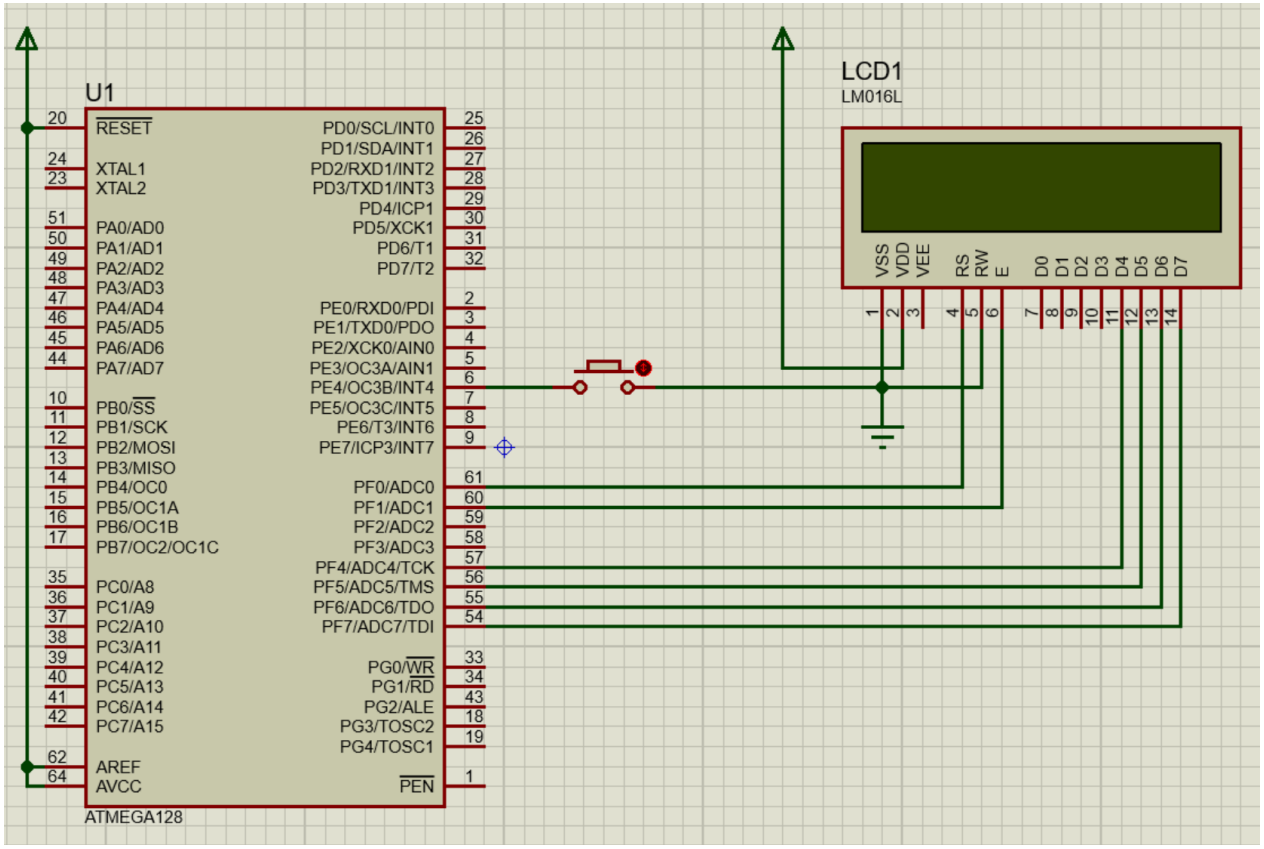
EEAR에 EEPROM 주소를 씀

EERE 비트를 1로 설정함

읽혀서 EEDR에 일시 저장된 데이터를 가지고 감

실습1: EEPROM 데이터 쓰고 읽기

- Proteus



실습1: EEPROM 데이터 쓰고 읽기

- ATmega128을 처음 사용할 때 EEPROM에 어떤 값이 들어 있었는지 모름

→EEPROM을 처음 사용한 지 확인해 야 함

- 확인 절차
  1. 지정된 주소(0x0000, 0x0001)에 코드(‘1’, ‘2’)를 저장함
  2. 프로그램이 시작할 때 0x0000과 0x0001 주소에 어떤 값이 들어 있는지 확인
  3. 저장한 코드(‘1’, ‘2’)가 아니면 EEPROM을 처음 사용하는 것임
  4. 저장한 코드(‘1’, ‘2’)가 이면 EEPROM을 처음 사용하는 것이 아님

실습1: EEPROM 데이터 쓰고 읽기

- 구현할 프로그램
  - 버튼 누름 횟수를 LCD에 표시함
  - 아이디어
    - EEPROM의 0x0002 주소에 버튼 누름 횟수에 대한 변수를 저장함
      - 처음 사용할 때 EEPROM의 0x0002 주소에 0을 씀
    - 버튼 누르면 인터럽트 발생
      - 버튼 누름 인터럽트 발생할 때마다 버튼 누름 횟수 하나씩 증가
      - 업데이트 된 누름 횟수를 EEPROM의 0x0002 주소에 저장함

실습1: EEPROM 데이터 쓰고 읽기

- 코드
  - EEPROM을 처음 사용한 지 확인 코드

```
if (check_EEPROM())
    data_EEPROM = EEPROM_read(addr_EEPROM);
else {
    set_EEPROM();
    data_EEPROM = 0;
}
```

0x0002

0x0002에 저장된 버튼 누름 횟수를 읽음

EEPROM에 EEPROM을 처음 사용한 지 확인 코드를 씀
- 0x0000에 '1'을 씀
- 0x0001에 '2'를 씀

0x0002에 0을 씀(처음 사용해서 버튼을 누른 적이 없음)

실습1: EEPROM 데이터 쓰고 읽기

- 코드
  - EEPROM을 처음 사용한 지 확인 코드

```
if (check_EEPROM())
   data_EEPROM = EEPROM_read(addr_EEPROM);
else {
   set_EEPROM();
   data_EEPROM = 0;
}
```

```
char check_EEPROM() {
   char data0, data1;
   data0 = EEPROM_read(0x0000);
   delay(1);
   data1 = EEPROM_read(0x0001);
   if (data0=='1' && data1=='2')
      return 1;
   else
      return 0;
}
```

```
void set_EEPROM() {
   EEPROM_write(0x0000, '1');
   delay(1);
   EEPROM_write(0x0001, '2');
   delay(1);
   EEPROM_write(0x0002, 0x00);
}
```

실습1: EEPROM 데이터 쓰고 읽기

- 코드
  - 버튼 누름 인터럽트 체크하고 누름 횟수 증가

```
while(1) {
   interruptNo = EIFR;
   if (interruptNo==0x10) {
      interruptNo = 4;
      EIFR |= 0x10;

      data_EEPROM++;
      EEPROM_write(0x0002, data_EEPROM);

      sprintf(msg, "Current row: %d", data_EEPROM);
      writeString_lcd4(0, 0, msg);

      sprintf(msg, "%d checked ", interruptNo);
      interruptNo = 0;
      writeString_lcd4(0, 1, msg);
   }
}
```

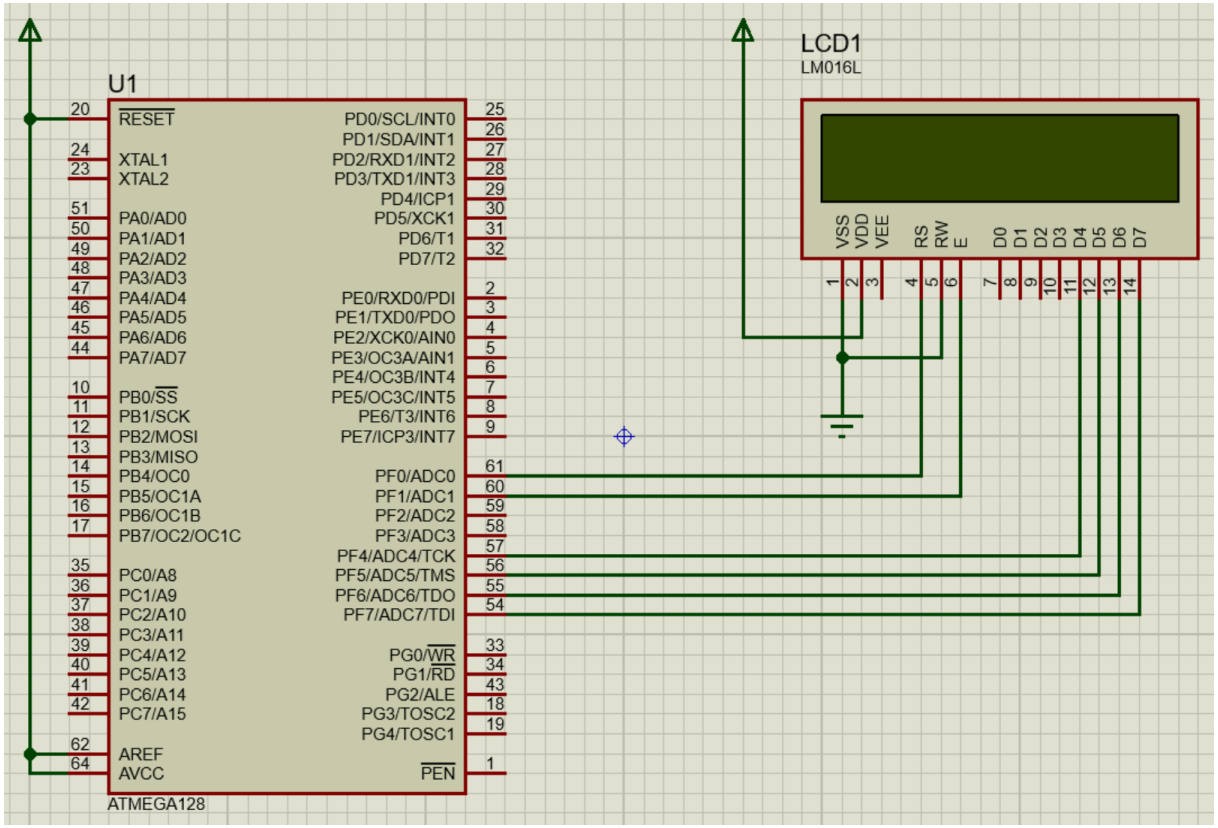
실습1: EEPROM 데이터 쓰고 읽기

- 전체 코드

```
#include <xc.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include "stdio.h"
#include "lcd4.h"

void EEPROM_write(unsigned int, unsigned char);
unsigned char EEPROM_read(unsigned int);
char check_EEPROM(); void set_EEPROM();

int main(void) {
   unsigned int addr_EEPROM = 0x0002;
   unsigned char data_EEPROM, interruptNo = 0;
   char msg[20];

   DDRF = 0xff;  // LCD 제어 포트
   DDRE = 0x00;  // PORTE 입력 모드
   PORTE = 0xff; // PORTE pull-up 설정
   EIMSK = 0x10; // INT4 인터럽트 허용
   EICRB = 0xd2; // falling edge 인터럽트 trigger
   init_lcd4();
   if (check_EEPROM())
      data_EEPROM = EEPROM_read(addr_EEPROM);
   else {
      set_EEPROM(); data_EEPROM = 0;
   }
   sprintf(msg, "Current row: %d", data_EEPROM);
   writeString_lcd4(0, 0, msg); delay(1);
   writeString_lcd4(0, 1, "Program restart");
   while(1) {
      interruptNo = EIFR;
      if (interruptNo==0x10) {
         interruptNo = 4; EIFR |= 0x10;
         data_EEPROM++;
         EEPROM_write(0x0002, data_EEPROM);
         sprintf(msg, "Current row: %d", data_EEPROM);
         writeString_lcd4(0, 0, msg);
         sprintf(msg, "%d checked ", interruptNo);
         interruptNo = 0;
         writeString_lcd4(0, 1, msg);
      }
   }
}
```

실습2: EEPROM 인터럽트

- Proteus



실습2: EEPROM 인터럽트

- 구현할 프로그램
  - EEPROM Ready 인터럽트를 사용하여 EEPROM에 데이터를 쓰고 읽음
  - EEPROM Ready 인터럽트를 사용하기 위해 다음과 같이 설정해 야 함
    1. SREG 레지스터의 7번째 비트를 1로 설정함
    2. EECR 레지스터의 EERIE 비트를 1로 설정함
  - EEPROM Ready 인터럽트의 ISR를 통해 EEPROM에 데이터를 씀

```
char *addr, *pData;
ISR(EE_READY_vect) {
   if (*pData)
      EEPROM_write((unsigned int)addr++, *pData++);
   else
      EECR &= 0xf7;
}
```

EEPROM Ready 인터럽트 백터

모두 데이터를 EEPROM에 쓰고 나서 *pData가 NULL이 되므로 EERIE 비트를 0로 설정하고 ISR에 나감

실습2: EEPROM 인터럽트

- 전체 코드

```
#include <xc.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include "stdio.h"
#include "string.h"
#include "lcd4.h"

void EEPROM_write(unsigned int,
                  unsigned char);
unsigned char EEPROM_read(unsigned int);

char *addr, *pData;

ISR(EE_READY_vect) {
   if (*pData)
      EEPROM_write((unsigned int)addr++,
                   *pData++);
   else
      EECR &= 0xf7;
}
```

```
int main(void) {
   char noMsg, msg[] = "Welcome";

   DDRF = 0xff;
   init_lcd4();

   pData = msg;
   addr = 0x0000;
   noMsg = sizeof(msg);

   SREG |= 0x80;
   EECR |= (1<<EERIE);

   while (EECR & (1<<EEWE));
   strcpy(msg, "Goodbye");
   addr = 0x0000;
   while (EECR & (1<<EEWE));
   for (int i=0; i<noMsg; i++)
      msg[i] = EEPROM_read((unsigned int)addr++);
   writeString_lcd4(0, 1, msg);
   while(1);
}
```