9주차

2017. 04. 26.

- 이번 학기 강의내용

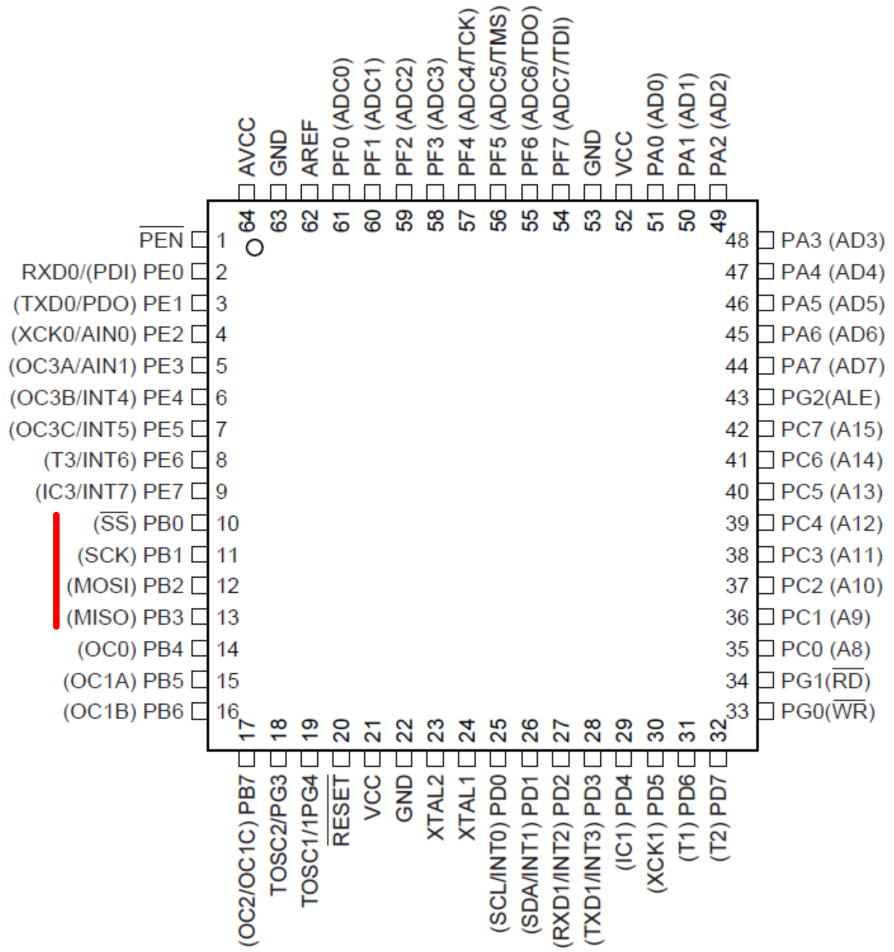
주	주제	강의내용
1	지난학기 review	지난학기에 배운 마이크로컨트롤러의 기본 기능에 대한한 review
2	모터 1	모터 구동 이론, DC 모터
3	모터 2	STEP 모터
4	LCD 1	Character Liquid Crystal Display 기본 실습
5	LCD 2	Character Liquid Crystal Display 응용 + 4x4 키패드
6	무선통신 1	적외선(Infrared) 통신 기본
7	무선통신 2	적외선(Infrared) 통신 remote controller제작
8	중간고사	중간고사
9	데이터변환 1	SPI 통신 (Digital-to-Analog Converter)
10	데이터변환 2	데이터 변환 응용(DAC 출력-음악 만들기)
11	데이터변환 3	타이머 카운터 응용(음악 만들기)
12	데이터변환 4	아날로그-디지털 변환기 (ADC) 아날로그 컴퍼레이터
13	센서 인터페이스 1	온도센서 압력센서 기울기센서
14	센서 인터페이스 2	광센서(cds) 포토인터럽터 텀 프로젝트 기안(1인 1 프로젝트, 졸업 작품과 연계 금지)
15	텀 프로젝트	텀 프로젝트 중간 점검
16	기말고사	텀 프로젝트 발표 및 시연

ATMEGA128의 주요 특징

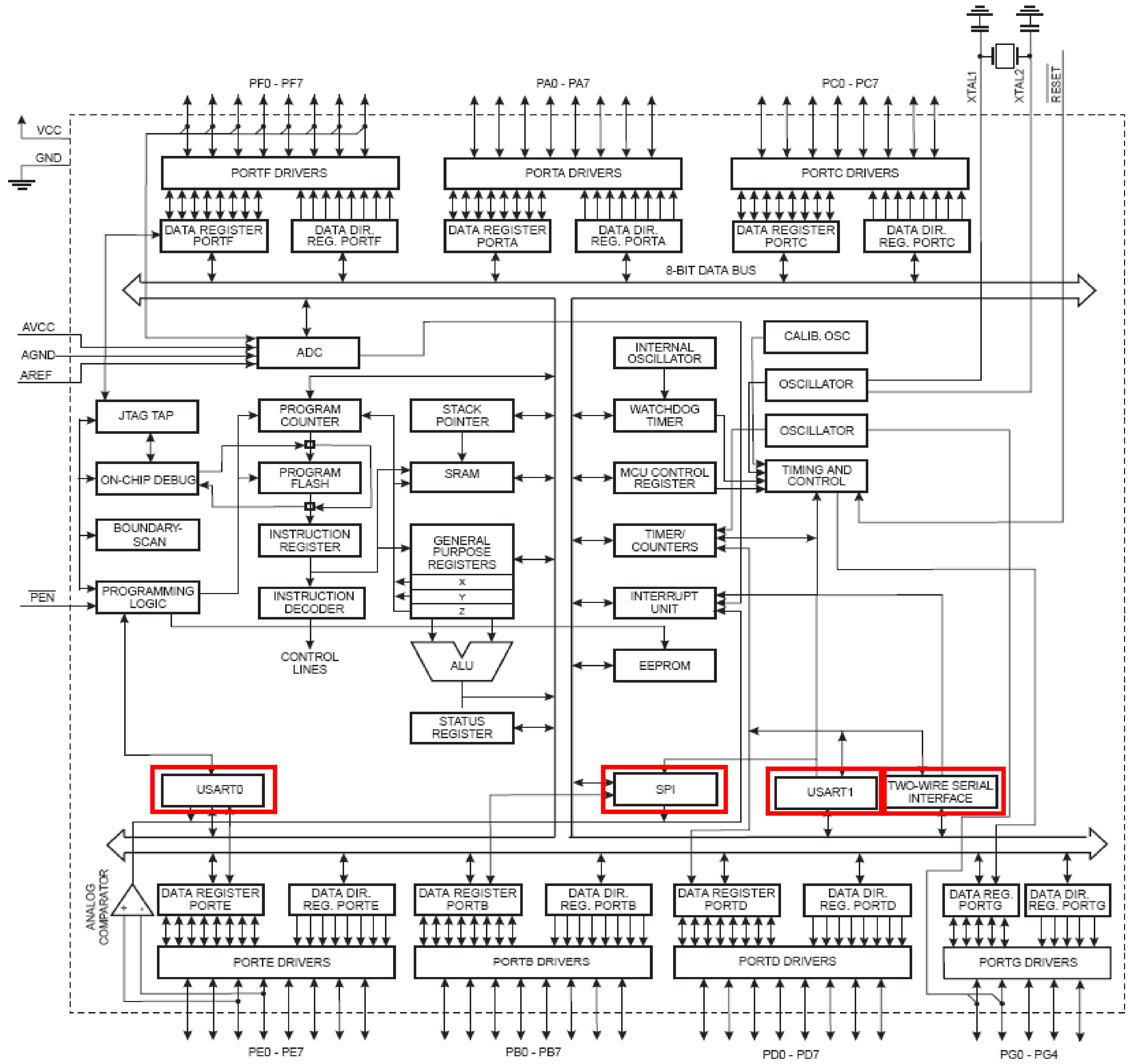
- High-performance, Low-power AVR® 8-bit Microcontroller
- Advanced RISC Architecture
  - 133 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers + Peripheral Control Registers
  - Fully Static Operation
  - Up to 16 MIPS Throughput at 16 MHz
  - On-chip 2-cycle Multiplier
- Nonvolatile Program and Data Memories
  - 128K Bytes of In-System Reprogrammable Flash
    Endurance: 10,000 Write/Erase Cycles
  - Optional Boot Code Section with Independent Lock Bits
    In-System Programming by On-chip Boot Program
    True Read-While-Write Operation
  - 4K Bytes EEPROM
    Endurance: 100,000 Write/Erase Cycles
  - 4K Bytes Internal SRAM
  - Up to 64K Bytes Optional External Memory Space
  - Programming Lock for Software Security
  - SPI Interface for In-System Programming
- JTAG (IEEE std. 1149.1 Compliant) Interface
  - Boundary-scan Capabilities According to the JTAG Standard
  - Extensive On-chip Debug Support
  - Programming of Flash, EEPROM, Fuses and Lock Bits through the JTAG Interface
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes
  - Two Expanded 16-bit Timer/Counters with Separate Prescaler, Compare Mode and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Two 8-bit PWM Channels
  - 6 PWM Channels with Programmable Resolution from 2 to 16 Bits
  - Output Compare Modulator
  - 8-channel, 10-bit ADC
    8 Single-ended Channels
    7 Differential Channels
    2 Differential Channels with Programmable Gain at 1x, 10x, or 200x
  - Byte-oriented Two-wire Serial Interface
  - Dual Programmable Serial USARTs
  - Master/Slave SPI Serial Interface
  - Programmable Watchdog Timer with On-chip Oscillator
  - On-chip Analog Comparator

- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated RC Oscillator
  - External and Internal Interrupt Sources
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
  - Software Selectable Clock Frequency
  - ATmega103 Compatibility Mode Selected by a Fuse
  - Global Pull-up Disable
- I/O and Packages
  - 53 Programmable I/O Lines
  - 64-lead TQFP and 64-pad MLF
- Operating Voltages
  - 2.7 - 5.5V for ATmega128L
  - 4.5 - 5.5V for ATmega128
- Speed Grades
  - 0 - 8 MHz for ATmega128L
  - 0 - 16 MHz for ATmega128

Figure 1. Pinout ATmega128



교재 15-20 페이지 참고



PF0 - PF7
PA0 - PA7
PC0 - PC7
XTAL1
XTAL2
RESET
VCC
GND
PORTF DRIVERS
PORTA DRIVERS
PORTC DRIVERS
DATA REGISTER PORTF
DATA DIR. REG. PORTF
DATA REGISTER PORTA
DATA DIR. REG. PORTA
DATA REGISTER PORTC
DATA DIR. REG. PORTC
8-BIT DATA BUS
AVCC
AGND
AREF
ADC
CALIB. OSC
INTERNAL OSCILLATOR
OSCILLATOR
OSCILLATOR
JTAG TAP
PROGRAM COUNTER
STACK POINTER
WATCHDOG TIMER
ON-CHIP DEBUG
PROGRAM FLASH
SRAM
MCU CONTROL REGISTER
TIMING AND CONTROL
BOUNDARY-SCAN
INSTRUCTION REGISTER
GENERAL PURPOSE REGISTERS
X
Y
Z
TIMER/ COUNTERS
PEN
PROGRAMMING LOGIC
INSTRUCTION DECODER
INTERRUPT UNIT
CONTROL LINES
ALU
EEPROM
STATUS REGISTER
USART0
SPI
USART1
TWO-WIRE SERIAL INTERFACE
ANALOG COMPARATOR
DATA REGISTER PORTE
DATA DIR. REG. PORTE
DATA REGISTER PORTB
DATA DIR. REG. PORTB
DATA REGISTER PORTD
DATA DIR. REG. PORTD
DATA REG. PORTG
DATA DIR. REG. PORTG
PORTE DRIVERS
PORTB DRIVERS
PORTD DRIVERS
PORTG DRIVERS
PE0 - PE7
PB0 - PB7
PD0 - PD7
PG0 - PG4

SPI 통신 개념도

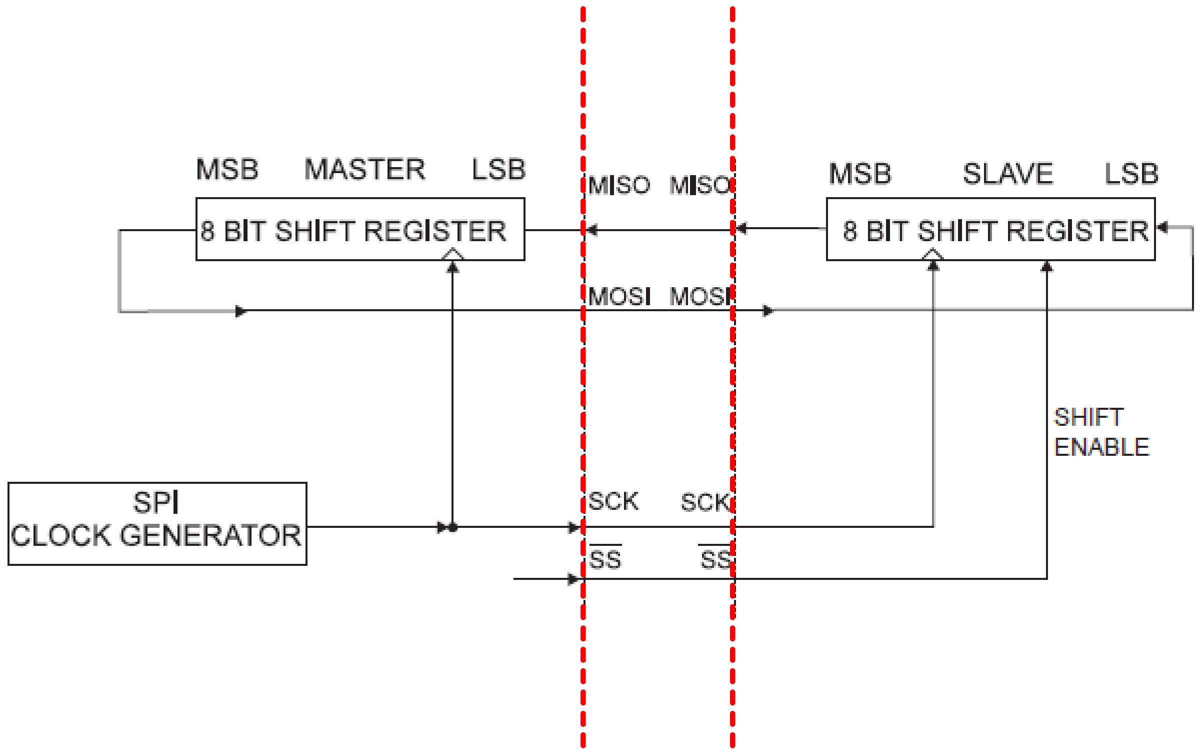
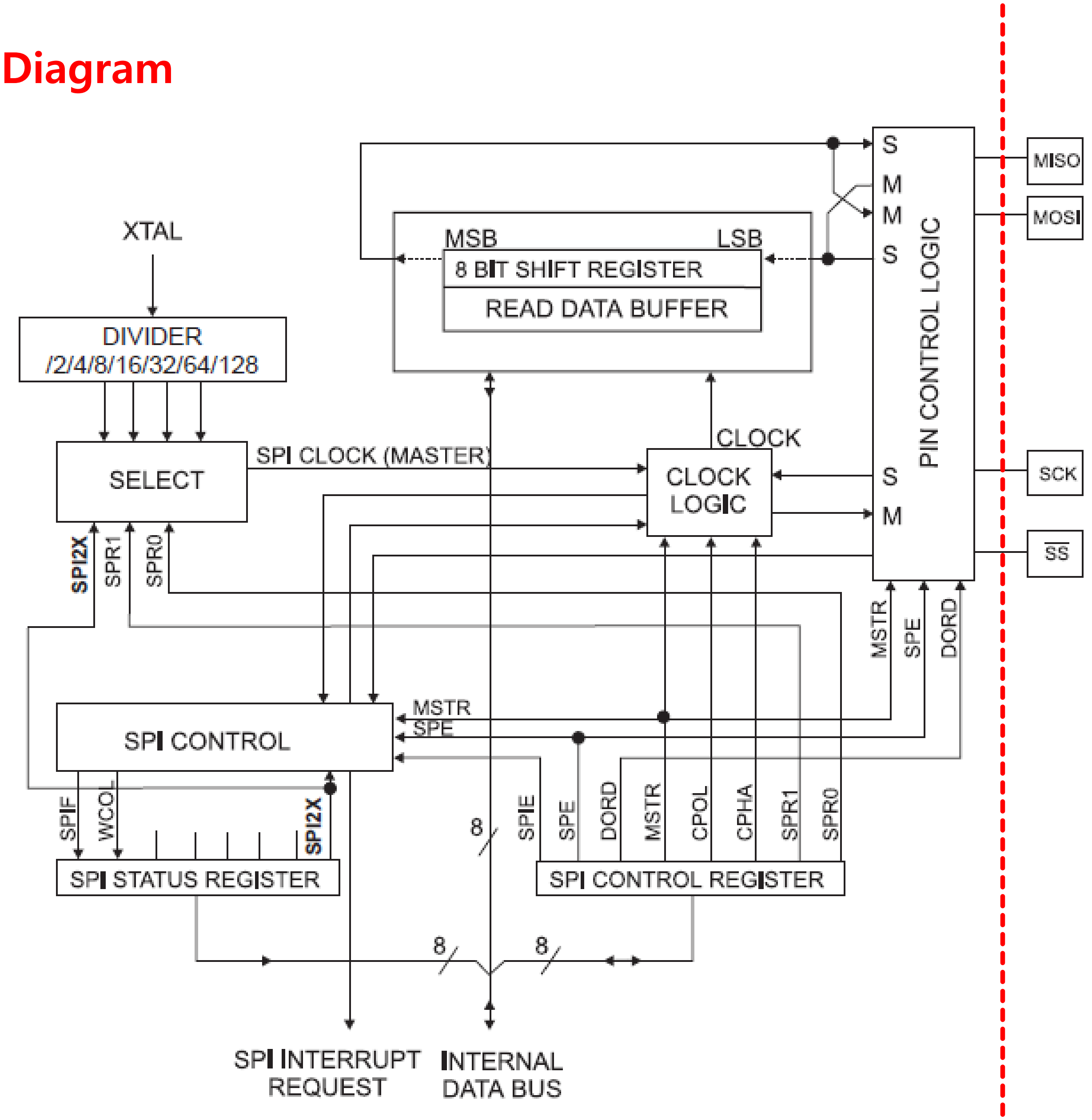


Table 69. SPI Pin Overrides(1)

Pin	Direction, Master SPI	Direction, Slave SPI
MOSI	User Defined	Input
MISO	Input	User Defined
SCK	User Defined	Input
$\overline{SS}$	User Defined	Input

SPI Block Diagram



SPI Registers

SPI Control Register – SPCR

Bit	7	6	5	4	3	2	1	0	
	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0	SPCR
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	

SPI Status Register – SPSR

Bit	7	6	5	4	3	2	1	0	
	SPIF	WCOL	–	–	–	–	–	SPI2X	SPSR
Read/Write	R	R	R	R	R	R	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

SPI Data Register – SPDR

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

SPI 통신의 주요동작

① SPI 관련 레지스터 설정
- SPI PORT 입/출력방향
- MASTER / SLAVE
- 통신속도
- 인터럽트 활성화(enable)

② SLAVE SELECT (/SS) 단자 LOW 출력

③ MASTER 로부터 Serial Clock (SCK) 발생

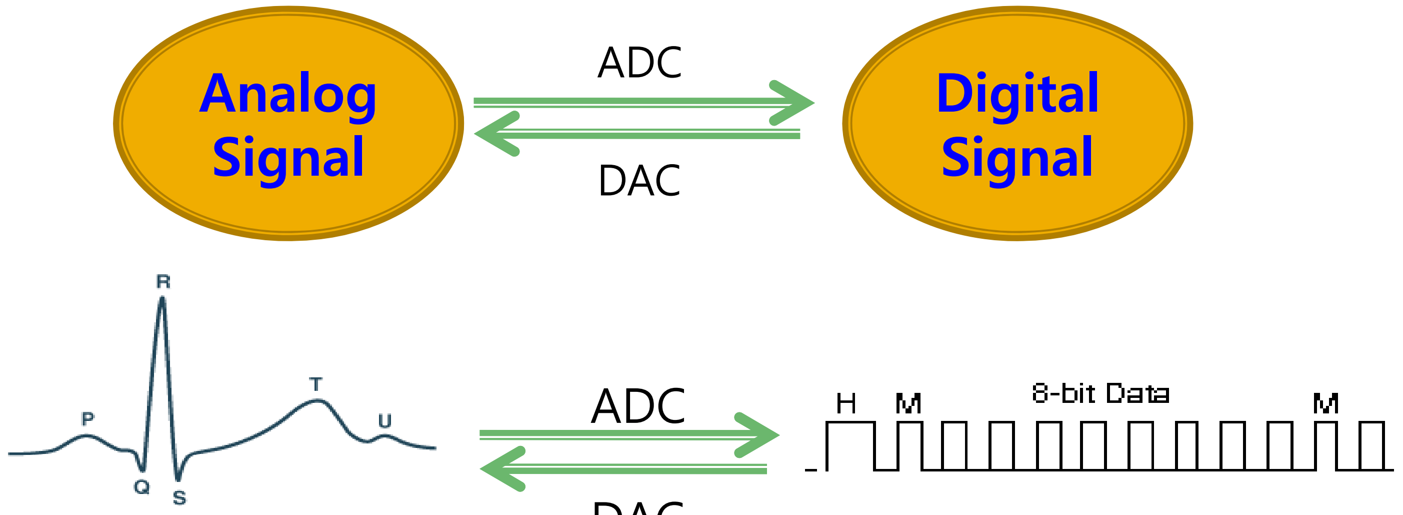
④ MASTER 의 데이터가 SLAVE 로 입력됨 (MOSI → MISO)

⑤ 8 bits (= 1 byte)의 데이터 전송 완료 후 전송 완료 플래그 (SPIF) 발생

⑥ 데이터 처리 후 ②~⑤의 동작 반복

DAC

Digital-to-Analog Converter



Analog ECG signal

Digitized ECG signal

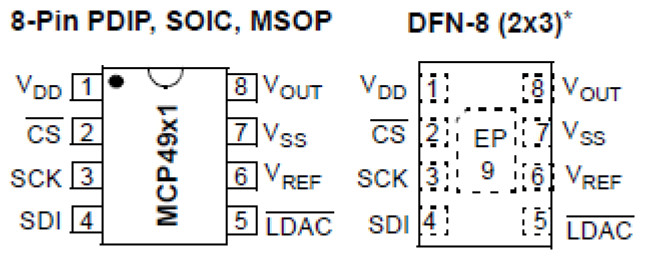
MCP4901

8/10/12-Bit Voltage Output Digital-to-Analog Converter with SPI Interface

Features

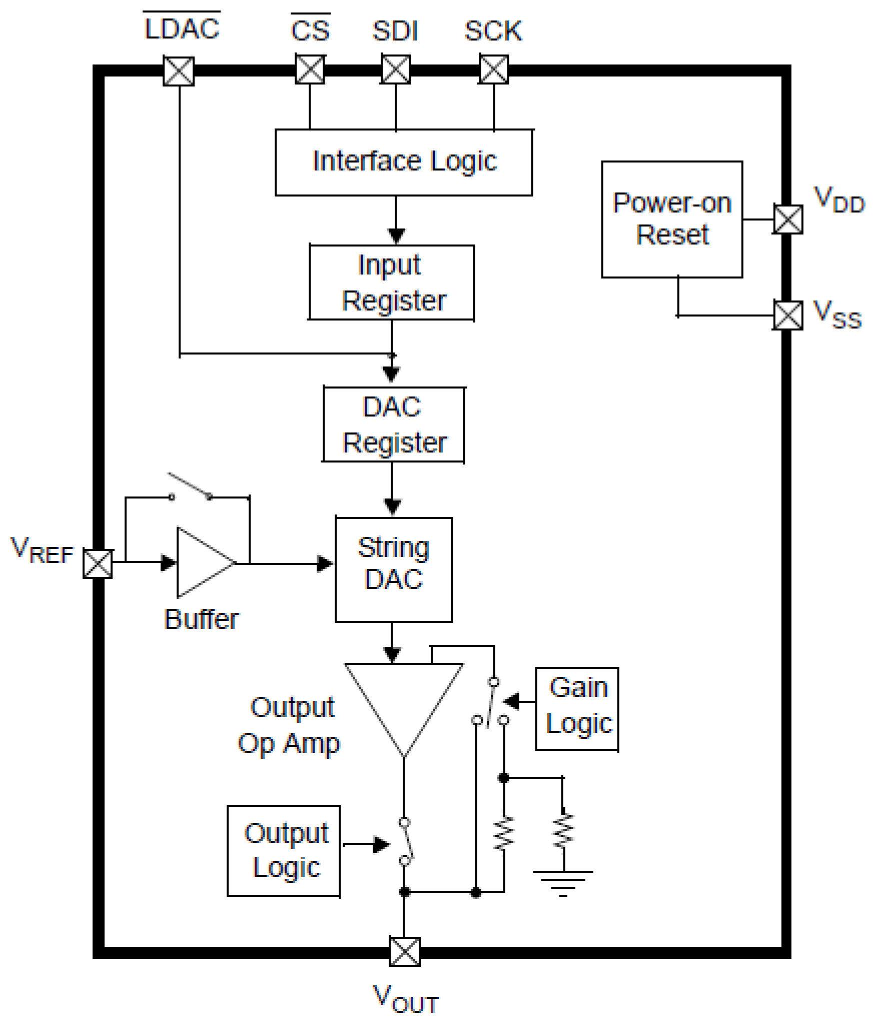
- MCP4901: 8-Bit Voltage Output DAC
- MCP4911: 10-Bit Voltage Output DAC
- MCP4921: 12-Bit Voltage Output DAC
- Rail-to-Rail Output
- SPI Interface with 20 MHz Clock Support
- Simultaneous Latching of the DAC Output with $\overline{\text{LDAC}}$ Pin
- Fast Settling Time of 4.5 µs
- Selectable Unity or 2x Gain Output
- External Voltage Reference Input
- External Multiplier Mode
- 2.7V to 5.5V Single-Supply Operation
- Extended Temperature Range: -40°C to +125°C

Package Types



MCP4901: 8-bit single DAC
MCP4911: 10-bit single DAC
MCP4921: 12-bit single DAC

* Includes Exposed Thermal Pad (EP); see Table 3-1.



MCP4901의 내부 구조

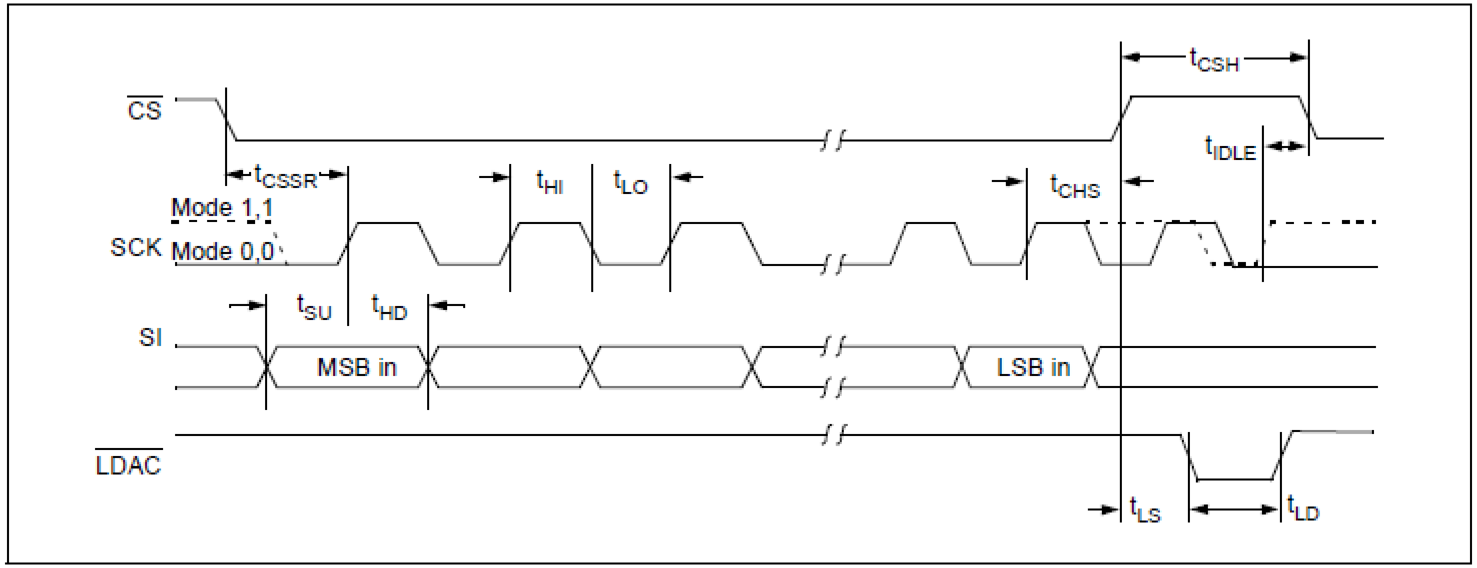


FIGURE 1-1: *SPI Input Timing Data.*

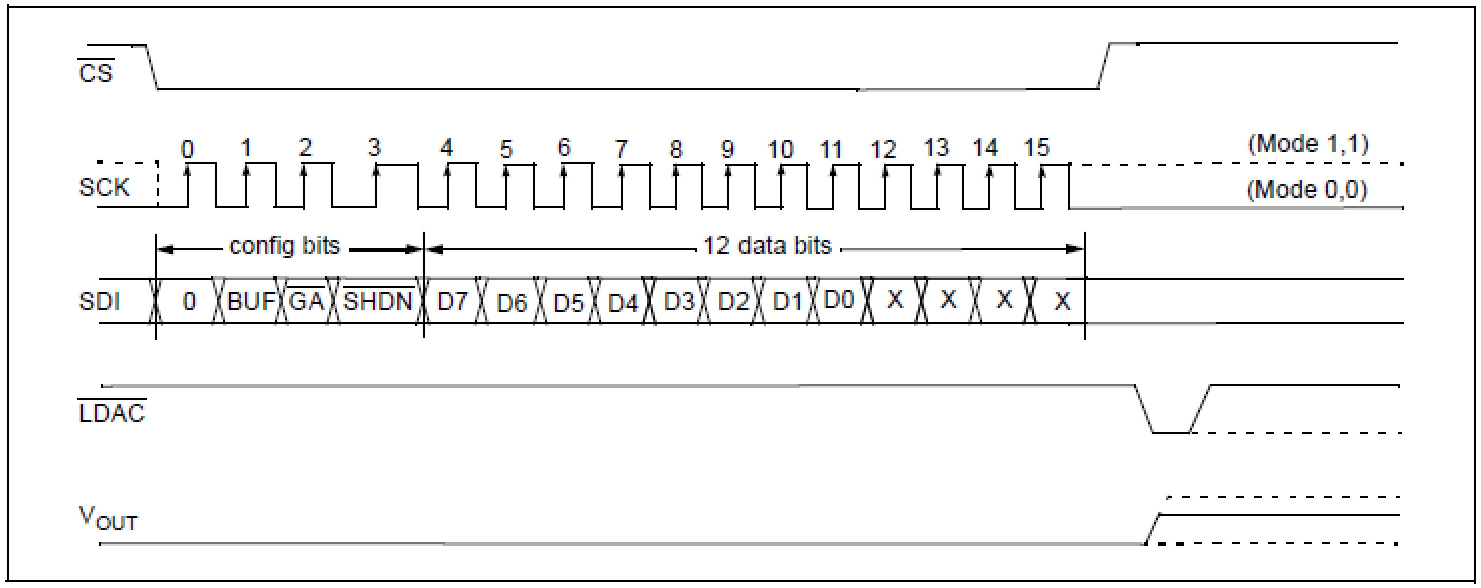
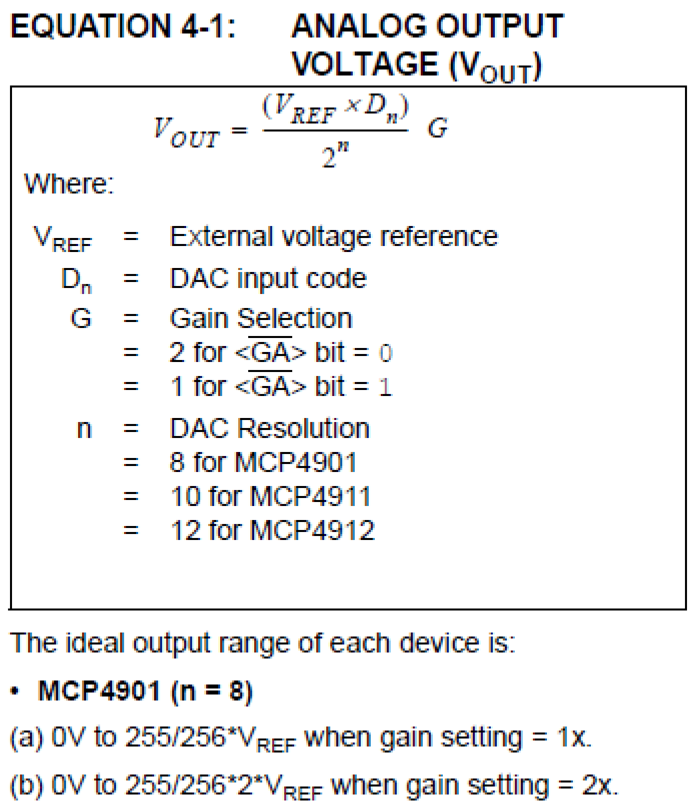


FIGURE 5-3: *Write Command for MCP4901(8-bit DAC). Note: X are don't care bits.*



EQUATION 4-1: ANALOG OUTPUT VOLTAGE (V_{OUT})

$$V_{OUT} = \frac{(V_{REF} \times D_n)}{2^n} G$$

Where:

V_{REF}	=	External voltage reference
D_n	=	DAC input code
G	=	Gain Selection
	=	2 for <$\overline{GA}$> bit = 0
	=	1 for <$\overline{GA}$> bit = 1
n	=	DAC Resolution
	=	8 for MCP4901
	=	10 for MCP4911
	=	12 for MCP4912

The ideal output range of each device is:

- **MCP4901 (n = 8)**

(a) 0V to 255/256*V_{REF} when gain setting = 1x.

(b) 0V to 255/256*2*V_{REF} when gain setting = 2x.

★1 Volts 출력하고자 할 때의 계산

$$1 = \frac{(5 \times D_n)}{2^8} \times G$$

G = 1 이라면 $D_n = 51$

G = 2 라면 $D_n = 25$

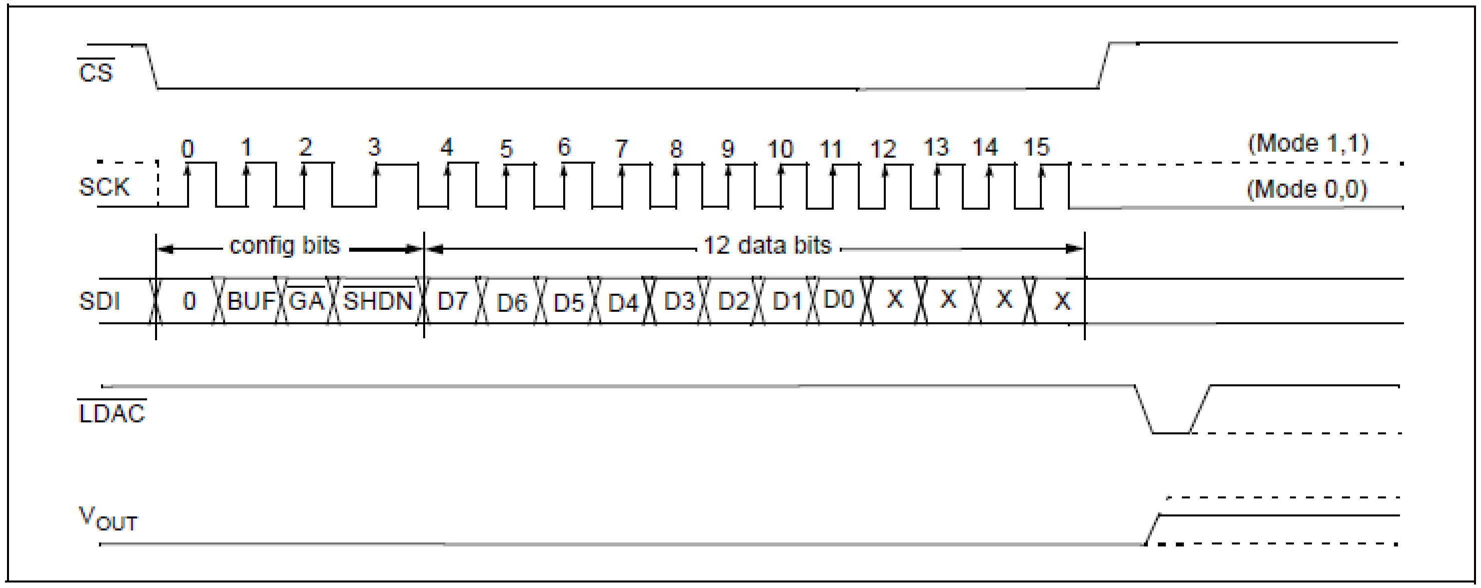


FIGURE 5-3: *Write Command for MCP4901(8-bit DAC). Note: X are don't care bits.*

```
void SPI_Trans()
{
  data = voltage[i];
  PORTB &= 0xfe;                    // 통신 시작 (ss low)

  // 16bit 전송 , Vout = Vr * G * Dn / 2^8 ( Dn = 0xff = 255 , Vr = 4.8)
  SPDR = 0x70 | (data >> 4);                    // Buffer On, Gain 1, DAC On
  while(!(SPSR & 0x80));           // 송신 확인
  SPDR = 0x00 | (data << 4);
  while(!(SPSR & 0x80));

  PORTB |= 0x01;                   // 통신 종료 (ss high)

  PORTE &= 0xfe;                   // LDCA low
  PORTE |= 0x01;                   // LDCA high
  __delay_cycles(100);
}
```

REGISTER 5-3: WRITE COMMAND REGISTER FOR MCP4901 (8-BIT DAC)

W-x	W-x	W-x	W-0	W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x
0	BUF	$\overline{\text{GA}}$	$\overline{\text{SHDN}}$	D7	D6	D5	D4	D3	D2	D1	D0	x	x	x	x
bit 15															bit 0

Where:

bit 15	0 = Write to DAC register 1 = Ignore this command
bit 14	**BUF:** V_{REF} Input Buffer Control bit 1 = Buffered 0 = Unbuffered
bit 13	**$\overline{\text{GA}}$:** Output Gain Selection bit 1 = 1x ($V_{OUT} = V_{REF} * D/4096$) 0 = 2x ($V_{OUT} = 2 * V_{REF} * D/4096$)
bit 12	**$\overline{\text{SHDN}}$:** Output Shutdown Control bit 1 = Active mode operation. VOUT is available. 0 = Shutdown the device. Analog output is not available. VOUT pin is connected to 500 kΩ (typical).
bit 11-0	**D11:D0:** DAC Input Data bits. Bit x is ignored.

Legend			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	1 = bit is set	0 = bit is cleared	x = bit is unknown

예제 1

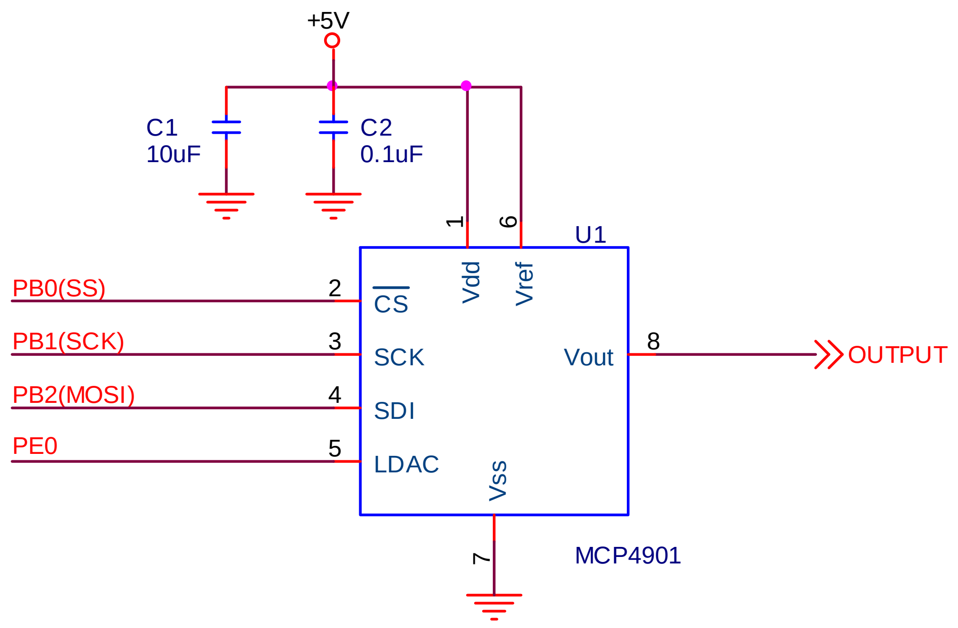
```
#include<iom128.h>
#include<intrinsics.h>

unsigned char voltage[5] = { 0, 53, 106, 159, 213};
unsigned char data, i;

void SPI_Trans();

#pragma vector=INT4_vect  // Voltage Up interrupt
__interrupt void INT4_interrupt(void)
{
  __disable_interrupt();

  i++;
  if(i >= 4) i = 4;

  SPI_Trans();

  __delay_cycles(100000);
  __enable_interrupt();
}

#pragma vector=INT5_vect  // Voltage Down interrupt
__interrupt void INT5_interrupt(void)
{
  __disable_interrupt();

  i--;
  if(i <= 0) i = 0;

  SPI_Trans();
  __delay_cycles(100000);
  __enable_interrupt();
}

void SPI_Trans()
{
  data = voltage[i];
  PORTB &= 0xfe;                    // 통신 시작 (ss low)

  // 16bit 전송 , Vout = Vr * G * Dn / 2^8 ( Dn = 0xff = 255 , Vr = 4.8)
  SPDR = 0x70 | (data >> 4);                    // Buffer On, Gain 1, DAC On
  while(!(SPSR & 0x80));          // 송신 확인
  SPDR = 0x00 | (data << 4);
  while(!(SPSR & 0x80));

  PORTB |= 0x01;                    // 통신 종료 (ss high)

  PORTE &= 0xfe;                    // LDCA low
  PORTE |= 0x01;                    // LDCA high
  __delay_cycles(100);
}

void main()
{
  DDRD = 0x0f, PORTD = 0x3f;  // Key Setting

  DDRB = 0xf7, PORTB = 0x01;  // SS SCK MOSI 출력 , MISO 입력 , SS set
  DDRE = 0x0f, PORTE = 0x01;  // LDCA 출력 , LDCA set

  SPCR = 0x50;              // SPE, Mastar Mode, SPI Mode0, SCK=fosc/4

  EIMSK = 0x30;
  EICRB = 0x0A;
  EIFR = 0x30;
  SREG = 0x80;
}
```

예제 2

```
#include<iom128.h>
#include<intrinsics.h>

unsigned char ECG[100] = {???????????????};
unsigned char data = 0, i = 0;

void SPI_Trans();

#pragma vector=INT4_vect  // Voltage Up interrupt
__interrupt void INT4_interrupt(void)
{
  __disable_interrupt();

  for(int k = 0; k < 100; k++)
  {
    SPI_Trans();
    __delay_cycles(20000);
    i++;
    if(i >= 116) i = 0;
  }
  __enable_interrupt();
}
```

```
void SPI_Trans()
{
  data = ECG[i];
  PORTB &= 0xfe;                    // 통신 시작 (ss low)

  // 16bit 전송 , Vout = Vr * G * Dn / 2^8 ( Dn = 0xff = 255 , Vr = 4.8)
  SPDR = 0x70 | (data >> 4);                    // Buffer On, Gain 1, DAC On
  while(!(SPSR & 0X80));           // 송신 확인
  SPDR = 0x00 | (data << 4);
  while(!(SPSR & 0X80));

  PORTB |= 0x01;                    // 통신 종료 (ss high)

  PORTE &= 0xfe;                    // LDCA low
  PORTE |= 0x01;                    // LDCA high
  __delay_cycles(100);
}

void main()
{
  DDRD = 0x0f, PORTD = 0x3f;  // Key Setting

  DDRB = 0xf7, PORTB = 0x01;  // SS SCK MOSI 출력 , MISO 입력 , SS set
  DDRE = 0x0f, PORTE = 0x01;  // LDCA 출력 , LDCA set
  SPCR = 0x50;                // SPE, Mastar Mode, SPI Mode0, SCK=fosc/4

  EIMSK = 0x30;
  EICRB = 0x0A;
  EIFR = 0x30;
  SREG = 0x80;
}
```

실험 전 절차



+5V
C1
10uF
C2
0.1uF
U1
MCP4901
PB0(SS)
PB1(SCK)
PB2(MOSI)
PE0
$\overline{CS}$
SCK
SDI
LDAC
Vdd
Vref
Vss
Vout
OUTPUT
1
2
3
4
5
6
7
8

Experiments – SPI (DAC 제어)

- **실습**

1. MCP4901 회로 납땜
2. 예제 1번 실습 및 이해
3. 예제 2번 구현