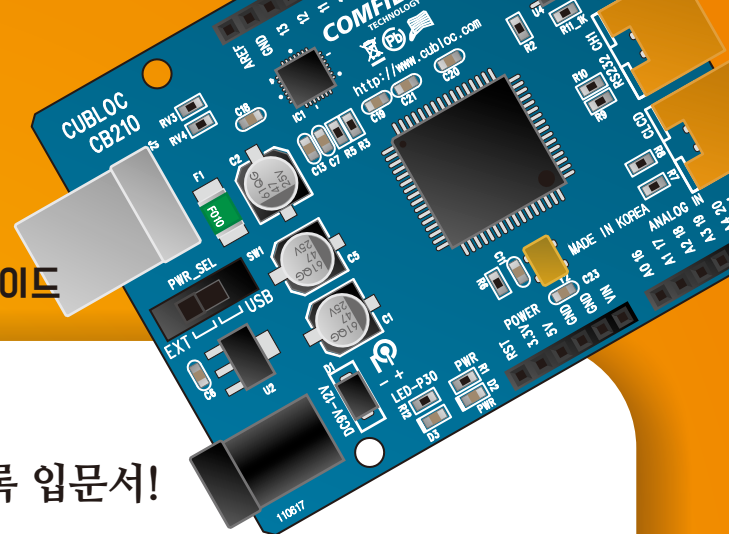




CB210을 처음 배우는 분들을 위한 초보자 가이드



초보자들이 사용하기 편리한 CB210 중심의 큐블록 입문서!

# CB210 가이드북

컴파일 테크놀로지 주식회사



**COMFILE**  
TECHNOLOGY  
[www.comfile.co.kr](http://www.comfile.co.kr)

## 본 책은...

큐블록을 처음 입문하시는 분들을 위한 책입니다.

초보자들이 사용하기 편리한 CB210 중심으로 되어 있어, 누구나 쉽게 큐블록의 세계로 입문하실 수 있습니다.

중, 고등학생들의 교재로 사용할 수 있도록 그림과 만화를 사용하여 쉽게 설명하였으며, 예제중심으로 되어 있습니다.

## CB210은...

베이직 언어를 사용하는  
임베디드 컨트롤러입니다.

임베디드 콘트롤분야를  
처음 배우고자하는 분들이

사용하기 편하도록  
만들어진 제품입니다.

본 교재와 함께 공부하시면  
보다 쉽게 임베디드 콘트롤  
분야에 입문하실 수 있습니다.



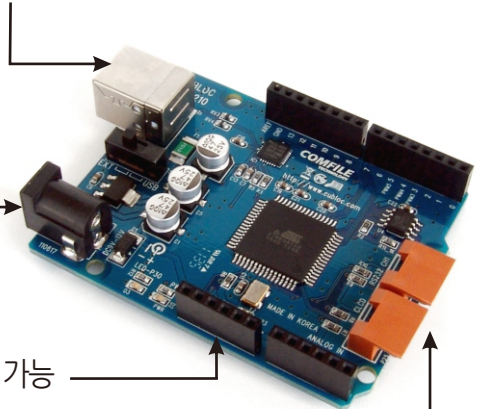
USB로 프로그램을 다운로드

USB연결 시 전원은 PC에서 공급됨

USB연결이 없는 경우,  
별도로 전원 입력 가능

점퍼선으로 I/O결선 가능

LCD와 7 세그먼트 모듈  
사용 가능



# CONTENTS

## 1장 만화로 배우는 큐블록 임베디드 컨트롤의 세계

- 6 임베디드 컨트롤을 배우자!
- 7 임베디드 컨트롤이란?
- 10 응용 분야
- 13 CB210이란?

## 2장 큐블록이란 무엇인가?

- 17 CUBLOC 사용하기

## 3장 베이직 프로그램

- 28 LAB 1. CUBLOC으로 LED 켜기
- 36 LAB 2. 변수의 사용
- 38 LAB 3. 입력 포트를 이용한 LED 제어

- 40 LAB 4. AD 값 입력 받기
- 46 LAB 5. PWM을 이용한 LED 제어
- 50 LAB 6. LCD를 이용한 데이터 출력
- 54 LAB 7. 키패드 입력
- 58 LAB 8. PC와의 RS232 통신

## 4장 레더로직 프로그램

- 66 레더로직프로그램이란?
- 67 레더 사용 방법
- 74 논리 회로
- 80 자기 유지 회로
- 82 인터록 회로
- 84 타이머 회로
- 86 카운터 회로
- 88 베이직과 레더의 연동



# CONTENTS

## 5장 응용 예제

- 94 키패드를 이용한 계산기
- 98 PWM을 이용한 전자 피아노
- 100 AD를 이용한 RC서보모터 제어
- 102 스위치 입력을 이용한 릴레이 동작
- 106 스텝모터 컨트롤
- 110 DC팬모터를 이용한 선풍기

# 01

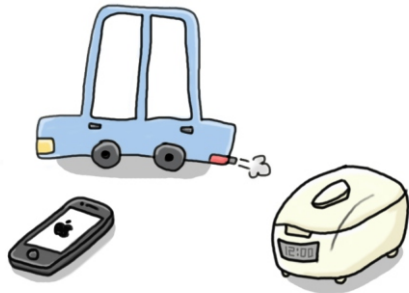
## 만화로 배우는 큐블록 임베디드 컨트롤의 세계

- 임베디드 컨트롤을 배우자!
- 임베디드 컨트롤이란?
- 응용 분야
- CB210이란?



# 임베디드 컨트롤을 배우자!

임베디드 컨트롤은 곧 “내장제어”를 의미합니다.  
어떤 기기나 장치속에 들어가서 그 장치를 제어하는  
컨트롤러를 만드는 기술이라고 보시면 됩니다.



세상은 나날이 발전해가고 있고, 임베디드 컨트롤을 필요로 하는 분야는 점점 더 늘어나고 있습니다.



그만큼,  
이 분야에 많은 일자리가 생겨나고 있다는 것을 뜻하죠.



일거리는 많은데  
개발자가 부족해  
고민이야.



뭔가 사회에서  
필요로 하는 분야를  
공부할걸 그랬나?  
쩍...

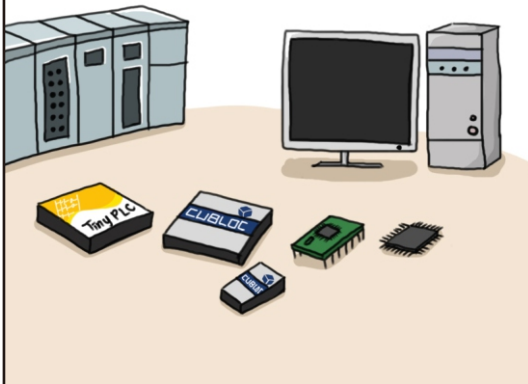


# 임베디드 컨트롤이란?

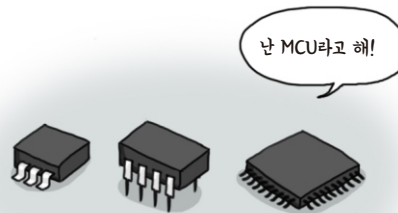
임베디드 컨트롤의 분야는 매우 광범위하죠.  
가전제품, 자동차, 공장 자동화, 스마트폰, POS분야 등등 이루 셀 수 없이 많아요.  
우리는 수도 없이 많은 임베디드 컨트롤 제품에 둘러싸여 살고 있다고 해도  
과언이 아니에요.



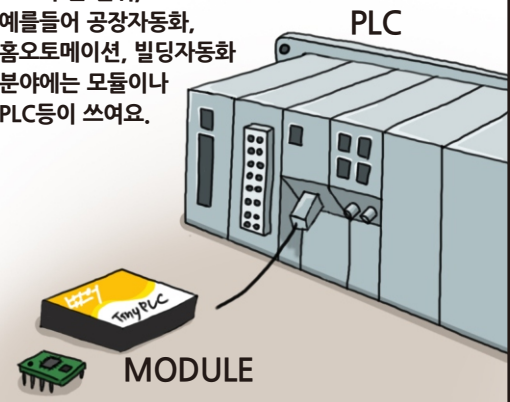
임베디드 콘트롤에 쓰이는 콘트롤러는 매우 다양해요.



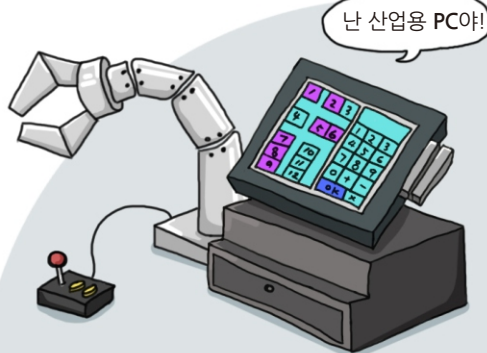
우선, 아주 작은 제어에는 마이크로 콘트롤러 라고 불리우는 반도체가 쓰이죠.



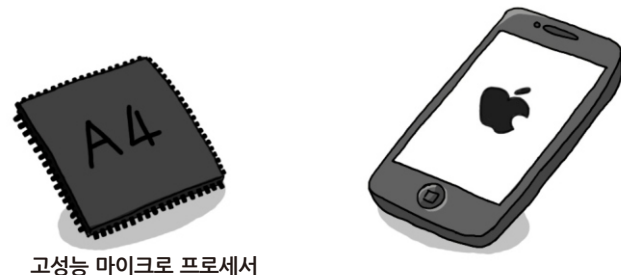
그보다 큰 단위, 예를들어 공장자동화, 홈오토메이션, 빌딩자동화 분야에는 모듈이나 PLC등이 쓰여요.



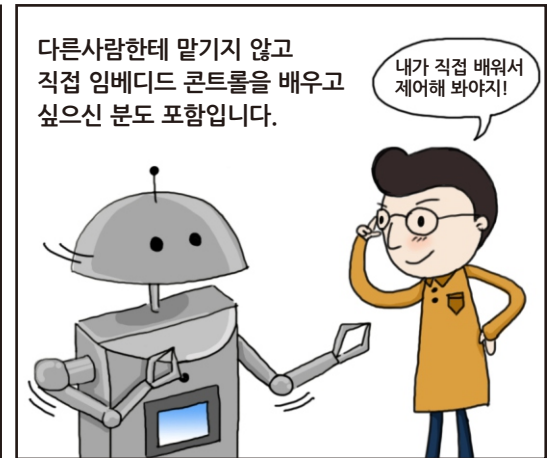
로봇제어나 POS분야에는 PC가 쓰이죠. 이런 곳에 쓰이는 PC는 우리가 알고있는 PC와는 조금 다르게 생겼답니다.



스마트폰에도 마이크로 프로세서 반도체가 들어가 있지요. 크기는 작지만 아주 고성능 칩입니다.



고성능 마이크로 프로세서





## 응용분야

우선, 여러분이 임베디드 컨트롤을 배우면 어떤 일을 하게 되는지 설명해 드릴게요. 우리 집안은 물론, 공장, 빌딩, 음식점, 대형마트 등 생활주변에 많은 분야에서 임베디드 컨트롤이 쓰이고 있다는 것은 앞에서 설명했는데요.



자동차 한 대에도  
대략 20여개의 MCU가  
쓰이고 있어요.

여러분이 상상하시는 것보다도 훨씬 많은 분야에서 응용이 되고 있어요.

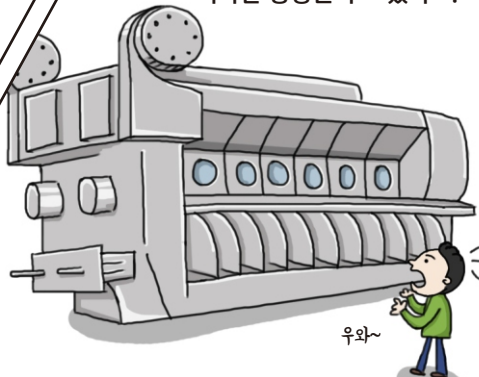


엄마!  
이 장난감이  
말을 해요!

리모컨처럼 작은 제품이  
있는가 하면,



선박에 들어가는 엔진처럼  
커다란 응용분야도 있지요.



우와~

회사마다 전문 분야가 있지요. 여러분이 어떤 회사에서 일하게 되느냐에 따라서 어떤걸 만들게 될지가 결정되는게 일반적이지요.



앞으로 보일러를  
만들게 될거야!

소형 경보기부터 전기밥솥, 모니터, 공장 자동화 분야까지 정말 다양한 어플리케이션을 경험해보았던 베테랑 엔지니어들도 항상 그 핵심로직(코어부분)은 별 차이가 없다는 것을 알고 있습니다.



따라서 여러분도 임베디드 콘트롤 엔지니어가 되고 나면, 나중에 무엇을 만들게 될지 즐거운 상상을 해보세요! 혹시 모르죠, 여러분이 우리나라 최초의 우주왕복선을 만들게 될지도요!

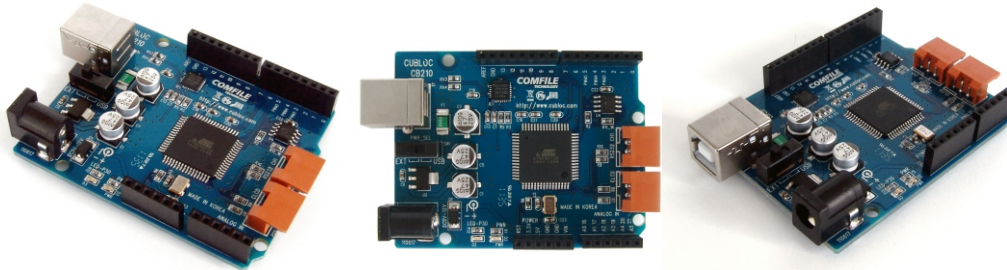




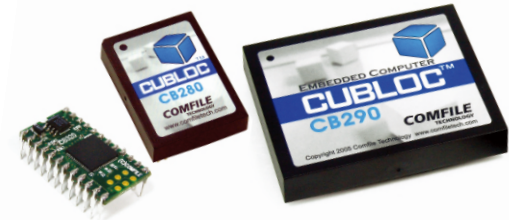


# CB210 이란?

CB210은 CUBLOC 모듈 중  
초보자가 입문하기 쉽도록 만든 제품이에요.



CUBLOC은 원래 자동제어 분야에 쓰일 수 있도록  
모듈 형태로 만든 제품인데,  
BASIC과 LADDER LOGIC을 사용할 수 있습니다.



그런데, **BASIC** 언어가 초보자가 배우기에 아주  
쉬운 언어예요.

```
PICBASIC Studio [ d:\source\Wmplate\Wmplate2.bas ]
파일(F)  편집(E)  보기(V)  실행(R)  설정(O)  도움말(H)
[Icons] [Run] [Stop] [Break] [Reset] [Help]

GOSUB I2C_WRITE      ' EEPROM에 라이팅 한다.
L = L + 1
DT = LPT.H
GOSUB I2C_WRITE      ' EEPROM에 라이팅 한다.
L = L + 1
NEXT L
LOCATE 0,0
PRINT "EEPWRITE ALL DONE "
OUT 0,1

20 GOTO 20          ' 대기

I2C_WRITE:          ' EEPROM 라이팅 서브루틴
L = 1
SHIFTOUT SCL, SDA, 2, L
J = L.H AND 6HOF
SHIFTOUT SCL, SDA, 2, J
J = L
SHIFTOUT SCL, SDA, 2, J
SHIFTOUT SCL, SDA, 2, DT
GOSUB I2C_STOP

' CONTROL WORD
' ADDRESS 1
' ADDRESS 2
```

그래서, 임베디드 콘트롤을 처음으로 입문하는  
사람들이, 임베디드 콘트롤이 무엇인지 그 맛을  
보기에 적당한 제품이지요.



CB210으로 임베디드 콘트롤이 무엇인지  
알아본 후에, 다른 콘트롤러를 공부해도 늦지 않아요.



처음부터 어려운 컨트롤러를 선택해서 힘들게 공부할 필요가 없잖아요.



그래!  
이걸로  
결정했어!

아주 예전에는 8051보드라는 것으로 처음 배우기 시작했는데, 보드 만드는데만 일주일 넘게 걸렸어요.



그 일주일동안 포기한 친구들도 많이 있었어요.



결국 보드를 다 만들었지만, 동작하지 않는 친구들도 많았답니다.



CB210을 사용하면, 이런 과정들이 필요없어요. 그냥 PC와 USB케이블로 연결하면 준비 끝이에요.



그리고, PC에 BASIC코드를 입력한 뒤, 다운로드하면 바로 동작해요.



사실, 임베디드 콘트롤에서 중요한 부분은 소프트웨어예요. 즉, 프로그램을 짜는 기술이죠. CB210을 사용하면 본질에 바로 접근할 수 있어서 좋아요.



프로그램을 잘 짜지만, 하드웨어를 잘 못 다루는 사람과 하드웨어를 잘 다루지만, 소프트웨어를 잘 모르는 사람 중 누가 더 현장에서 대우를 받을까요?



난 하드웨어 밖에 몰라~



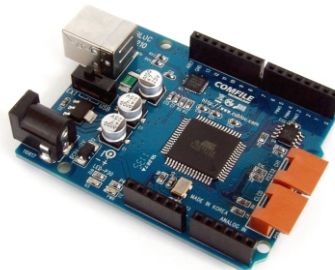
난 소프트웨어 밖에 몰라~

물론 하드웨어와 소프트웨어를 모두 잘 다룬다면 금상첨화겠지만, 그건 이제 걸음마를 배우는 아기한테 뛰라고 하는 것과 같아요.



이제 뛰어봐!

하드웨어는 실무에서 일을 하면서 천천히 배울 수 있지만, 프로그램을 짜는 스킬은 초기 시간 투자가 필요합니다. 그래서 이 책을 보실때에도 프로그램 짜는 방법과 그 동작원리에 집중 하시면 되요.



CB210은 전원만 넣으면 동작이 되도록 준비된 보드예요!



# 02

## 큐블록이란 무엇인가?

이 장에서는 큐블록이란 무엇이고, 왜 공부해야 하는가에 대한 궁금증을 해결해드릴 것입니다.

- CUBLOC 사용 하기







# CUBLOC을 사용하기 위해 필요한 것들

CUBLOC을 사용하기 위해 다음 준비가 필요합니다.

1



본 책에서는  
CUBLOC 코어 모듈 중  
CB210을 가지고 설명을 진행  
합니다.  
CB210은 입문자용 큐블록  
모듈입니다.

2



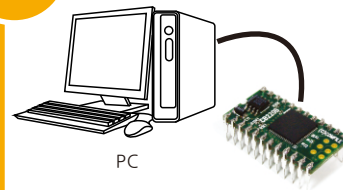
CUBLOC Studio를 최상위  
버전의 프로그램으로 설치합니다.  
홈페이지([www.comfile.co.kr](http://www.comfile.co.kr))  
로 접속하셔서,  
최상위 버전의 소프트웨어로  
설치 하시면 됩니다.

3



CB210을 사용하기 위해서는  
USB케이블이 필요합니다.

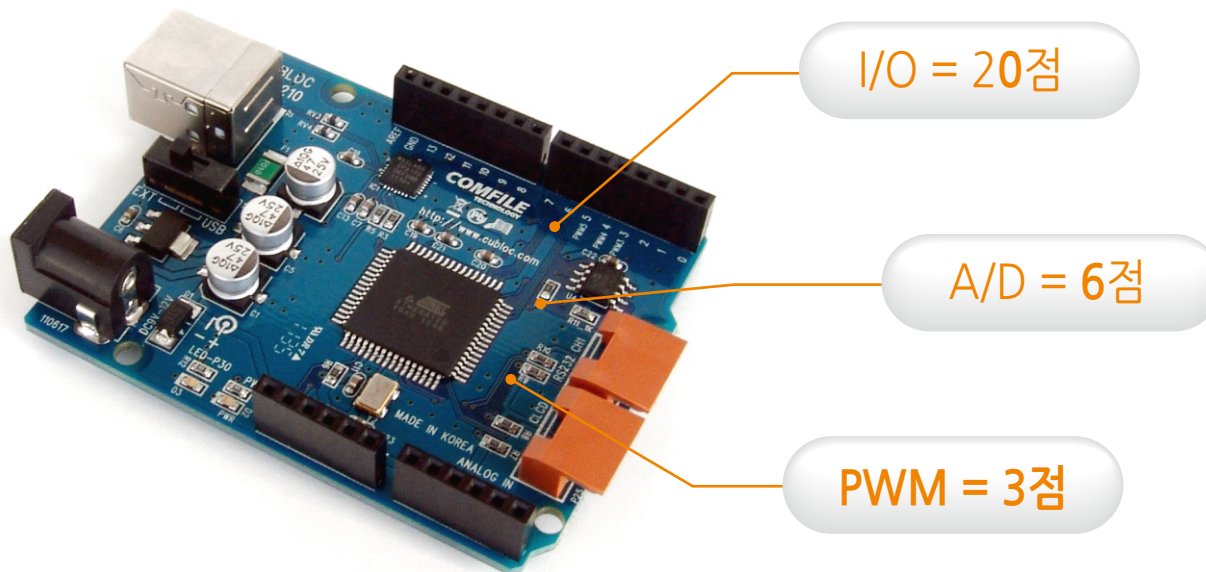
4



USB케이블로 CB210과  
PC를 연결합니다.

# CUBLOC - CB210

CB210을 살펴보겠습니다.



■ I/O란? 입력과 출력기능을 하는 것을 I/O라 합니다. INPUT, OUTPUT의 I와 O를 써서 I/O라고 표현합니다.

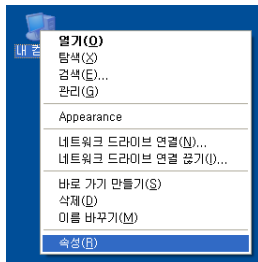
■ A/D란? Analog와 Digital의 합성어로 아날로그 신호를 디지털로 바꾸어 주는 기능을 합니다. 자세한 내용은 "AD사용법"에서 설명하도록 하겠습니다.

■ PWM이란? Pulse Width Modulation의 약자로 펄스 파형을 만들어 내는 기능을 합니다. 자세한 내용은 "PWM사용법"에서 설명하도록 하겠습니다.



# CUBLOC - CB210의 드라이버 설치

CB210 을 사용하기 위해서는 사전에 USB드라이버를 설치해주어야 합니다.

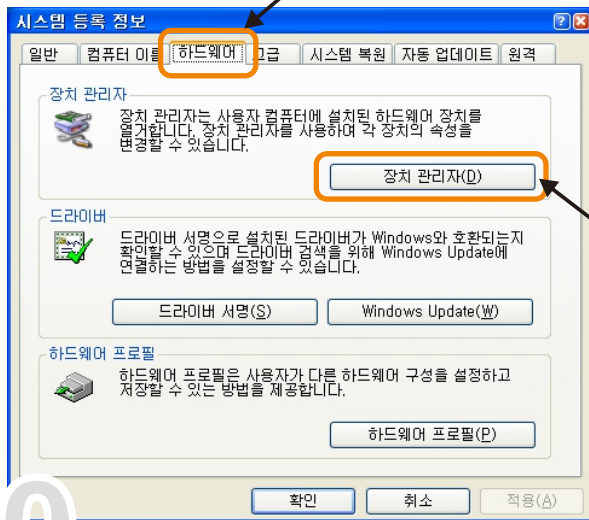


1

바탕화면의 내 컴퓨터에 커서를 위치 시킨 후 마우스의 오른쪽 버튼을 눌러서 "속성"을 클릭합니다.

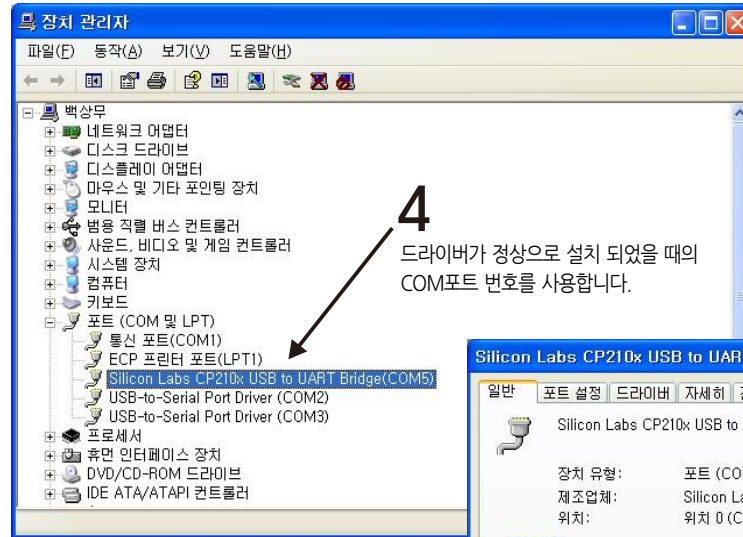
2

하드웨어 탭을 선택하신 후



3

장치 관리자를 누르세요!

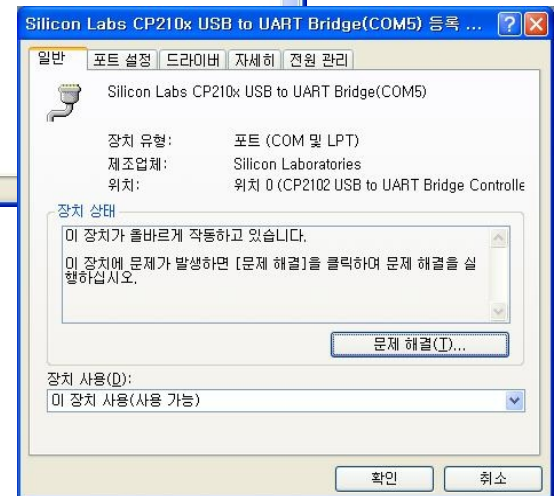


4

드라이버가 정상으로 설치 되었을 때의 COM포트 번호를 사용합니다.

5

장치가 올바르게 작동 중이라는 메시지가 나온다면.. 정상적으로 설치된 것입니다.



만약, "올바르게 작동 중"이라고 표시되지 않는다면, 주위의 컴퓨터 전문가에게 도움을 요청하거나, 관련 서적을 참조해서 이 문제를 해결하시기 바랍니다.

# CUBLOC Studio

CUBLOC Studio는 홈페이지(www.comfile.co.kr)에서 다운로드 가능합니다.

Home > 자료실 > 프로그램

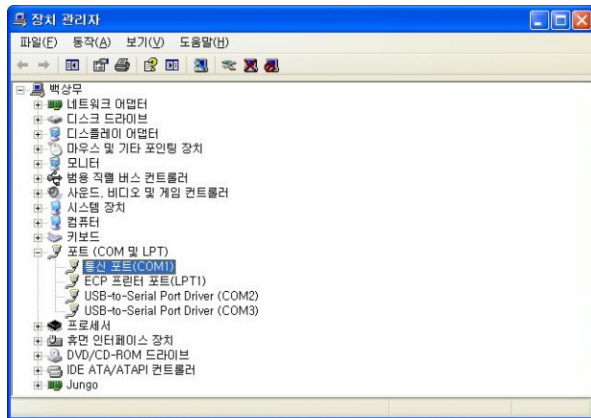
## Program Download

번호	제목	날짜	조회
17	SmartScreen에서 연석할 수 없는 앱이라고 뜬때 해결방법	2015-09-22	1813
16	ComfileHMI Editor 휴먼 대신 인터페이스	2016-08-29	12214
15	<b>CUBLOC Studio 큐블록 통합개발환경</b>	2015-09-22	22070
15	MOACON Studio 모아콘 통합개발환경	2015-03-21	5417
14	TinyPLC용 프로그램 모음	2014-05-01	15342
13	윈도우 CE 터치패널 PC 용 SDK 와 USB드라이버	2014-04-30	4935
12	디버그 터미널 (통신 테스트 프로그램)	2017-10-31	1788
11	CPCV5 드라이버 모음	2017-05-24	1050
10	윈도우 모바일 디바이스 센터 (윈도우 7,10사용자용)	2017-05-16	5553
9	JControls CF35 (CUWIN용 닷넷 콘트롤 라이브러리)	2016-09-05	928
8	CUBLOC 다운로드	2014-12-09	7980
7	M디스플레이 콤웨어	2014-07-03	2501
6	CB210, MOACON 용 USB 드라이버 (32비트, 64비트용)	2014-06-30	3544
5	윈도우 CE 터치패널 PC - OS 자료실	2014-05-01	4931
4	인텔리 LCD 콤웨어	2014-04-19	2056
3	통신 체크 유틸리티 CF-TERM	2014-04-19	9656
2	CUTOUCH 유틸리티 모음	2014-04-19	4359
1	USB-T0-RS232C 드라이버 (FTDI(강원전자) 또는 Prolific 칩셋 모델용)	2014-04-19	6715

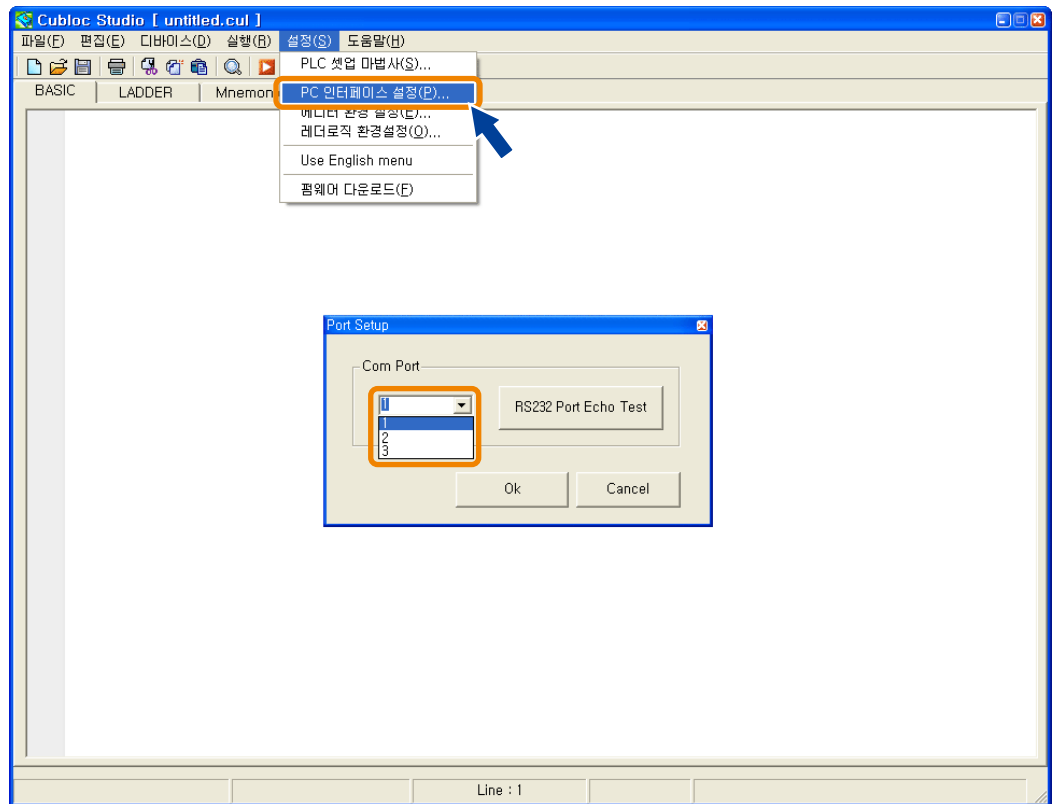
목록 글쓰기

# CUBLOC 소스 다운로드

앞서 USB드라이버가 설치된 COM포트를 CUBLOC STUDIO 상에서 "PC 인터페이스 설정" 으로 선택해야 합니다.  
COM 포트는 컴퓨터 등록정보의 시스템에서 확인 할 수 있습니다.



▲ 컴퓨터 등록정보의 시스템에서 확인

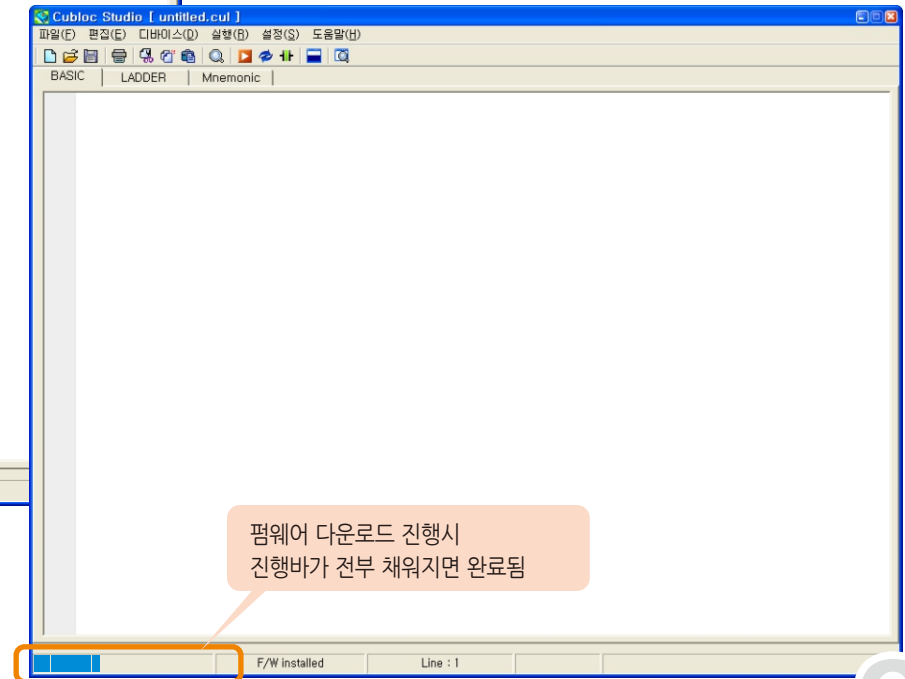
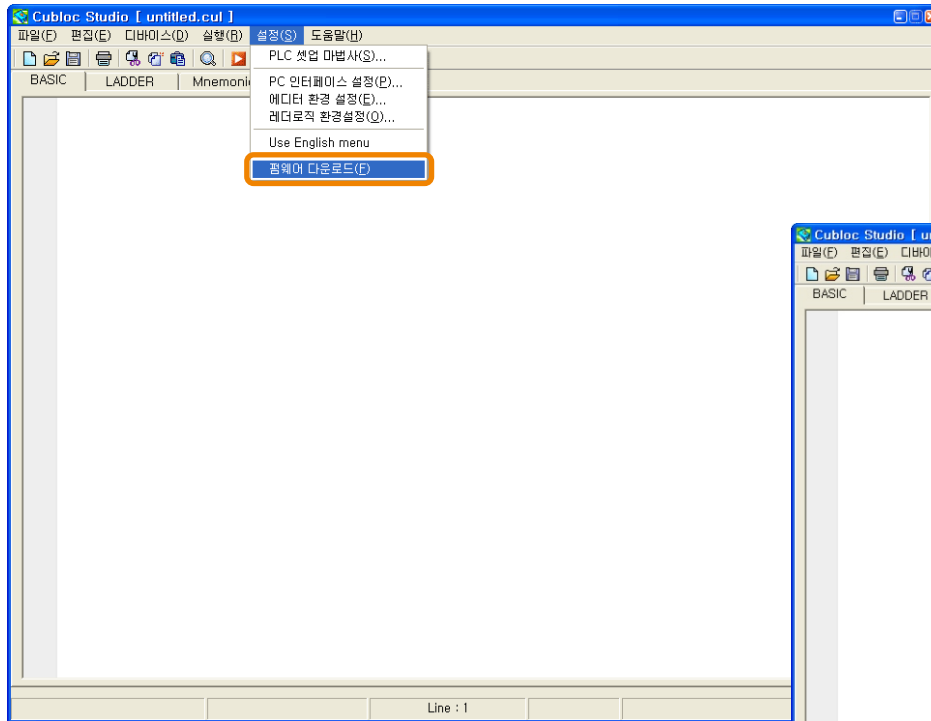


▲ CUBLOC Studio에서 COM포트 적용하기

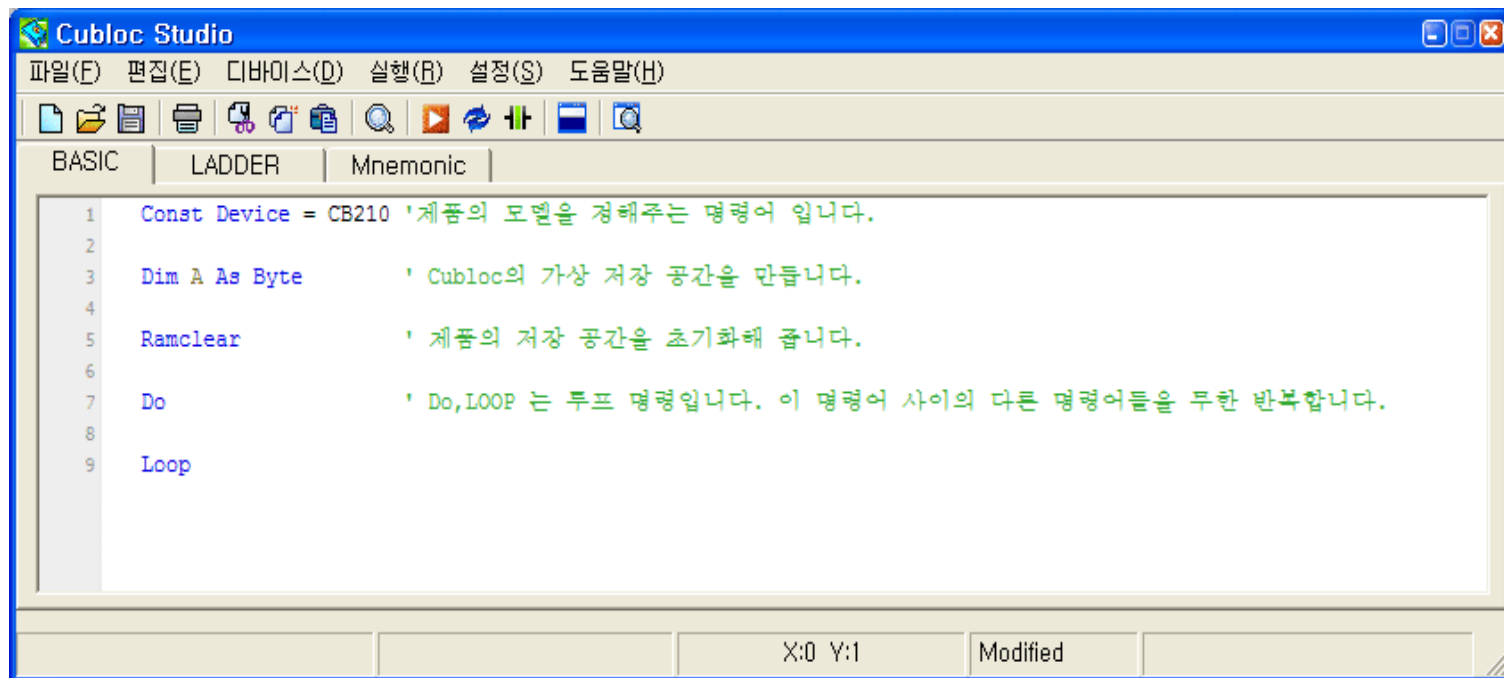
COM 포트를 설정하셨다면, 이젠 펌웨어를 다운로드 해야 합니다.

펌웨어는 매번 다운로드 할 필요 없이, CUBLOC STUDIO를 업데이트 및 새로 설치하셨을 경우에만 해주시면 됩니다.

공장출하시 펌웨어가 다운로드 되어 있긴 하지만, 유저가 현재 설치한 CUBLOC STUDIO 버전에 맞춰주기 위해 한번 더 다운로드하는 것입니다.

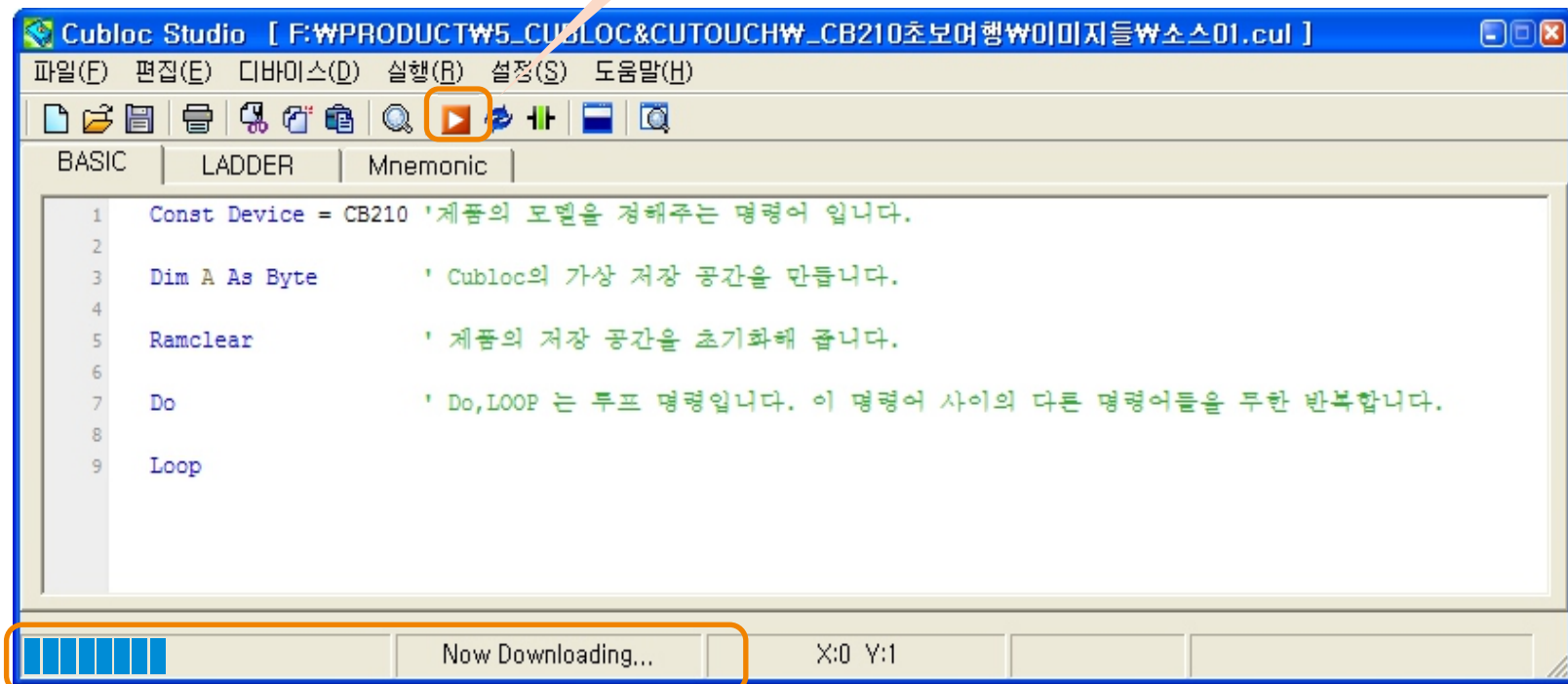


펌웨어 다운로드가 완료 되었다면 이제 본격적인 프로그램을 다운로드 하셔야 합니다.  
이제 소스 작성법을 알아보도록 하겠습니다.



위의 소스를 작성하였다면 다운로드를 합니다.

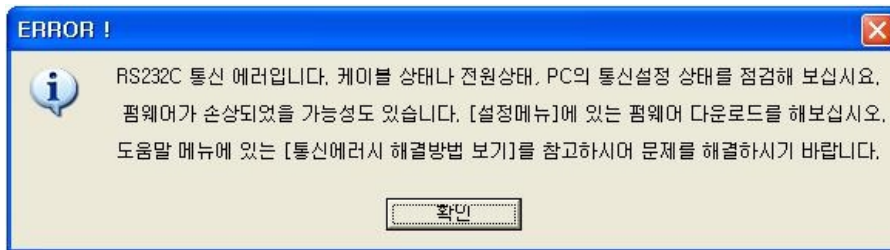
화살표 모양의 아이콘을 누르면 다운로드가 진행됩니다.



다운로드가 정상으로 진행될 경우  
위의 Now downloading... 메시지와 함께 진행그래프가 움직이게 됩니다.

# CUBLOC 소스 다운로드 에러

아래와 같은 오류가 발생 할 경우,  
COM포트를 확인해 주시고 펌웨어를 다시 다운로드 해보시기 바랍니다.  
또는 전원상태를 확인해보시기 바랍니다. 전원이 연결되어 있지 않을 수 있습니다.  
또는 USB케이블 접속상태를 확인해보시기 바랍니다. 코드가 빠져있을 수도 있습니다.



아래와 같은 오류는 드라이버가 잘못 설치 되었거나, COM포트 설정에 문제가 있는 경우입니다.  
다시한번 확인해 주시기 바랍니다.



# 03

## 베이직 프로그램

이 장에서는 큐블록 BASIC을 사용하는 기초적인 단계의 예제를 진행하면서, 큐블록 BASIC의 사용법에 대해서 공부하게 됩니다.

- LAB 1. CUBLOC으로 LED 켜기
- LAB 2. 변수의 사용
- LAB 3. 입력 포트를 이용한 LED 제어
- LAB 4. AD 값 입력 받기
- LAB 5. PWM을 이용한 LED 제어
- LAB 6. LCD를 이용한 데이터 출력
- LAB 7. 키패드 입력
- LAB 8. PC와의 RS232 통신



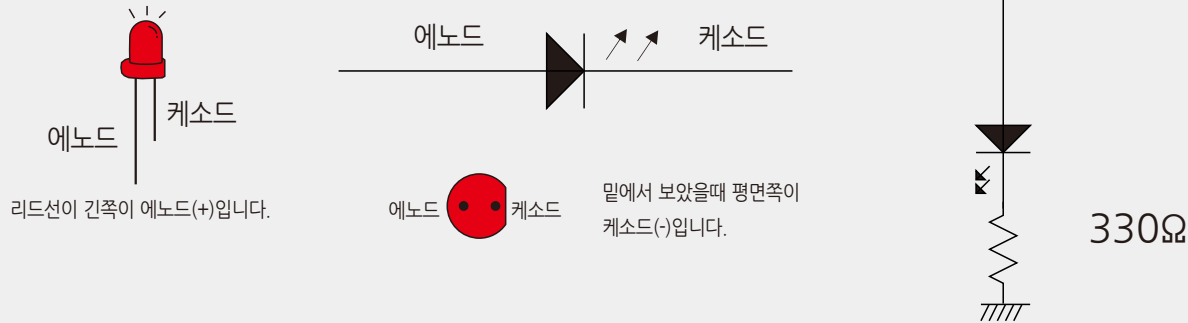


## LAB 1.

## CUBLOC으로 LED 켜기

이제 실제로 LED를 켜는 소스를 작성해 보도록 하겠습니다.

LED의 동작 원리는 +극(에노드)에 2V정도의 전압을 공급하고, -극(케소드)에 GND를 연결하면 동작합니다.

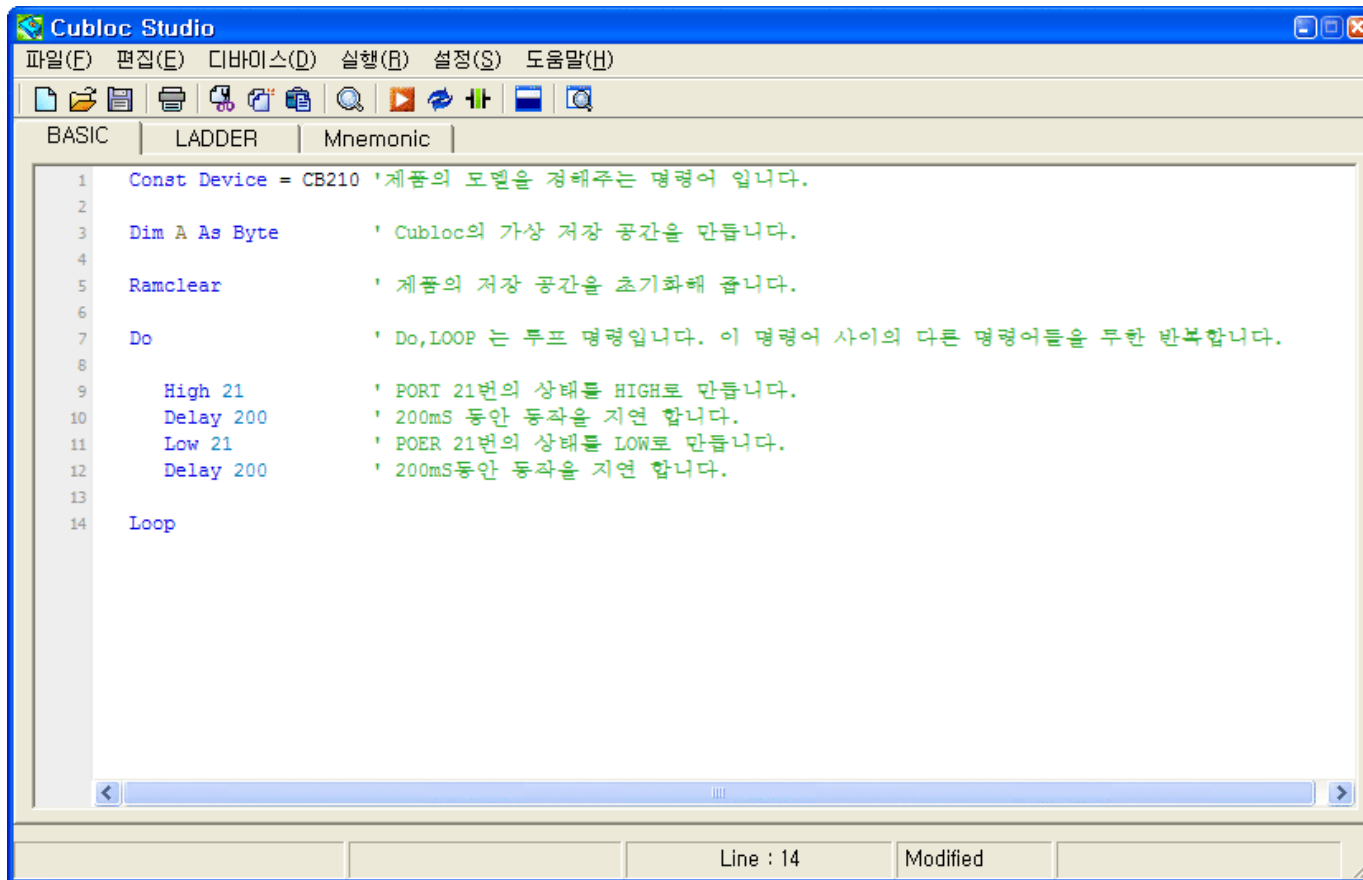


LED는 대략 2V의 전압에서 동작을 하게 됩니다. 하지만 CUBLOC의 출력 전압은 5V로 너무 많은 전압과 전류가 흐르게 됩니다.

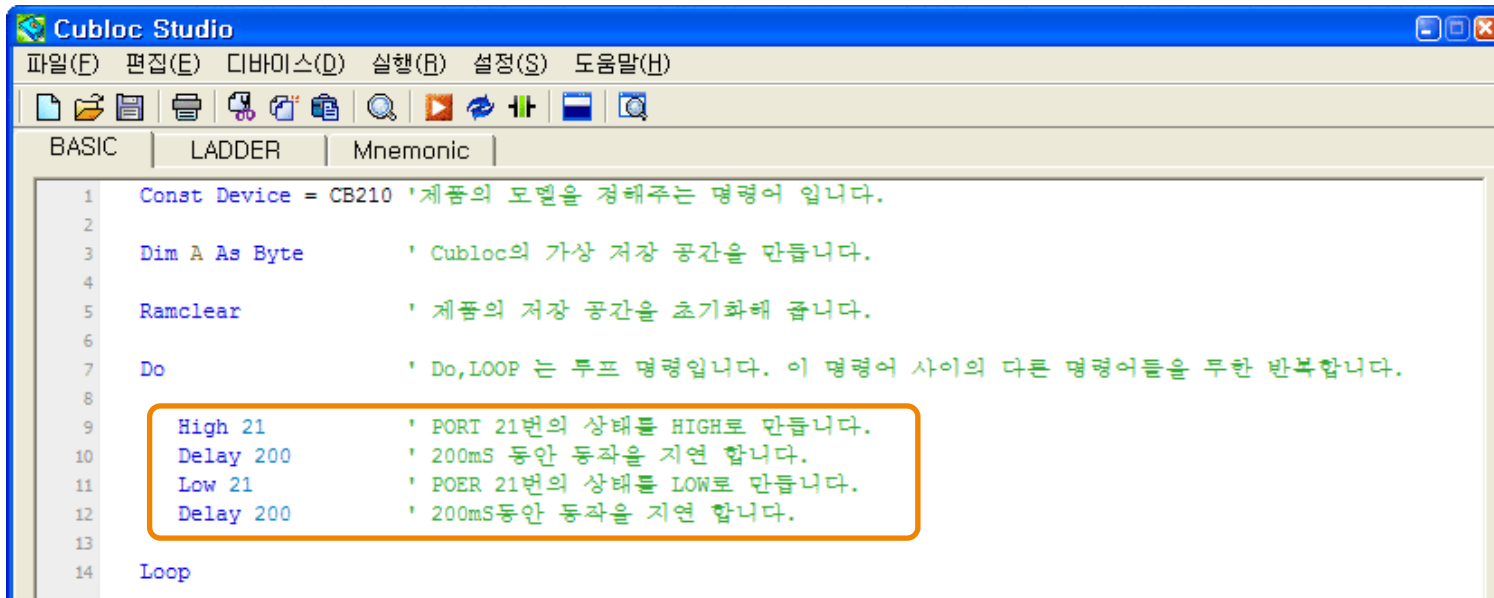
이렇게 되면 LED가 쉽게 파손되어 사용할 수 없게 되므로, 위의 회로와 같이 330Ω의 저항을 연결하여 전압과 전류를 제한하는 회로를 구성하여 사용하게 됩니다.

옴의 법칙을 이용하면 쉽게 전압과 전류 값을 구할 수 있습니다.

CUBLOC에 아래 소스를 다운로드 합니다.



## 소스 설명



### 새로운 명령어

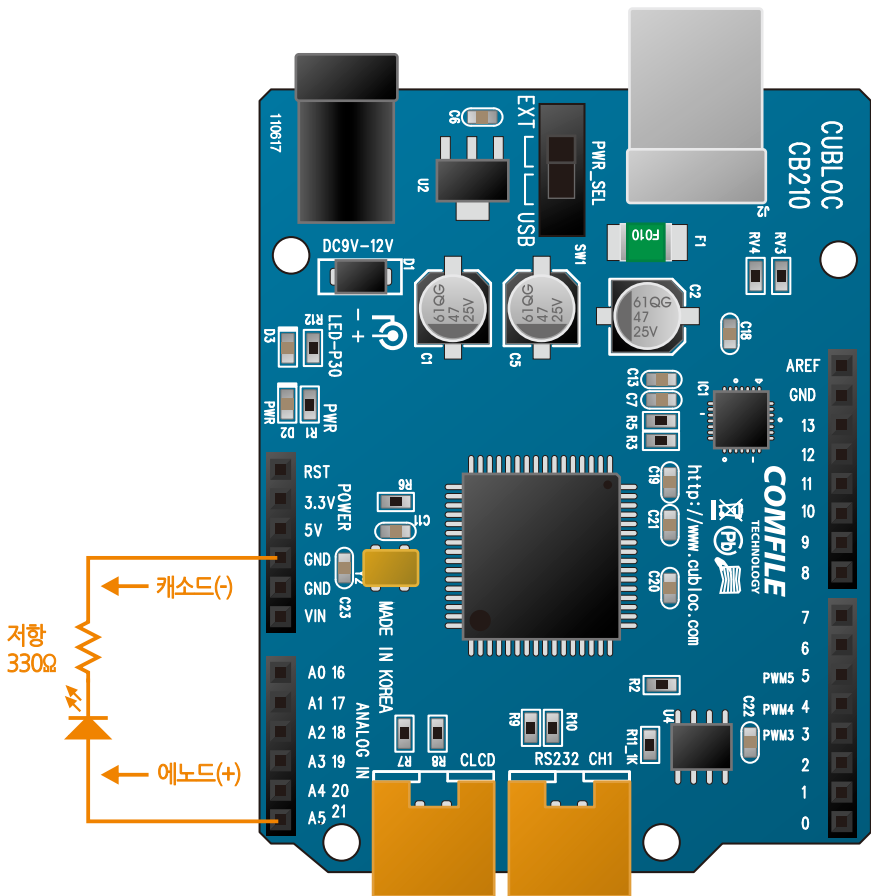
#### HIGH, LOW, DELAY

HIGH 0 : 0 포트를 켭니다. (포트의 상태를 1로 만듭니다.)

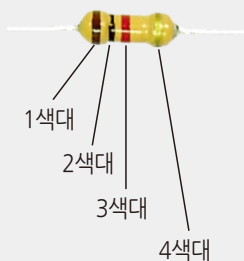
LOW 0 : 0 포트의 상태를 끕니다. (포트의 상태를 0으로 만듭니다.)

DELAY 200: 동작 시간을 200ms 지연 시킵니다.

# 회로



## 저항



	1색대	2색대	3색대	4색대
흑	0	0	10 <sup>0</sup>	
갈	1	1	10 <sup>1</sup>	
적	2	2	10 <sup>2</sup>	
노랑	3	3	10 <sup>3</sup>	
파랑	4	4	10 <sup>4</sup>	
초록	5	5	10 <sup>5</sup>	
보라	6	6	10 <sup>6</sup>	
주황	7	7	10 <sup>7</sup>	
빨	8	8	10 <sup>8</sup>	
분	9	9	10 <sup>9</sup>	
흰			0.1	± 5%
노			0.2	± 10%

저항의 값을 읽을 때는 색으로 그 값을 알 수 있습니다.

1색대와 2색대의 값은 위의 표대로 그냥 읽으면 되고,

3색대의 값을 10의 배승으로 읽어 계산하면 저항의 값을 읽을 수 있습니다.

4색대의 경우 저항 값의 오차 범위를 표시합니다.

예) 330Ω = 1색대 동색 = 3

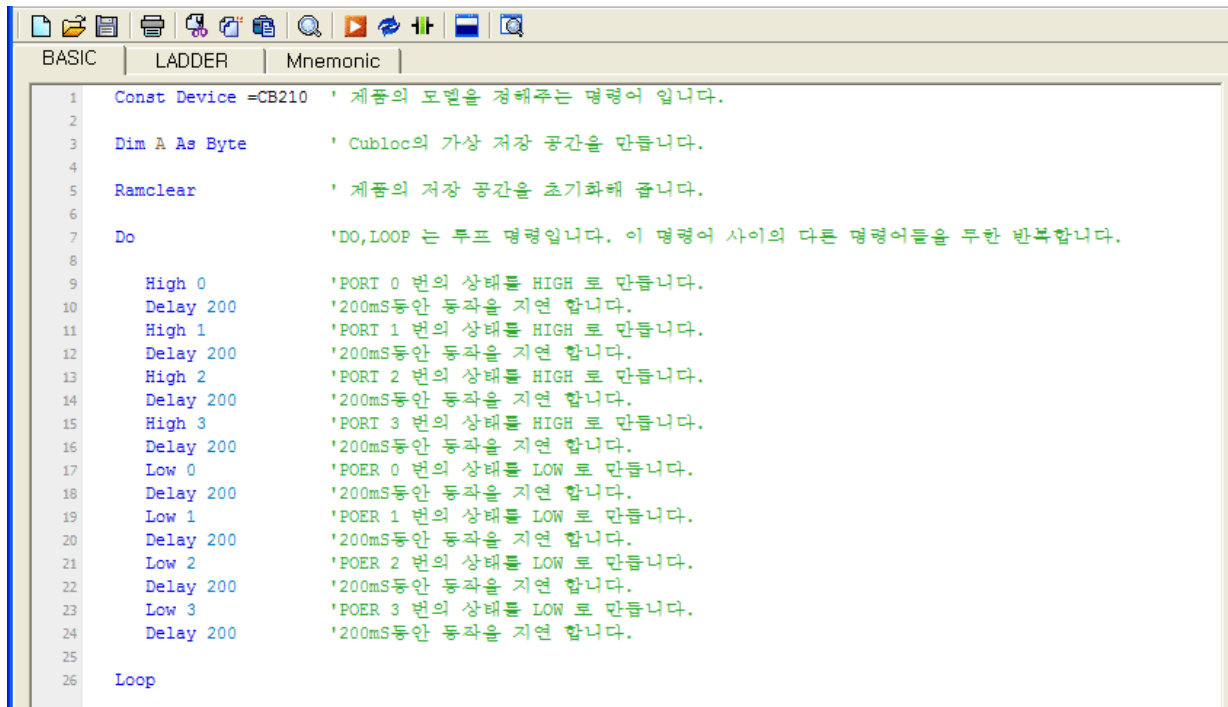
2색대 동색 = 3

3색대 갈색 = 10<sup>1</sup>

33 x 10<sup>1</sup> = 330

## CUBLOC으로 다중 LED 켜기

위의 소스와 연결 방법으로 LED를 200mS마다 깜빡이는 소스를 만들어 보았습니다.  
이 방법을 응용하여 LED 여러 개를 순차적으로 켜지는 제품을 만들어 보도록 하겠습니다.

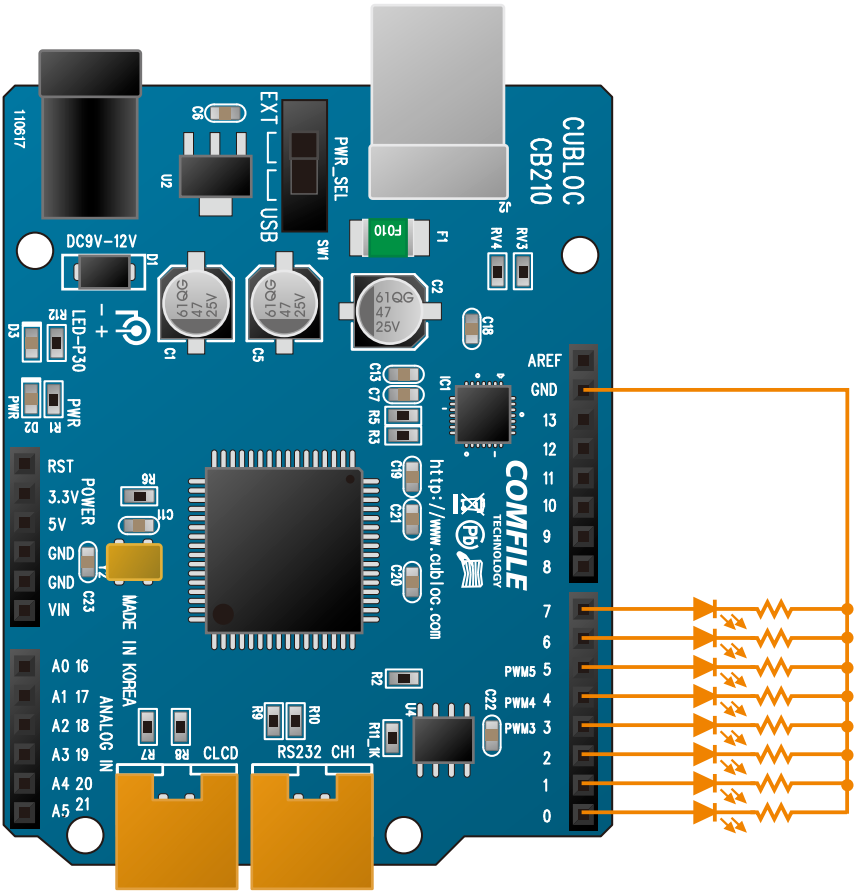


```

1  Const Device =CB210  ' 제품의 모델을 정해주는 명령어 입니다.
2
3  Dim A As Byte        ' Cubloc의 가상 저장 공간을 만듭니다.
4
5  Ramclear             ' 제품의 저장 공간을 초기화해 줍니다.
6
7  Do                  ' DO, LOOP 는 루프 명령어입니다. 이 명령어 사이의 다른 명령어들을 무한 반복합니다.
8
9      High 0           'PORT 0 번의 상태를 HIGH 로 만듭니다.
10     Delay 200        '200mS동안 동작을 지연 합니다.
11     High 1           'PORT 1 번의 상태를 HIGH 로 만듭니다.
12     Delay 200        '200mS동안 동작을 지연 합니다.
13     High 2           'PORT 2 번의 상태를 HIGH 로 만듭니다.
14     Delay 200        '200mS동안 동작을 지연 합니다.
15     High 3           'PORT 3 번의 상태를 HIGH 로 만듭니다.
16     Delay 200        '200mS동안 동작을 지연 합니다.
17     Low 0            'POER 0 번의 상태를 LOW 로 만듭니다.
18     Delay 200        '200mS동안 동작을 지연 합니다.
19     Low 1            'POER 1 번의 상태를 LOW 로 만듭니다.
20     Delay 200        '200mS동안 동작을 지연 합니다.
21     Low 2            'POER 2 번의 상태를 LOW 로 만듭니다.
22     Delay 200        '200mS동안 동작을 지연 합니다.
23     Low 3            'POER 3 번의 상태를 LOW 로 만듭니다.
24     Delay 200        '200mS동안 동작을 지연 합니다.
25
26 Loop
  
```

LED를 하나 켜는 것과 비슷한 내용입니다. 단지 ON/OFF 하고 싶은 포트들만 많아졌을 뿐 소스의 내용은 다르지 않습니다.

회로



## 소스 설명

위의 소스와 같이 동일 동작을 하는 여러 포트들을 나열함으로써 간단한 동작을 시킬 수 있습니다.

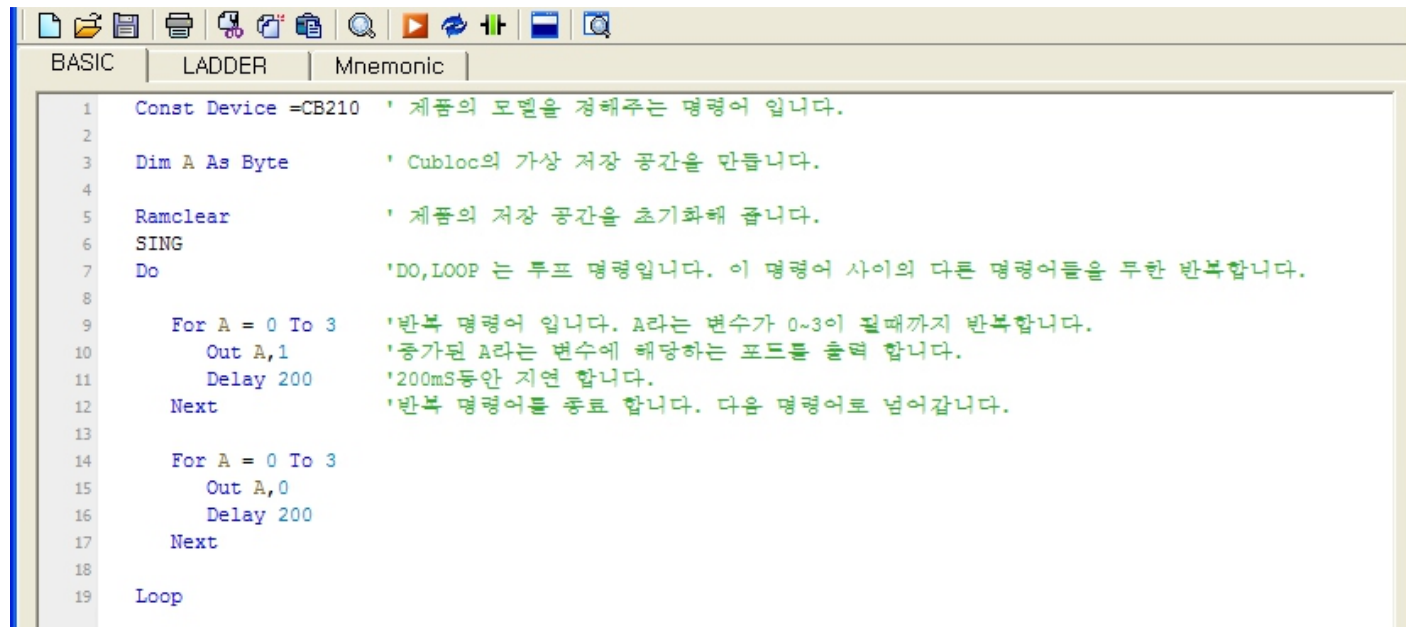
하지만 같은 동작을 반복 하는 소스일 뿐인데 소스의 길이가 엄청나게 늘어난 것을 볼 수 있습니다.

단지 LED를 같은 동작으로 3개만 늘렸을 뿐인데 소스의 길이는 2배 이상 늘어난 것입니다.

지금은 단순 동작이기 때문에 이렇게 길게 나열하여 소스를 작성해도 실수 한 부분이나 잘못된 동작을 찾아내기 쉬우나, 동작이 복잡해 질수록 이를 찾아내기 어려워 집니다.

프로그램은 어느 누가 보더라도 이해 할 수 있도록 짜는 것이 좋습니다. 그래야 차후 수정이나 응용이 쉬워지므로 나중에 다시 같은 프로그램을 작성 할 때 유용합니다.

그러면 위의 소스는 어떤 식으로 간략화 할 수 있을지 알아보도록 하겠습니다.



```

1  Const Device =CB210  ' 제품의 모델을 정해주는 명령어 입니다.
2
3  Dim A As Byte        ' Cubloc의 가상 저장 공간을 만듭니다.
4
5  Ramclear              ' 제품의 저장 공간을 초기화해 줍니다.
6  SING
7  Do                    ' DO, LOOP 는 루프 명령어입니다. 이 명령어 사이의 다른 명령어들을 무한 반복합니다.
8
9      For A = 0 To 3    '반복 명령어 입니다. A라는 변수가 0~3이 될때까지 반복합니다.
10         Out A,1        '증가된 A라는 변수에 해당하는 포트를 출력 합니다.
11         Delay 200      '200mS동안 지연 합니다.
12     Next              '반복 명령어를 종료 합니다. 다음 명령어로 넘어갑니다.
13
14     For A = 0 To 3
15         Out A,0
16         Delay 200
17     Next
18
19 Loop
  
```

```

Do                                ' Do,LOOP는 루프 명령어입니다. 이 명령어 사이의 다른 명령어들을 무한 반복합니다.

    For A = 0 To 3                ' 반복 명령어 입니다. A라는 변수가 0~3이 될 때까지 반복합니다.
        Out A,1                  ' 증가된 A라는 변수에 해당하는 포트를 출력합니다.
        Delay 200                ' 200ms동안 지연합니다.
    Next                          ' 반복 명령어를 종료합니다. 다음 명령어로 넘어갑니다.

    For A = 0 To 3
        Out A,0
        Delay 200
    Next

Loop

```

A라는 변수를 사용한 FOR 명령어를 분석해보면,  
A란 변수는 처음에 0으로 시작하여, 반복 될 때마다 1씩 증가하여 3이 되었을 때 종료 된 후, 다음 명령어가 실행 됩니다.  
그러므로 똑같은 동작을 하는 명령어를 0~3까지 4번을 반복하여 동작합니다.  
또한, 포트를 적어 넣는 부분에 변수를 넣음으로써 반복 될 때마다 포트번호도 증가되어 순차적으로 LED를 ON/OFF 할 수 있습니다.



## 새로운 명령어

### For A = 0 TO 3 ...NEXT

DO...LOOP는 무한반복 명령어라고 앞에서 설명한 바 있습니다.  
FOR 명령어는 위에 설정한 숫자만큼만 반복하는 명령어 입니다.  
먼저 소스의 위쪽에서 dim A as byte를 선언하여 변수를 만들었습니다.  
FOR 명령어는 지정된 A라는 변수를 0~3까지 증가 시키면서 반복합니다.  
Next 명령을 만나게 되면 다시 for의 처음으로 올라가 A를 증가시키고 명령어를 반복합니다.



## LAB 2.

## 변수를 이용한 LED 제어

CUBLOC에서 사용되는 변수는 4가지 입니다.

- **BYTE**: 8BIT의 구성을 가지고 있으며, 가장 작은 단위의 변수입니다.
- **INTEGER**: 16BIT의 구성을 가지고 있으며, BYTE 변수의 2배 입니다.
- **LONG**: 32BIT의 구성을 가지고 있으며, INTEGER의 2배 입니다.
- **SINGLE**: LONG형과 같이 32BIT이지만, 음수,실수의 표현이 가능합니다.

변수를 선언하는 방법은 아래와 같습니다.

```
Dim A As Byte      'BYTE 변수를 선언합니다.
Dim B As Integer    'INTEGER형 변수를 선언합니다.
Dim C As Long       'LONG형 변수를 선언합니다.
Dim D As Single     'SINGLE형 변수를 선언합니다.
```

변수명은 알파벳으로 시작하는 이름으로 작성합니다.

예약어를 변수명으로 사용할 수 없습니다.

예약어란 IF, THEN과 같이 BASIC에서 사용중인 단어를 말합니다.

## 2진수와 16진수

CUBLOC에서 주로 사용되는 값들은 대부분 2,10,16진수를 많이 사용하게 됩니다.  
이를 표를 통하여 알아 보도록 하겠습니다.

10진수	2진수	16진수	10진수	2진수	16진수
0	0000	0	8	1000	8
1	0001	1	9	1001	9
2	0010	2	10	1010	A
3	0011	3	11	1011	B
4	0100	4	12	1100	C
5	0101	5	13	1101	D
6	0110	6	14	1110	E
7	0111	7	15	1111	F

CUBLOC의 I/O포트들은 8개씩 묶여 있습니다. 위의 변수를 이용한 컨트롤을 쉽게하기 위함입니다.  
이를 BLOCK라 하는데 CB210에서는 3개의 BLOCK가 있습니다.  
0~7포트까지의 0 BLOCK, 8~13포트까지 1 BLOCK, 16~21포트까지 2 BLOCK로 3개의 BLOCK를 가지고 있습니다.  
이 BLOCK에 변수의 값을 넣어 컨트롤하게 되면 보다 쉽게 많은 I/O들을 동작 시킬 수 있습니다.

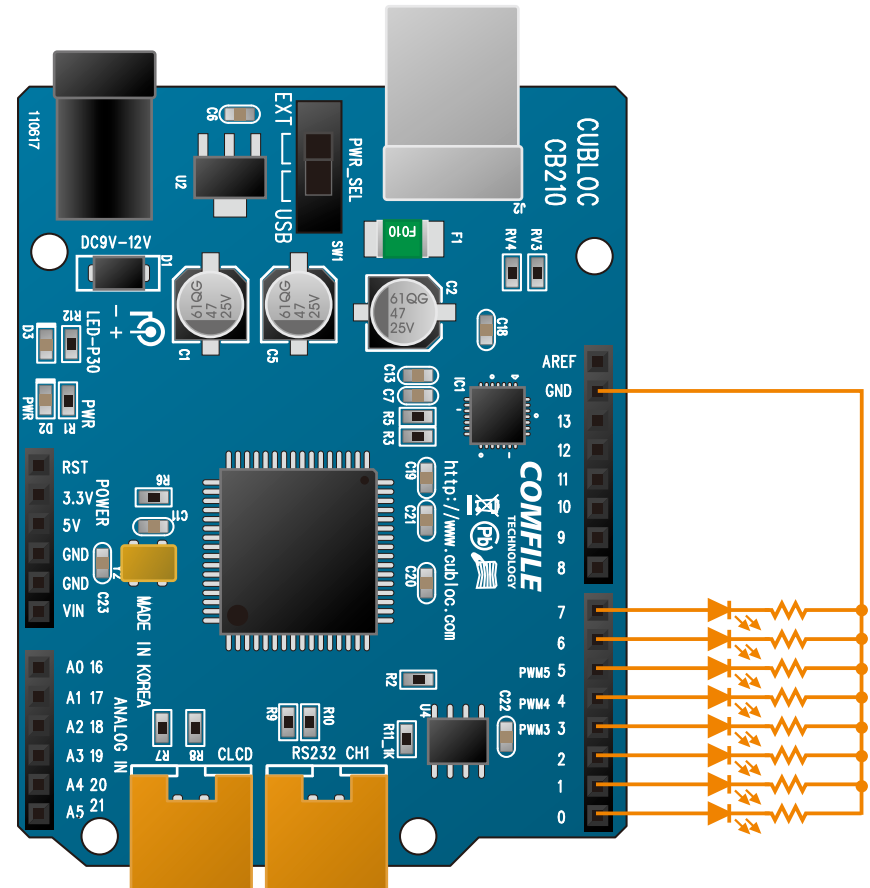
```
Do
  Dim A As Byte
  Ramclear

  Do
    Incr A      ' A변수의 값을 1씩 증가 시킨다.
    Byteout 0,A ' 8BIT의 I/O포트에 A변수를 넣는다.
    Delay 100   ' 100ms를 지연한다.

  Loop
```

이 소스의 LED는 어떻게 동작 할까요?

A의 변수에는 0~255까지의 값이 순차적으로 증가 됩니다.  
또한 BYTEOUT이라는 명령으로 0BLOCK에 A의 변수를 넣도록 하였습니다.  
그렇다면 0BLOCK에는 A의 값이 그대로 LED에 표현 될 것입니다.  
앞의 표에 2진수의 값이 그대로 LED에 표현 되는 것입니다.



## LAB 3.

## 입력을 받아 LED 켜기

LED를 ON/OFF 동작을 하는 방법을 알아 보았습니다.

이제는 이 LED를 원하는 입력을 받아 컨트롤 하는 방법을 알아보겠습니다.

CUBLOC의 I/O는 입출력이 별도로 정해지지 않아 선택적으로 입력 포트를 선택할 수 있습니다.

이 입력 포트를 이용하여 외부에서 들어오는 신호를 체크할 수 있습니다.

```
Const Device = CB210

Dim A As Byte

Ramclear

Do

    If In(8) = 1 Then
        Incr A           ' A변수의 값을 1씩 증가 시킨다.
        Byteout 0,A      ' 8BIT의 I/O포트에 A변수를 넣는다.
    Endif

    Delay 100
Loop
```

입력은 어떻게 받는지 알아보도록 하겠습니다.

- If ...END IF 명령은 조건을 지정해 주는 명령어 입니다.  
조건이 맞지 않으면 실행하지 않습니다.
- IN()명령은 뒤에 선택된 포트에 5V의 전압이 인가되면 상태 값이 1이 됩니다.



## 새로운 명령어

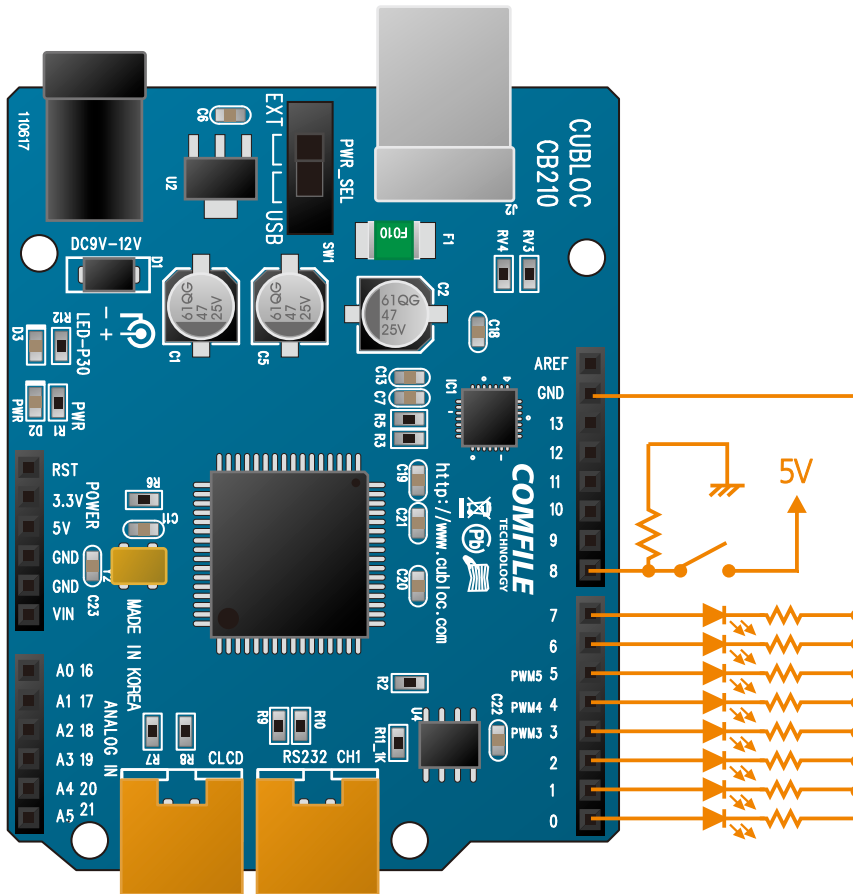
## IF IN(8) = 1 THEN

만약 8포트에 전압이 인가되면 아래의 명령을 실행하는 명령어입니다.

## IN( )

해당 포트를 감시하였다가,  
해당 포트에 전압이 인가되면 상태 값을 1로 만드는 명령어입니다.

# 회로



위의 회로와 소스를 이용하여 동작을 시키게 되면, 스위치를 눌렀을때 LED가 움직이는 것을 볼 수 있습니다.

## LAB 4.

## 아날로그 입력 받기



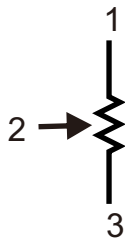
## 아날로그란 무엇인가?

- 아날로그는 디지털 신호 같은 0 또는 1이 아닌 그 사이의 값을 나타내는 것을 말한다.  
즉 0.1, 0.2와 같이 중간의 값을 수렴하는 것을 아날로그라고 말합니다.
- CUBLOC의 경우 0~5V의 아날로그 값을 가지고 있습니다. 0~5V 사이의 값을 0~1024의 디지털 수치로 변환되어 입력을 받게 되며,  
이 값을 이용하여 여러 데이터로 이용할 수 있습니다.
- 아날로그 신호는 센서를 사용 할 때 많이 사용됩니다. 현장의 온도나 거리 값 등은 단순 디지털 신호인 ON/OFF로는 알 수 없기 때문입니다.  
이 아날로그 값을 이용하여 어떤 동작을 하거나 현재의 상태를 표시 한다면, 좀 더 정밀한 제어를 할 수 있게 됩니다.

아날로그 입력을 받기 위해선 0~5V의 출력을 내보내 줄 센서가 필요합니다.

센서가 없을 경우, 가변 저항을 이용하여 전압 값을 만들어 줄 수도 있습니다.

## 가변 저항



## 가변 저항이란?

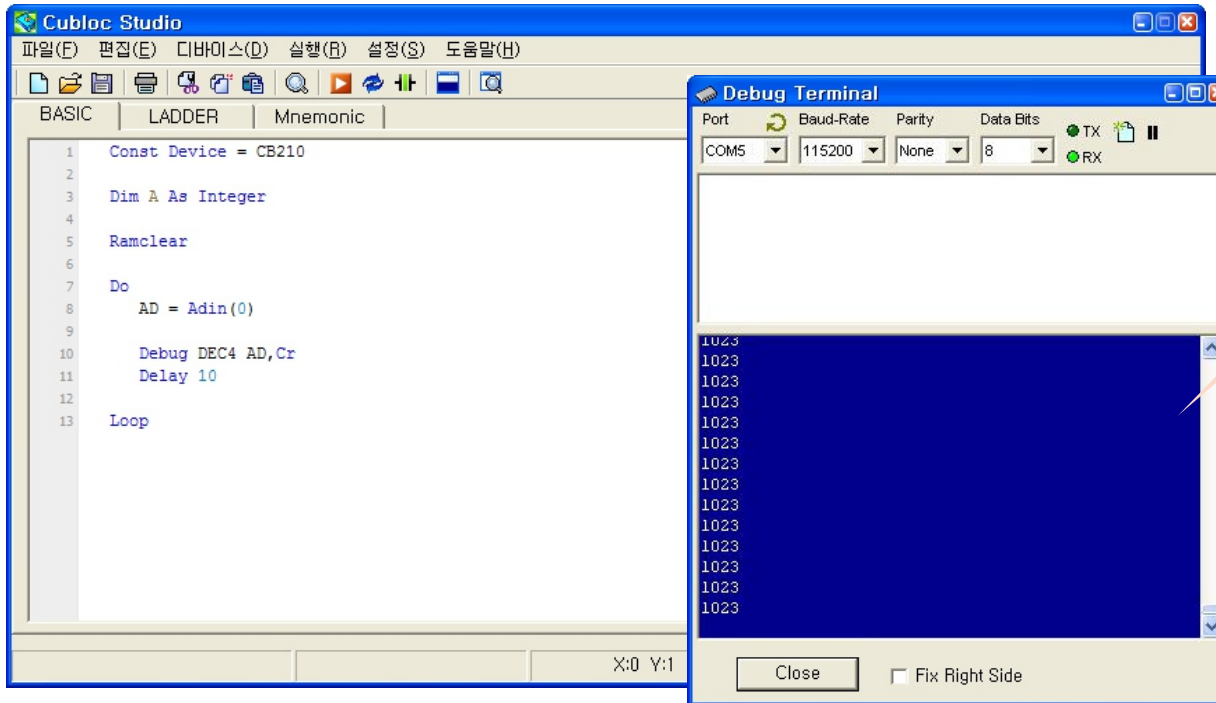
값이 고정되어 있는 일반 저항과 달리,  
필요에 따라 저항 값을 바꿔 줄 수 있는 저항을 가변 저항이라고 합니다.

가변 저항은 그림과 같이 손잡이가 있으며, 이 손잡이를 돌림으로써  
저항 값을 변경 할 수 있습니다.

CB210에는 아날로그 입력을 받을 수 있는 포트가 6개가 있습니다.

이 포트들은 일반적인 포트 번호가 아닌 AD채널로 구분됩니다. 일반 포트로는 사용되는 16번 포트는 AD채널 0으로 사용됩니다.

이 포트를 사용하여 AD 입력을 받아보도록 하겠습니다.



디버그 명령어를 이용한 값을 확인할 수 있는 창



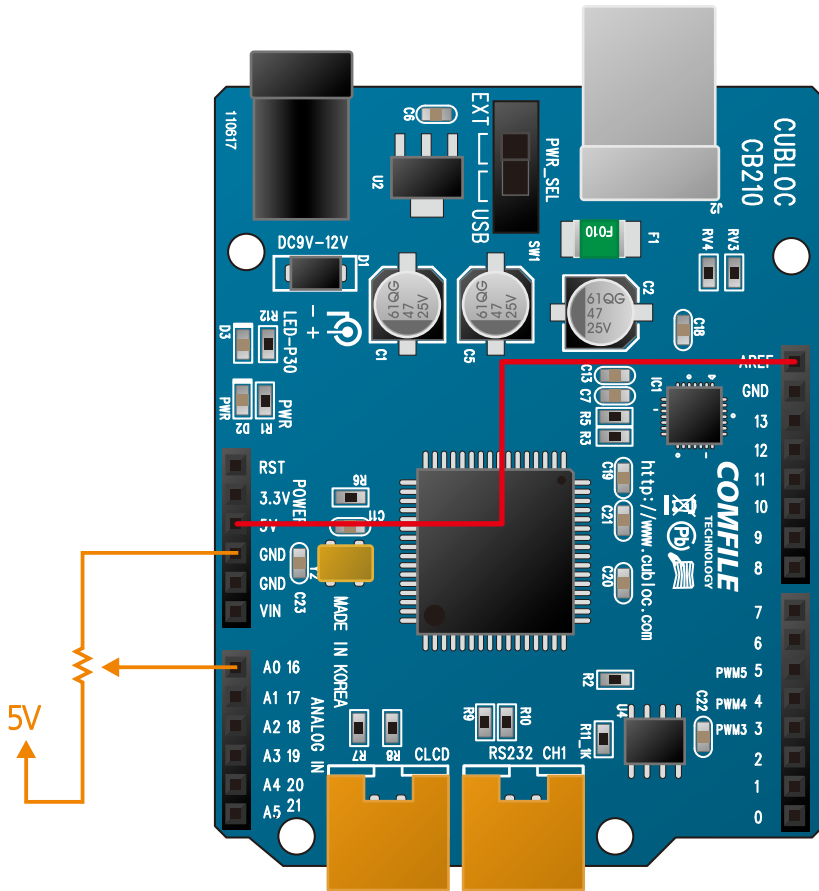
## 새로운 명령어

**ADIN( )** AD값을 받을 수 있는 명령어 입니다. AD값을 받아 AD변수에 0~1023의 상수로 저장합니다.

**DEBUG** CUBLOC의 변수에 저장된 값을 보여주는 명령어입니다.

10진수, 16진수 아스키코드 등 실제로 어떤 값이 들어있는지 확인하기 힘든 변수의 값들을 실제로 볼 수 있는 명령어입니다.

## 회로



AREF를 5V에 연결 해야 AD값을 정상적으로 받을 수 있습니다.

AREF는 AD의 기준을 잡아주는 포트입니다.

이 포트에 3.3V또는 5V를 넣게 되면, AD의 최대 치가 3.3~5V로 사용할 수 있습니다.

이는 분해능을 바꿔줄 수 있으므로 상황에 맞는 전압을 넣어주시면 됩니다.



## 아날로그 값을 이용한 LED 켜기

```

Const Device =CB210

Dim AD As Integer
Dim LED As Byte

Ramclear

Do
AD = Adin(0)

Debug DEC4 AD,Cr

If AD > 0 And AD < 200 Then
    LED = 1
Elseif AD > 200 And AD < 400 Then
    LED = 3
Elseif AD > 400 And AD < 600 Then
    LED = 7
Elseif AD > 600 And AD < 800 Then
    LED = 15
Elseif AD > 800 And AD < 1000 Then
    LED = 31
Endif

Byteout 0,LED

Delay 10

Loop

```

아날로그 값을 받아 확인 하는 방법을 알아 보았습니다.  
이젠 이 아날로그 값을 이용하여 LED를 제어해 보도록 하겠습니다.

AD 값을 이용하여 조건 문을 우선 만들어 줍니다.  
일정 조건이 되었을 때 LED변수에 값을 넣어,  
이 값을 BLOCK로 된 I/O에 적용하게 되면 해당 포트가 켜지게 됩니다.  
AD를 높일 수록 LED가 많이 켜지고,  
줄어들면 LED도 적게 켜지는 동작을 합니다.

## LED켜기 동작방법

동작을 자세히 확인해 보도록 하겠습니다.

가변저항을 돌리게 되면 AD값이 변화하게 됩니다. 이 값을 조건명령어인 IF 명령어를 통하여 조건을 만들어 줍니다.

AD값이 200~400까지의 값이 들어올 경우 LED 변수에 1의 값을 넣어 줍니다.

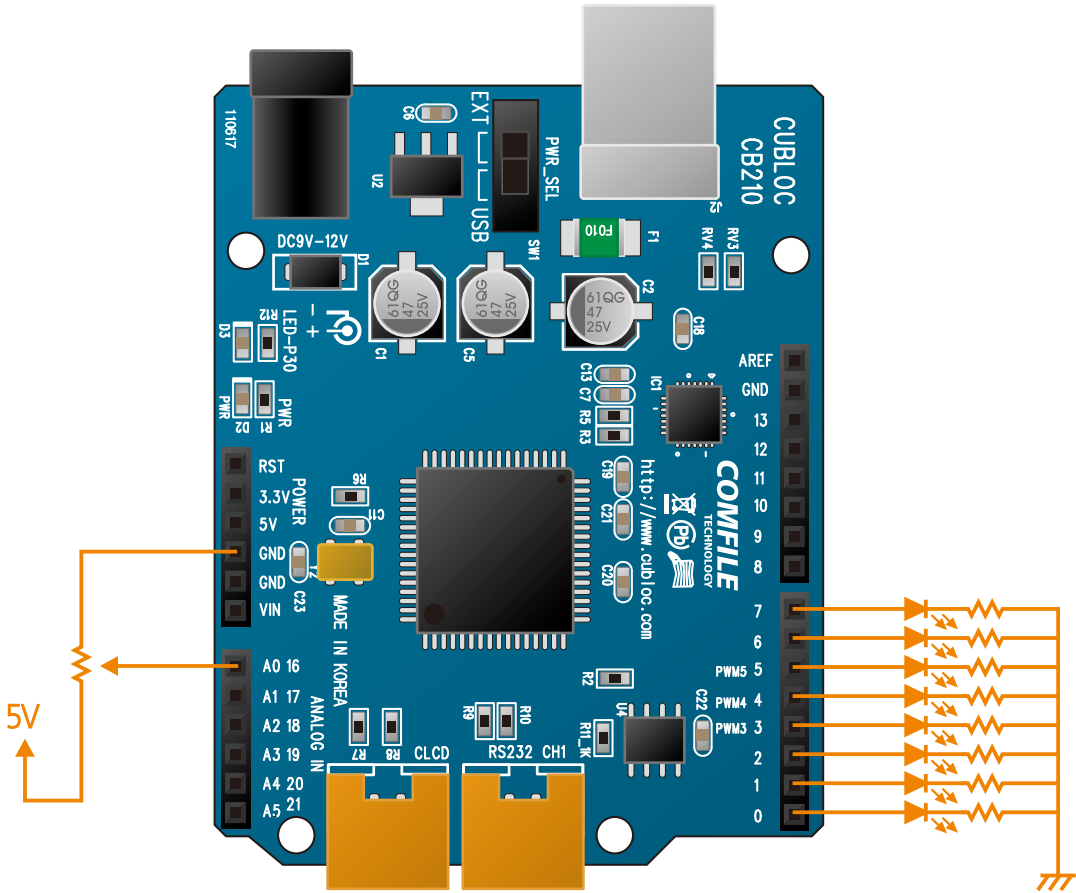
이어서 ELSEIF 명령어를 통하여 다른 조건을 만들어 줍니다.

조건에 따라 1,3,7,15,31의 값을 각각 넣어 주는 이유는 아래의 표와 같이 10진수를 2진수로 변경하였을 경우

LED의 순서를 순차적으로 켜기 위함입니다.

10진수	2진수							
1	0	0	0	0	0	0	0	1
3	0	0	0	0	0	0	1	1
7	0	0	0	0	0	1	1	1
15	0	0	0	0	1	1	1	1
31	0	0	0	1	1	1	1	1

회로



## LAB 5.

## PWM을 이용한 LED 제어



## PWM이란?

- Pulse Width Modulation의 약자로 펄스를 만들어 내는 기능을 합니다.
- 펄스 파형을 이용하여 모터나 전구를 제어 할 수 있습니다. 펄스는 주파수 출력이라고 할 수 있습니다.  
주파수는 구형 파형을 1초에 몇 번을 ON/OFF 하는가 입니다. 이를 이용하여 LED를 제어 할 경우 빠른 속도로 ON/OFF 되기 때문에 밝기에 영향을 끼치게 됩니다.  
사람의 눈으로는 확인 할 수 없을 정도의 빠르기이기 때문에 깜빡인다는 인식을 하지 못하며 이를 밝기로 인식하는 것입니다.



밝기가 희미



밝기가 밝음

```
Const Device =CB210

Dim AD As Integer

Ramclear

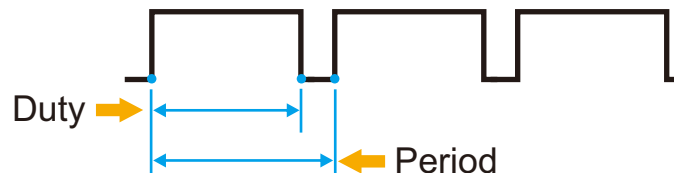
Do
  AD = Adin(0)
  PWM ,AD,1030

Loop
```

## PWM 명령어의 인수 설명

### PWM 채널 번호, Duty, Period

- 채널번호: PWM을 출력할 포트의 채널을 적어 줍니다.
- Duty: 주파수의 폭을 정해 줍니다.
- Period: 주파수의 주기를 정해 줍니다.



## 동작 방법

AD 값과 PWM을 동시에 이용하여 LED를 제어해 보도록 하겠습니다.  
AD의 볼륨을 돌리면 LED의 밝기가 달라지게 됩니다.

```
Const Device =CB210

Dim AD As Integer

Ramclear

Do
  AD = Adin(0)
  PWM 3,AD,1030

Loop
```

소스를 보면 AD값을 Duty에 적용한 것을 볼 수 있습니다.

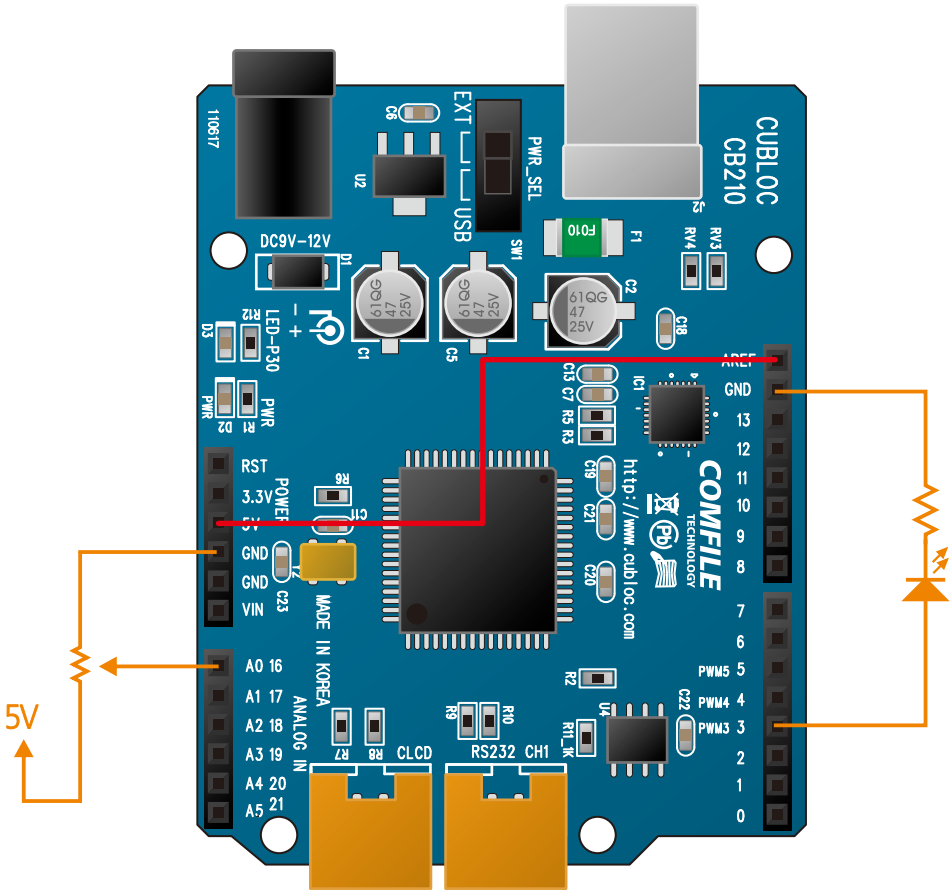
이는 주파수 주기의 길이로, 이 길이는 전압을 ON해주는 시간을 말합니다. 고로 전압이 ON되어있는 시간이 길어 질수록 LED는 밝아 집니다.

이와 같이 인수는 변수로 대체가 가능합니다.

이 변수는 상수로 이루어져 있어야 하며, 해당 인수의 범위 내 값이어야 합니다.

이 AD의 값이 뒤의 인수인 Period보다 높게 되면 펄스가 발생하지 않으며, 단순히 ON 되어있는 동작을 하게 됩니다.

회로



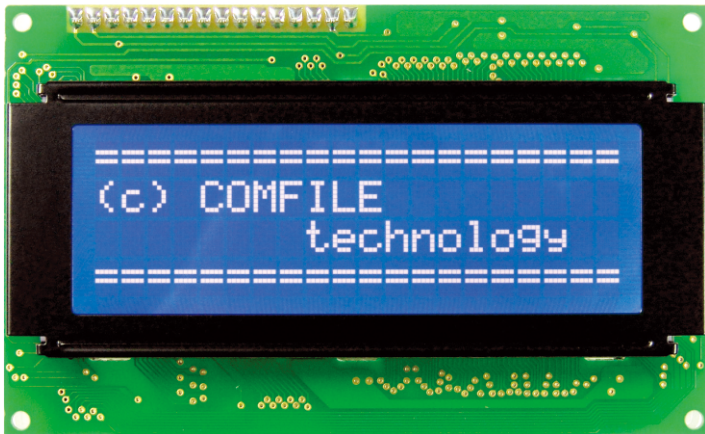


## LAB 6.

## LCD를 이용한 데이터 출력

앞에서 디버그 명령을 이용하여 변수의 값이나 출력하고 싶은 문자를 출력하는 방법을 배웠습니다. 하지만 현장에서 PC를 연결할 수 있는 상태가 아니거나 제품으로써 표시 기능을 보고 싶을 경우, 이를 출력해줄 디스플레이 장치가 필요합니다. 이를 쉽게 표시 할 수 있는 장치가 LCD 장치입니다. LCD에도 많은 종류가 있지만 간단한 문자만 표현 할 수 있는 캐릭터 LCD가 대표적으로 사용됩니다.

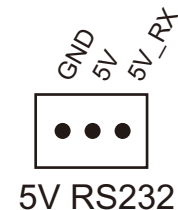
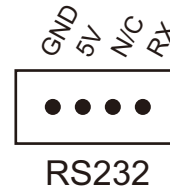
CUBLOC 모델인 CB210와 캐릭터 LCD 모델인 CLCD420-B 모델을 이용하여 LCD에 데이터를 출력 해보도록 하겠습니다.



LCD의 사양을 자세히 알아 보겠습니다.

## [CLCD420-B]

- 전원 5V: LCD가 동작되는 메인 전원.
- 가로 20자, 세로 4줄: LCD에서 출력 가능한 캐릭터량. 총 80자의 캐릭터 출력 가능.
- CUNET통신 가능: 컴파일 제품의 LCD 및 SGN을 구동할 때 사용되는 컴파일 전용 통신 방식.
- RS232통신 가능: 일반 PC와 통신 가능한 시리얼 통신.
- 5V RS232통신 가능: 일반 CPU(MCU)들과 통신 가능한 시리얼 통신.



```









1  Const Device = CB210
2
3  Dim ad As Integer
4
5  Set Display 2,0,0,50 'LCD를 설정을 합니다.
6
7  Ramclear
8
9  Cls                'LCD를 초기화 합니다.
10
11 Csrloff            'LCD 커서를 OFF합니다.
12
13 Delay 20           'LCD 초기화 후 최소 딜레이 시간
14
15 CLCDOUT 0,0,"LCD AD TEST!!"    ' LCD에 글씨를 출력 합니다.
16
17 Do
18
19     ad = Adin(0)      'LCD에 표시할 AD 값을 입력 받습니다.
20     CLCDOUT 0,1,"AD 1 = ",DEC4 ad 'LCD에 AD 값을 표시 합니다.
21     Delay 10         'LCD에 출력될 시간 지연을 줍니다.
22 Loop
```

### SET DISPLAY의 상수 설명

SET DISPLAY LCD 종류, 통신방식, 어드레스 및 통신 속도, 버퍼 크기

- LCD 종류: 0 = ALCD, 1 = GHBLCD, 2 = CLCD
- 통신 방식: 0 = CUNET, 1 = RS232
- 어드레스 및 통신 속도: LCD 컨트롤러 뒤쪽의 DIP 스위치 설정 상태
- 버퍼 크기: 가장 저장 공간으로 데이터가 한번에 처리 할 수 있는 데이터량을 버퍼라 합니다.

LCD의 뒷면에 보면 DIP스위치가 있습니다. 아래 표를 보고 I2C 슬레이브 어드레스를 0으로 맞추십시오.

DIP 스위치 상태	RS232 보레이트	I2C의 SLAVE ADDRESS
	2400	0
	4800	1
	9600	2
	19200	3
	28800	4
	38400	5
	57600	6
	115200	7

```

1  Const Device = CB210
2
3  Dim ad As Integer
4
5  Set Display 2,0,0,50 'LCD를 설정을 합니다.
6
7  Ramclear
8
9  Cls                'LCD를 초기화 합니다.
10
11  Csroff             'LCD 커서를 OFF합니다.
12
13  Delay 20           'LCD 초기화 후 최소 딜레이 시간
14
15  CLCDOUT 0,0,"LCD AD TEST!!" ' LCD에 글씨를 출력 합니다.
16
17  Do
18
19      ad = Adin(0)    'LCD에 표시할 AD 값을 입력 받습니다.
20      CLCDOUT 0,1,"AD 1 = ",DEC4 ad 'LCD에 AD 값을 표시 합니다.
21      Delay 10        'LCD에 출력될 시간 지연을 줍니다.
22  Loop

```

## CLCDOUT의 상수 설명 CLCDOUT X축, Y축, "데이터"

- X축: LCD의 가로축의 위치를 정합니다.
- Y축: LCD의 세로축의 위치를 정합니다.
- "데이터": LCD에 출력될 데이터나 변수를 넣어 줍니다.

변수 앞에 붙은 DEC4는 뒤에 나올 변수의 자리수를 정해주는 명령어 입니다.

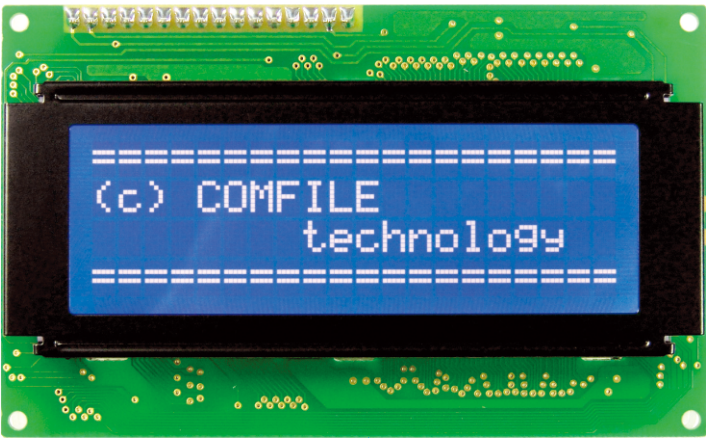
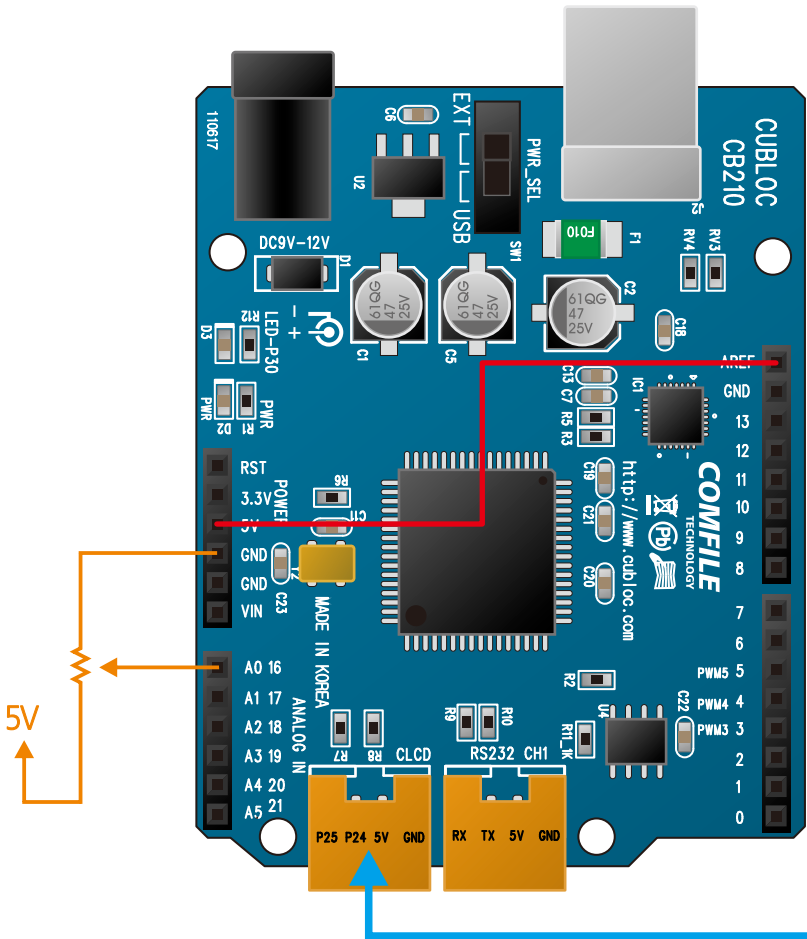
이 명령어를 생략 할 경우,

숫자의 변화에 대해서 좌측 정렬이 되지 않기 때문에, 자릿수가 변경 될 때 뒤에 남은 자릿수가 남아 표시된 값이 헛갈릴 수 있습니다.

## 출력 동작

- LCD를 출력 하기 위해서는 LCD 설정을 해주어야 합니다. SET DISPLAY명령을 이용하여 LCD의 종류와 통신 방법 등을 설정해 줄 수 있습니다.
- 처음 켜졌을 때 LCD의 초기화를 위해 CLS명령을 사용하여 초기화를 한 후 커서를 OFF합니다.  
커서를 OFF하는 이유는 값을 표시하는 부분에 커서가 남게 되면 값의 갱신에 따라 커서가 따라다니기 때문에, 빠른 값을 갱신하는 데이터에서는 OFF해주시는 것이 좋습니다.
- LCD의 첫 줄 첫번째 행에 LCD AD TEST라는 문구를 넣어 줍니다.
- AD입력을 받아 두번째 줄 첫번째 행에 AD 1 = AD데이터를 4자리까지만 출력해 줍니다.  
\*AD의 변수를 INTEGER로 선언 하였기 때문에 최대 값이 65535까지 표현이 가능합니다.  
단, AD의 최대 값이 1023이기 때문에 BYTE의 255보단 크고, INTEGER인 65535보단 적기 때문에 INTEGER로 선언 하였습니다.

회로



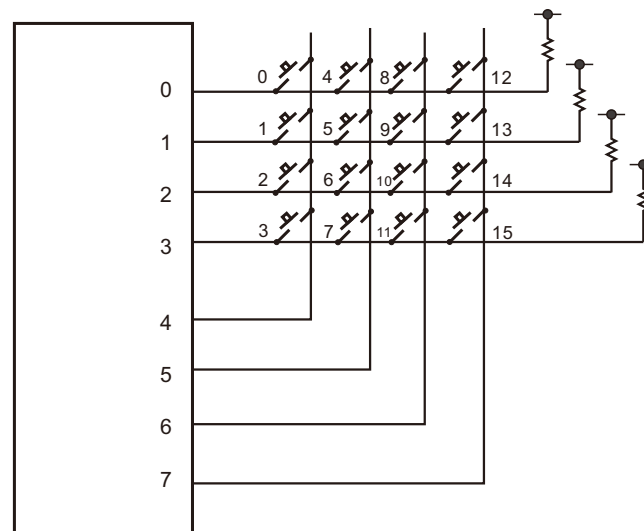
## LAB 7.

## 키패드 입력

입력은 스위치 같은 ON/OFF 외에도 키패드 입력을 받을 수 있습니다.

키패드란 적은 I/O를 이용하여 많은 양의 입력을 받을 수 있도록 되어 있습니다.

키패드를 입력 받는 방식을 키 매트리스 방식의 입력이라고 하는데, 키패드 외에도 적은 입력포트로 많은 양의 입력을 받을 때 사용합니다.



```

1  Const Device = CB210
2
3  Dim KEY_PAD As Integer
4  Dim KEY_DATA As Byte
5
6  Set Display 0,0,0,50
7
8  Const Byte KEY_TABLE = (1,4,7,11,2,5,8,0,3,6,9,12,13,14,15,16) '키매트릭스 테이블
9
10 Ramclear
11
12 Cls
13 Delay 20
14 Csriff
15 CLCDOUT 0,0,"KEY DATA = "
16 Do
17
18 KEY_PAD = Keypad(0)          '키매트릭스 값을 입력 받습니다.
19 KEY_DATA = KEY_TABLE(KEY_PAD) '키 매트릭스로 받은 값을 사용하기 쉽도록 테이블을 참조한다.
20
21
22 If KEY_PAD < 255 Then CLCDOUT 0,0,"KEY DATA = ",Dec1 KEY_DATA '키값이 입력되면 LCD에 값을 표시한다.
23
24
25 Loop

```

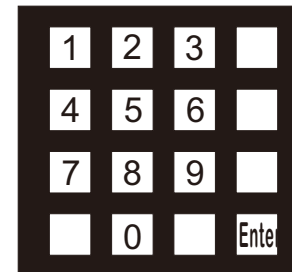
## KEYPAD(0)

키 매트릭스 값을 OBLOCK에 받아 변수에 저장하는 명령어입니다.

## CONST BYTE KEY\_TABLE = {}

KEY\_TABLE라는 테이블 변수를 만듭니다. 테이블 변수는 일정하지 않은 값을 가지는 데이터를 실제로 사용하기 쉬운 값으로 변경해줄 수 있는 변수입니다.

실제 들어는 스캔 코드 값과 현재 사용하고 싶은 값이 다르기 때문에, 위와 같이 배열 변수를 이용하는 것입니다.



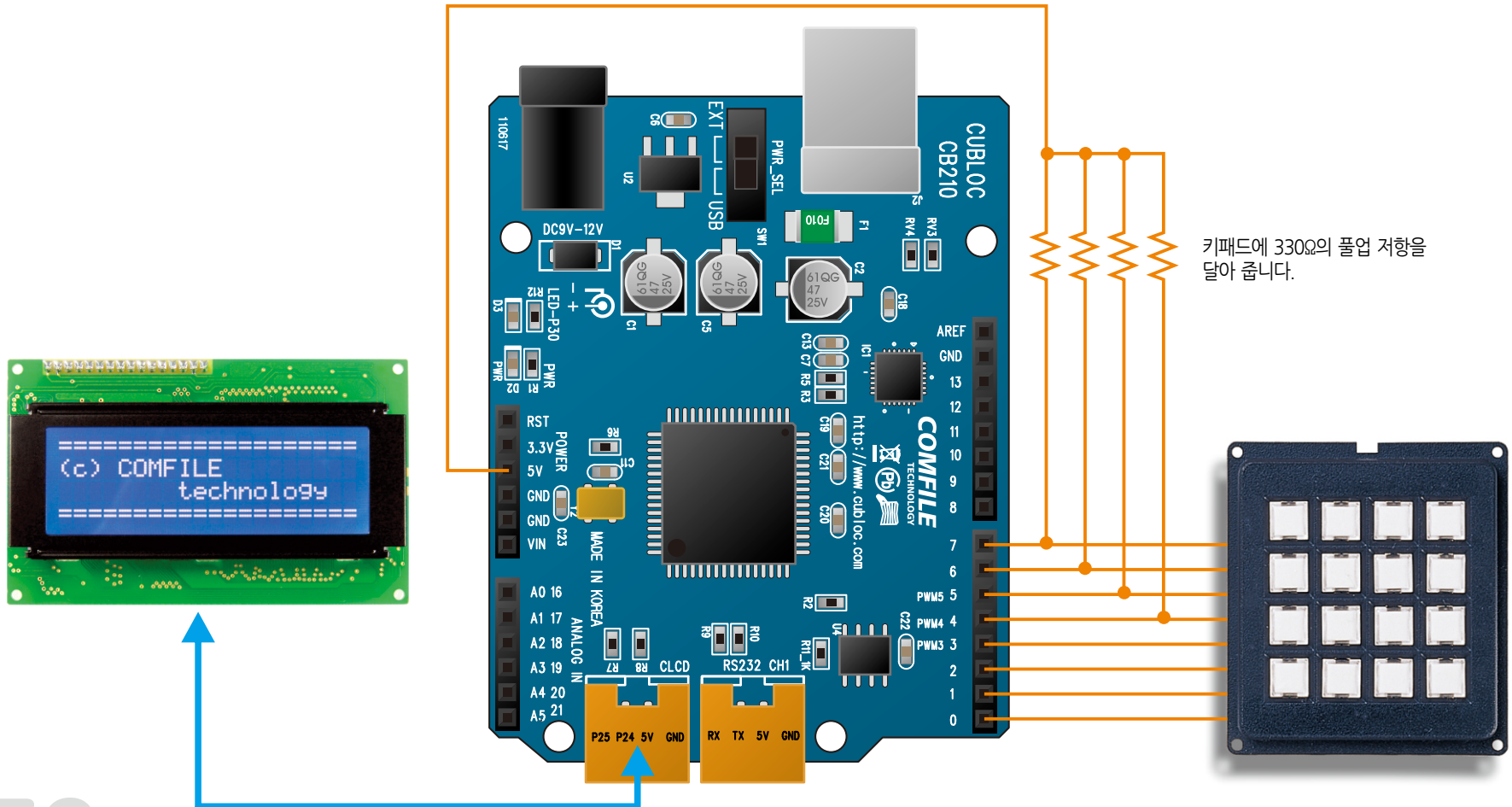
실제 키패드에  
마킹된 숫자

0	4	8	12
1	5	9	13
2	6	10	14
3	7	11	15

해당 위치의  
스캔 코드 값



## 입력 회로





# 키패드 입력받아 비밀번호 입력하기

```

1  Const Device = CB210
2
3  Dim KEY_PAD As Integer
4  Dim KEY_DATA As Byte
5
6  Dim REG As Long
7  Dim DATA As Long
8
9  Dim CNT As Byte
10
11 Dim PW As Integer
12 Dim A As String * 4
13
14
15 Set Display 0,0,0,50
16
17 Const Byte KEY_TABLE = (1,4,7,11,2,5,8,0,3,6,9,12,13,14,15,16) '키매트릭스 테이블
18
19 Ramclear
20
21 Cls
22 Delay 20
23 Csoff
24 CLCDOUT 0,0,"KEY DATA = "
25 CLCDOUT 0,1,"Enter = "
26 PW = 1234 '사용될 패스워드 설정
27
28 Do
29
30 KEY_PAD = Keypad(0) '키매트릭스 값을 입력 받습니다.
31
32 If KEY_PAD < 10 Then ' 키값을 0~9 사이의 값만 받아 데이터에 저장합니다.
33
34     If CNT = 4 Then '패스워드가 4자리이기 때문에 4자리를 넘었을 경우
35         A = " " '나머지 값들을 초기화 해주는 명령어입니다.
36         DATA = 0
37         CNT = 0
38         REG = 0
39     Endif
40
41     KEY_DATA = KEY_TABLE(KEY_PAD) '키 매트릭스트 받은 값을 사용하기 쉽도록 테이블을 참조한다.
42     DATA = DATA << 4 '키 값을 4bit 좌측으로 쉬프트 합니다.
43     DATA = DATA + KEY_DATA '쉬프트한 값에 새로 들어온 키값을 더합니다.

```

```

44
45 Do While Keypad(0) < 255 '키가 눌렀다가 떨어질때 까지 대기 합니다.
46 Loop
47
48 Incr CNT '키가 눌리는것을 카운트 합니다.
49 CLCDOUT 0,0,"KEY DATA = ",Dec2 KEY_DATA '키값이 입력되면 LCD에 값을 표시한다.
50 REG = Bcd2bin(DATA) '키값을 쉬프트 한것을 2진수 값에서 10진수 값으로 변경 합니다.
51
52 A = "*" + A '키값이 누려졌을때 *을 표시 합니다.
53 CLCDOUT 0,1,"Enter = ",A
54
55 Elseif KEY_PAD = 15 Then 'enter에 해당하는 키를 눌렀을때를 감지 합니다.
56
57 If PW = REG Then ' 설정된 키 값과 입력된 데이터가 일치할때 lcd에 ok명령을 출력 합니다.
58     CLCDOUT 0,2, "Password OK!!!"
59 Elseif PW <> REG Then ' 설정된 키 값과 입력된 데이터가 불일치 할경우 err를 출력 합니다.
60     CLCDOUT 0,2, "Password ERR!!!"
61 Endif
62
63 Endif
64
65 Loop

```



## 새로운 명령어

### 《 》 쉬프트 명령어

해당 변수의 데이터값을 쉬프트합니다. 현재 00001111의 값을 << 4로 쉬프트를 한다면 11110000로 데이터를 밀어주는 명령어입니다. 이 명령어는 연속되는 값을 받으면서 자릿수를 변경할때 주로 사용 됩니다.

### BCD2BIN ( )

위의 쉬프트 명령을 이용하여 쉬프트를 하게 되면 쉬프트 된 값은 2진수로 변환됩니다. 이를 다시 10진수로 바꾸기 위한 명령어 입니다. 차후 사용될 값들이 2진수가 아닌 10진수 임으로 이렇게 변환 명령을 이용하여 변환해 주어야 연산이 쉬워집니다.

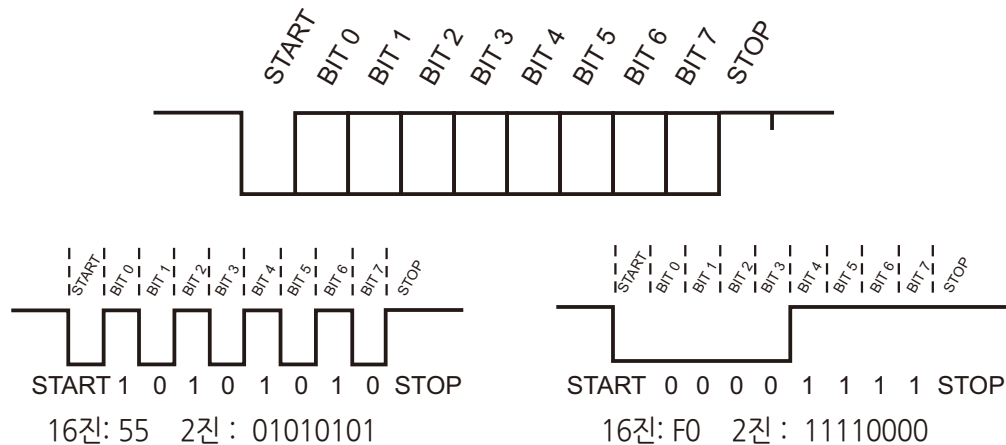
## LAB 8.

## RS232 통신



## RS232는

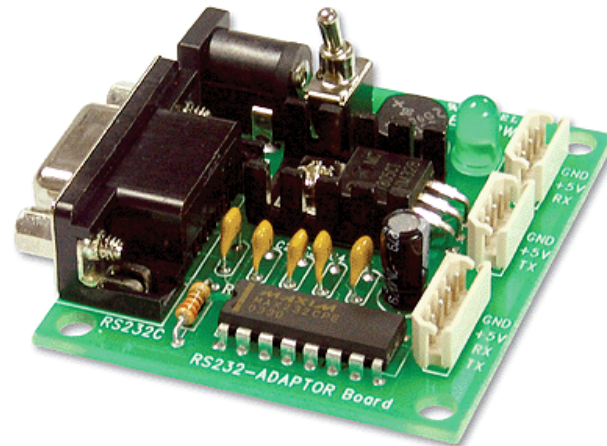
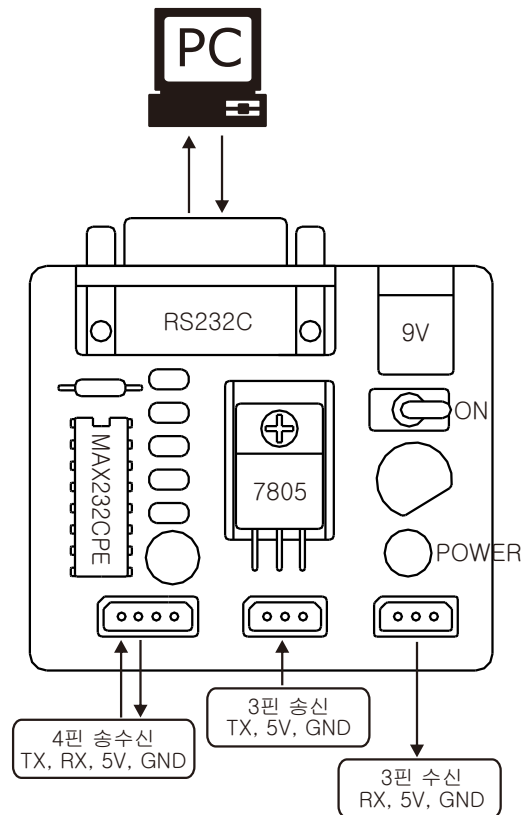
- 「Recommend Standard number 232」의 약자이며, PC와 통신하기 위한 수단으로 많이 쓰이는 방식입니다. 일반 PC에 내장되어있는 RS-232통신 포트는 9핀으로 되어있으며, 이중 CUBLOC과의 통신에서 사용되는 핀은 3핀으로 RX, TX, GND입니다.
- 최신 PC에서는 RS-232포트가 없는 것도 있으며, USB TO RS232 변환 케이블을 사용하면 RS-232통신을 사용 할 수 있습니다.
- 송수신은 8BIT의 데이터를 시간으로 나누어 해당 데이터를 분석하게 됩니다. 이 속도는 보레이트로 표시를 하는데 초당 전송 가능한 BIT수를 나타냅니다.



CB210에 내장되어 있는 RS232통신 레벨은 5V 입니다.

이 레벨을 PC와 같이 통신하기 위해서는 그에 맞는 레벨로 맞추어 주어야 하는데, 이를 위해서 RS232 Adapt Board가 필요합니다.

이 보드에는 MAX232라는 칩을 이용하여 통신 레벨을 맞추어 줄 수 있도록 제작된 인터페이스 보드입니다.



```

1  Const Device = cb210
2
3  Opencom 1,19200,3,50,50    '통신 설정 명령어
4
5  Set Display 0,0,0,20
6
7  Dim RX_DATA As Byte
8
9  Ramclear
10
11 Cls
12
13 Delay 20
14
15 Do
16
17     If Blen(1,0) > 0 Then    '통신이 들어와 수신 버퍼에 0이상의 데이터가 쌓였을 경우를 조건으로 만듭니다.
18         RX_DATA = Get(1,1)    '통신 수신 명령어로 1바이트의 데이터를 해당 변수에 넣습니다.
19         Put 1, &H31,1        '송신 명령어로 &H31이라는 1바이트의 데이터를 송신합니다.
20     Endif
21
22     CLCDOUT 0,0,"RX_DATA = ",RX_DATA
23     Delay 100
24
25 Loop

```

## OPENCOM 상수 설명

OPENCOM 통신 포트, 보레이트, 통신 프로토콜, 송신 버퍼, 수신 버퍼

■ **통신 포트:** CB210은 1만 사용 가능합니다.

■ **보레이트:** 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, 76800, 115200, 230400

보레이트는 다른 기기와 연결할 때 같은 속도로 맞추어 사용해야 합니다. 서로 속도가 다를 경우 데이터가 정상으로 받아지지 않으니 유의해야 합니다.

■ **통신 프로토콜:** 기본 프로토콜로 가장 많이 사용되는 프로토콜은 3입니다. 8비트, NONE패리티, 1스톱비트의 설정으로 되어있습니다.

■ **송,수신 버퍼:** 통신을 송,수신 하기 위한 여유 공간입니다. 이 공간 만큼 송수신 할 수 있습니다.

```

1  Const Device = cb210
2
3  Opencom 1,19200,3,50,50    '통신 설정 명령어
4  On Recv1 Gosub RX_RECV    '통신 인터럽트 명령어
5
6
7  Set Display 0,0,0,20
8
9  Dim RX_DATA As Byte
10
11  Ramclear
12
13  Cls
14
15  Delay 20
16
17  Do
18
19      CLCDOUT 0,0,"RX_DATA = ",RX_DATA
20      Delay 100
21
22  Loop
23
24
25  RX_RECV :                '통신인터럽트 영역입니다.
26
27      RX_DATA = Get(1,1)    '통신 수신 명령어로 1바이트의 데이터를 해당 변수에 넣습니다.
28      Put 1,&H31,1         '송신 명령어로 &H31이라는 1바이트의 데이터를 송신 합니다.
29
30  Return

```



## 새로운 명령어

### ON RECV 1 GOSUB xxxx

통신 인터럽트 명령어입니다. GOSUB에 입력된 라벨로 인터럽트됩니다.

수신 데이터가 들어오게 되면 메인 루틴의 현재 실행 중인 명령어를 끝내고 인터럽트가 실행되며, 라벨~RETURN 영역까지의 명령어들을 실행합니다.

RETURN명령을 만나게 되면 메인 루틴의 마지막 실행했던 명령어의 아래 명령어부터 다시 실행 합니다.

앞의 내용에선 통신의 수신 버퍼를 DO, LOOP사이에서 감시하도록 하였습니다.

그렇다면 위의 프로그램이 길어지고 시간 지연이 많다면 수신 버퍼를 확인하는 시간이 길어지고 동작이 늦어지게 됩니다.

통신이 언제 들어올지 알 수 없기 때문입니다. 외부 기기와 연결한 상태에서 외부기기의 통신이 언제 들어오는 지를 알 수 없습니다.

이런 문제점을 해결하기 위해서는 통신이 들어오자마자 반응할 수 있는 인터럽트라는 기능이 필요 합니다.

CUBLOC에는 통신 인터럽트 명령을 이용하여 수신 받는 즉시, 해당 서브영역으로 이동하여 해당 명령어를 실행 할 수 있습니다.

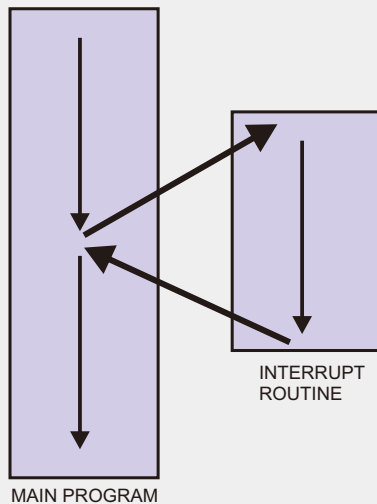
## 인터럽트

### 인터럽트란?

외부의 입력 또는 통신을 별도의 감시 영역에 두었다가, 외부의 입력 또는 통신이 있을 경우 이를 먼저 실행하도록 하는 기능입니다.

CUBLOC는 베이직 언어를 기반으로 동작하게 됩니다. 이는 순차 제어이며, 순차 제어란 위의 명령어 순서대로 차례대로 실행하는 것을 말합니다.

순차 제어 시간차로 발생하는 외부 입력과 통신을 받기 위해서는 이 인터럽트 기능을 활용해야만 합니다.

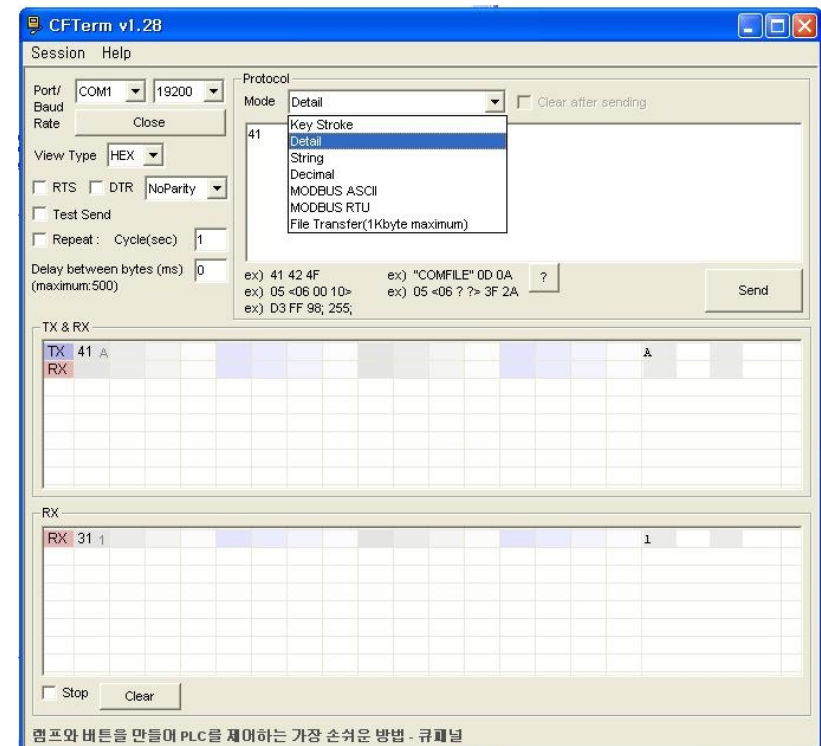
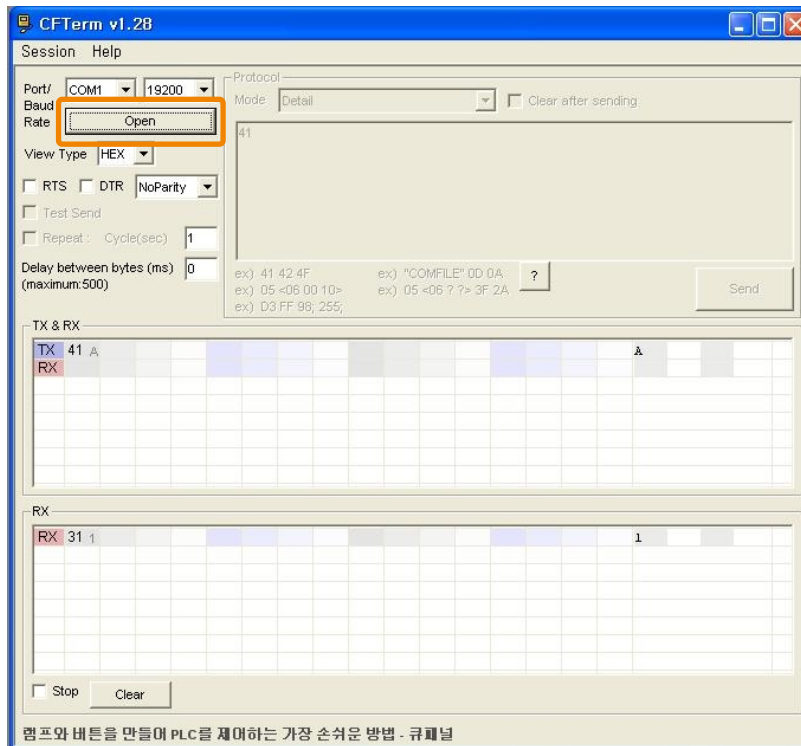


인터럽트 명	설 명
On Timer	일정 시간 간격으로 인터럽트를 발생
On Int	외부 핀 입력의 상태 변화를 감지하여 인터럽트를 발생
On Recv	RS232 수신이 발생되면 인터럽트 발생
On LadderInt	LADDER에서 인터럽트 요구가 있을 때 인터럽트 발생
On Pad	키패드 등에서 Pad통신 요구가 있을 때 인터럽트 발생

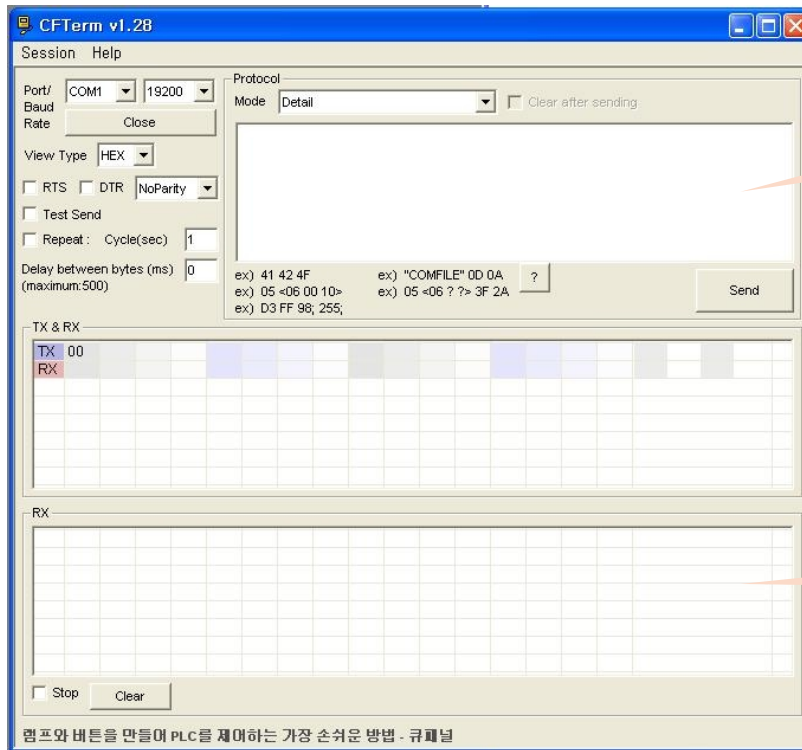
# CF Term (통신 프로그램)

PC에서 시리얼 통신을 보낼수 있는 프로그램을 알아보도록 하겠습니다.

CFTerm이라는 프로그램으로 아스키코드를 시리얼로 전송 및 전송받을 수 있는 프로그램입니다.







해당 영역에 아스키코드 및 문자형의 데이터를 넣고  
Send를 누르게 되면 해당 데이터가 시리얼통신으로 전송 됩니다.

송신 데이터가 표시되는 부분입니다.  
CUBLOC에서 작성한 1바이트 전송 명령어를 사용하게 되면  
해당 데이터가 표시 됩니다.

# 04

## 레더로직 프로그램

레더 로직을 처음 공부하는 사람들을 위해 레더의 기초에 대해서 공부해보도록 하겠습니다.

- 레더로직 프로그램이란?
- 레더 사용 방법
- 논리 회로
- 자기 유지 회로
- 인터록 회로
- 타이머 회로
- 카운터 회로
- 베이직과 레더의 연동



## 레더 프로그래밍

CUBLOC에서는 베이직 소스외에 레더를 사용 할 수 있습니다.

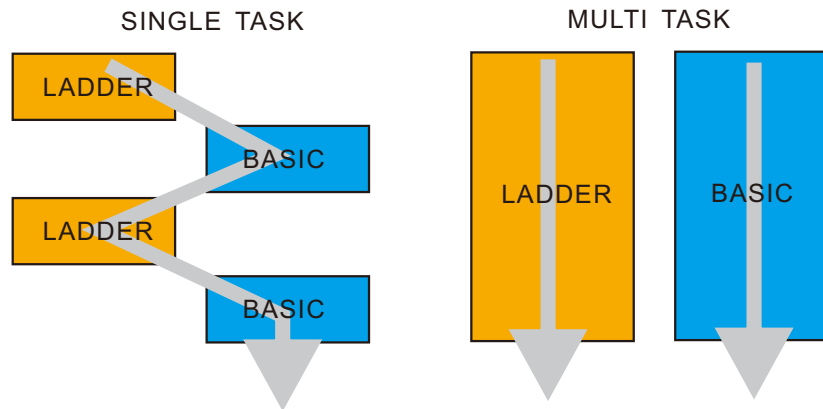
레더란, PLC에서 사용되는 문법으로 그림을 그리듯이 논리 회로를 그리는 방식으로 사용합니다.

베이직 소스와는 달리, 순차 제어가 아닌 병렬 제어 방식이라는 점이 가장 큰 차이점이라고 볼 수 있습니다.

레더의 다른 특징은 베이직 소스 처럼 변수를 사용하는 것이 아니라, 기본적으로 설정되어 있는 메모리를 사용해야 합니다.

16BIT 데이터가 들어가는 D영역 카운터 값이, 들어가는 C타이머 값이 들어가는 T영역으로 나뉘져 있으며,  
1BIT의 내부 릴레이 영역인 M영역이 있습니다.

큐블록의 레더는 약 10ms의 속도의 스캔속도를 가지고 있으며, 해당 스캔속도로 병렬 처리를 하게 됩니다.



# 레더 화면 구성

베이직 편집 화면 또는  
레더 편집 화면으로  
바꾸는 탭(Tab)입니다.  
Ladder Mnemonic은  
레더 컴파일 후 번역  
결과를 볼 수 있는  
탭입니다.

왼쪽의 숫자는  
라인번호입니다.

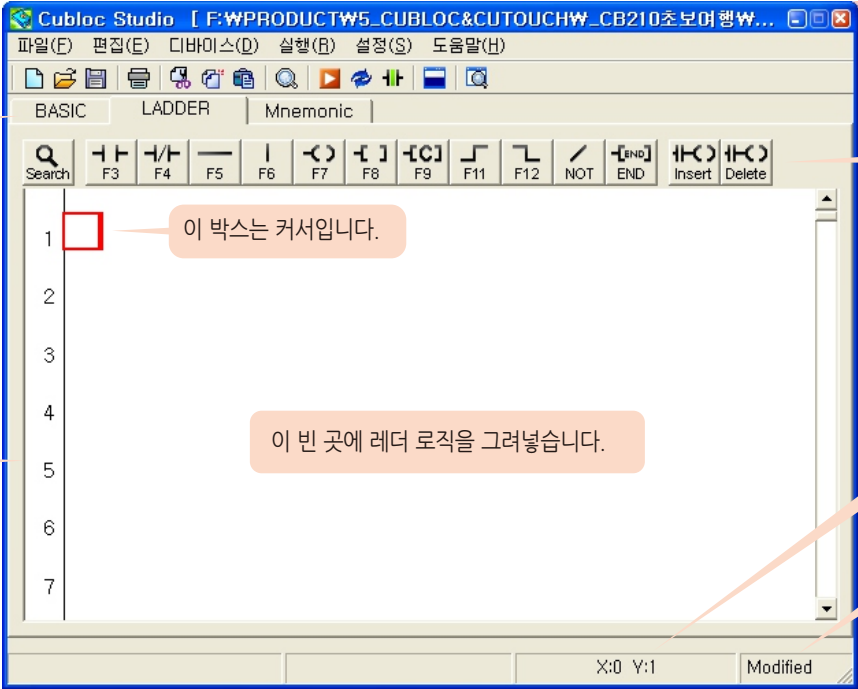
이 박스는 커서입니다.

이 빈 곳에 레더 로직을 그려넣습니다.

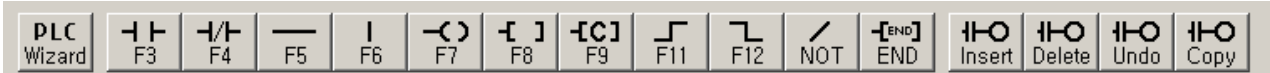
레더 심볼이 배치된  
아이콘 톨바 입니다.  
레더 작성에 필요한  
기본적인 요소를 보기  
쉽게, 그리고 선택하기  
쉽게 톨바 형식으로  
배치시켜 놓은 것입니다.

현재 레더 커서가 위치한  
곳의 좌표입니다.

수정된 적이 있으면  
Modified라고 표시됩니다.



PLC 셋업 마법사  
A 접점  
B 접점  
정지  
출력 심볼  
Function 심볼  
카운터 심볼  
상승 미분  
하강 미분  
Not 심볼  
End 라인 표시  
라미입  
라미제  
삭제된 라미복구  
바로 위에 있는 라미  
똑같은 라미 표시



## 레더의 메모리 영역

CUBLOC의 레더에는 메모리 영역이 한정되어 있습니다

베이직을 중심으로 되어있기 때문에, 레더영역의 메모리는 베이직에 비해 적을 수 있으니, 메모리 영역을 확인하여 사용해야 합니다.

종 류	릴레이명칭	사용범위	기 능
입출력 릴레이	P	P0~P127	외부 포트
내부 릴레이	M	M0~M511	내부상태 보존
특수기능 릴레이	F	F0~F127	시스템 상태

### [BIT 영역]

P영역의 경우 0~21까지 사용이 가능합니다.

종 류	릴레이명칭	사용범위	기 능
타이머	T	T0~T99	타이머 값 보존
카운터	C	C0~C49	카운터 값 보존
데이터	D	D0~D99	범용 데이터 보존

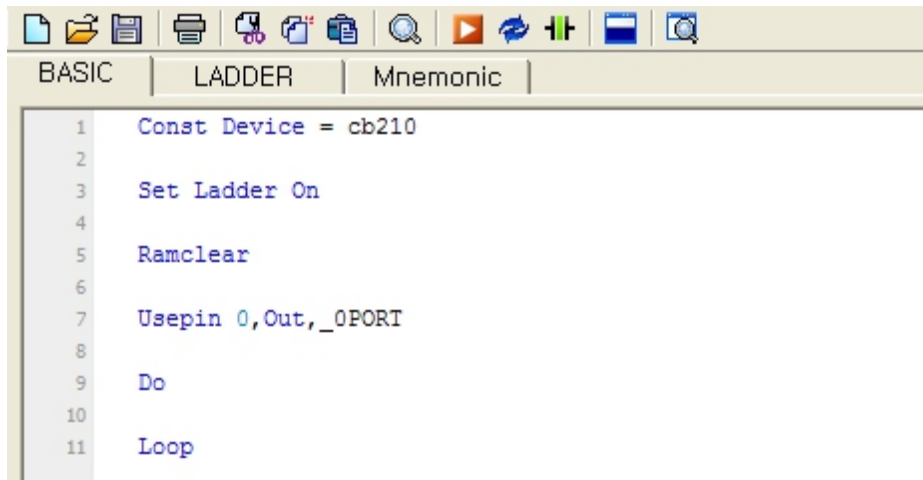
### [WORD 영역]

T영역과 C영역의 경우, BIT영역과 WORD영역을 동시에 가지고 있으며, 사용 가능한 용량도 같습니다.

# 레더 기본 설정

레더로직을 사용하기 위해선 베이직에서 기본적인 설정을 해주어야 합니다.

CUBLOC의 경우 베이직을 메인으로 사용되며, 레더로직은 보조 기능으로 사용하시기를 추천해 드립니다.



## SET LADDER ON

## USEPIN 인수 설명

### USEPIN , IN/OUT,

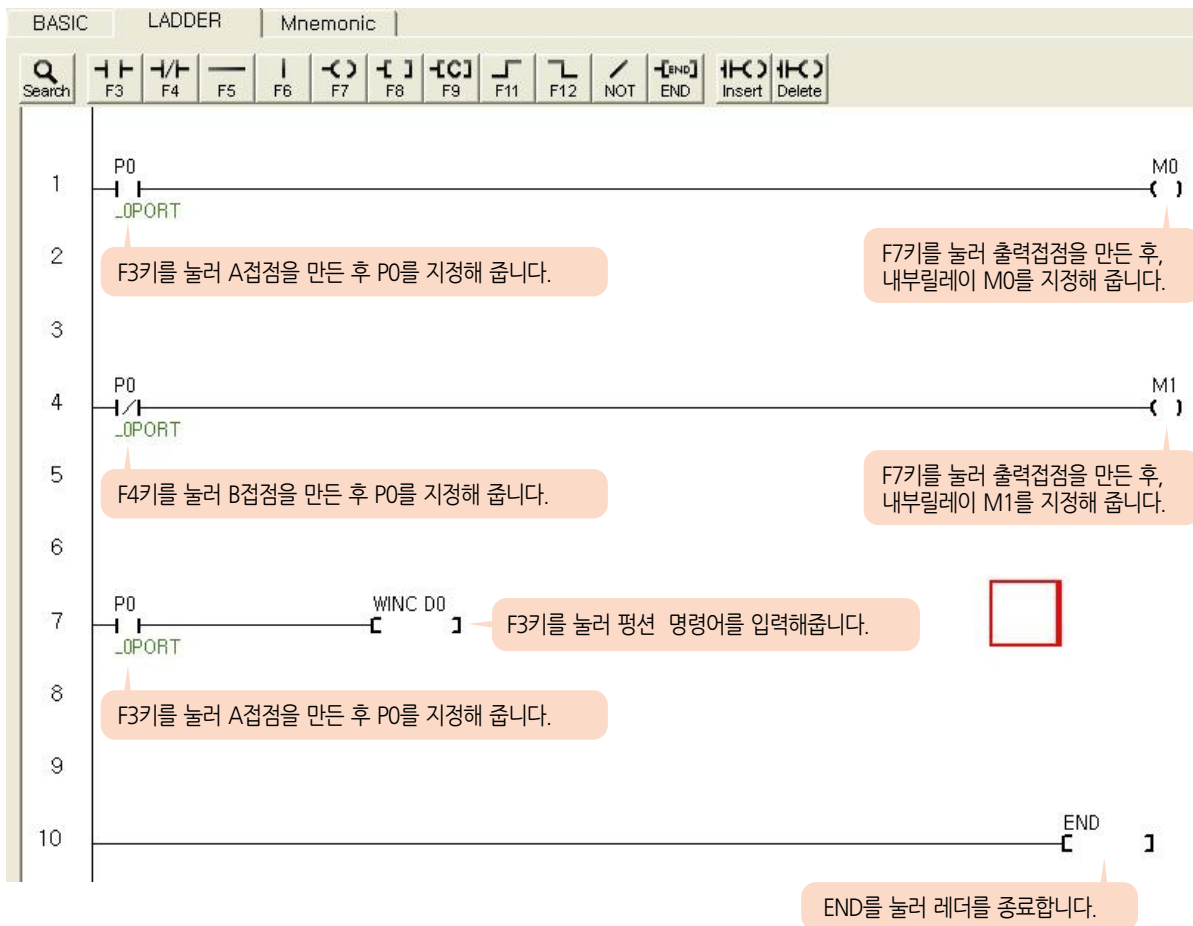
• **포트번호:** 베이직에서 사용하는 포트를 레더에서 사용하도록 선언합니다.

**IN/OUT:** 레더에서는 IN/OUT을 미리 설정해 놓아야 합니다. IN 또는 OUT을 설정합니다.

**별명:** 포트번호 외에 직접 구분 할 수 있는 별명을 정합니다.

START 또는 STOP 등등 해당 포트의 기능을 써 놓기도 합니다.

# 레더 작성



## A접점과 B접점

레더에서 가장 많이 사용되는 접점으로 A접점과 B 접점이 있습니다.

A접점은 입력이 들어오게 되면 ON이 되고, 입력이 들어오지 않을 경우 OFF 되는 접점을 말하며,  
B접점은 반대로 입력이 들어오게 되면 OFF가 되며, 입력이 들어오지 않으면 ON 되는 접점을 말합니다.



[A접점 기호 및 심볼]



[B접점 기호 및 심볼]



## 미분을 이용한 입력받기

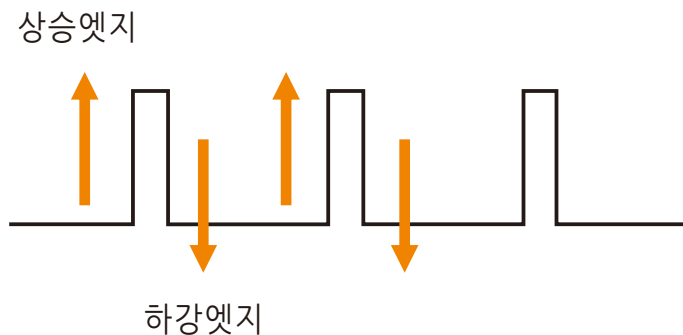
미분이란 무엇인지 알아 보도록 하겠습니다.

입력의 상승 엣지 부분을 감지하여 입력 신호를 인식하는 것을 상승미분이라고 합니다.

상승 미분을 사용하는 이유는, 어떠한 명령어를 한번만 실행 하고 싶지만 입력 신호는 레더 스캔 속도 보다 느릴 경우나, 빠른 속도의 카운터를 측정할 때 사용 합니다.

입력속도가 레더 스캔속도보다 느리기 때문에 명령어가 여러 번 실행 되는 경우를 막기 위해 사용됩니다.

상승 미분 기호를 사용하게 되면 처음 만나는 상승 엣지를 인식하며, 다음 만나는 상승 엣지 전까지 입력으로 인식하지 않습니다.



# 레더 작성시 주의사항

레더 작성 화면에서 아무 그림이나 그릴 수 있습니다.  
문제는 나중에 컴파일 할 때, 에러가 난다는 것이죠!  
컴파일 할 때 에러가 발생하지 않는 레더 작성법에 대해서 알아보겠습니다.

- 1** 회로가 중간에 끊어져 있으면 안됩니다.



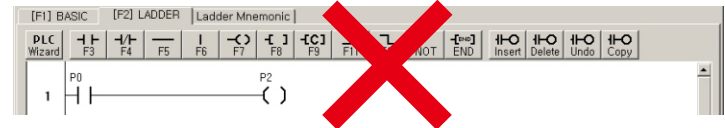
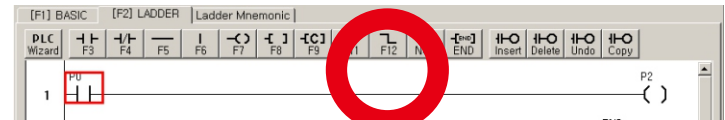
- 2** 이름없는 심볼이 있으면 안됩니다.



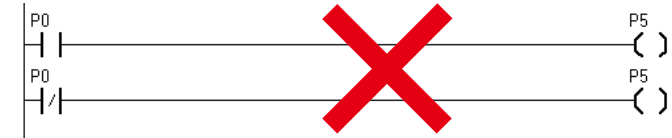
- 3** 부가 회로가 있다면, 아래 쪽으로 전개해야 합니다.



- 4** 출력 심볼은 반드시 오른쪽 끝에 위치해야 합니다.



- 5** 같은번호 릴레이가 서로다른 상황으로 중복 출력되어서는 안됩니다.



# 논리 회로

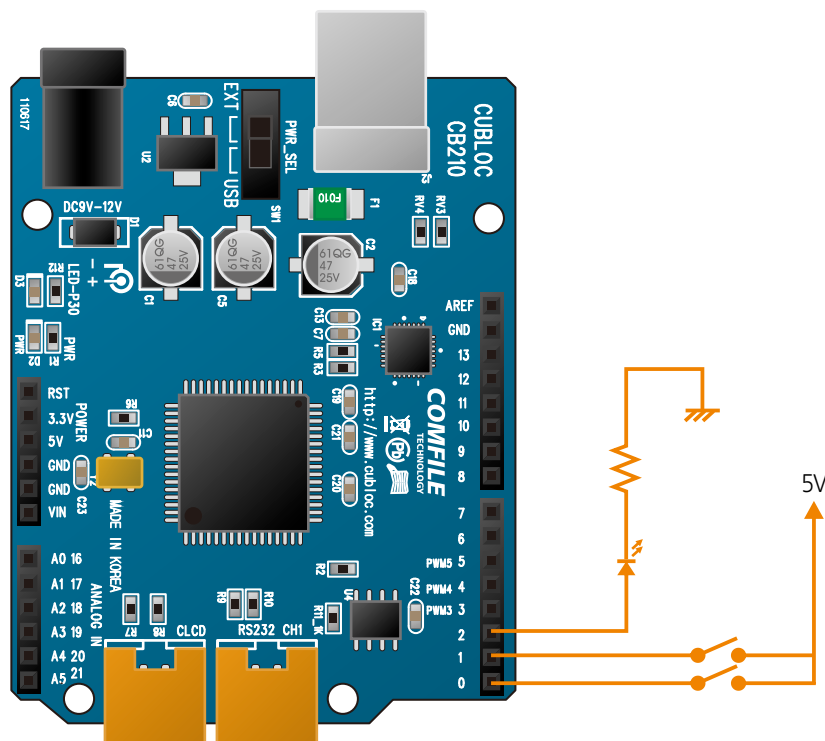


논리 회로란?

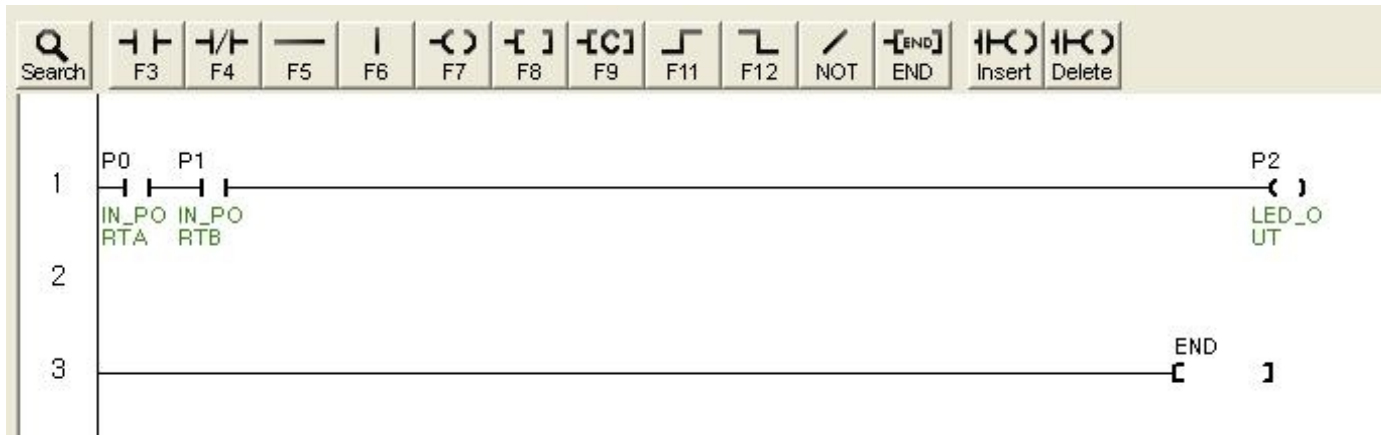
AND, NAND, OR, NOR, NOT의 논리 게이트 들을 말합니다.  
논리 게이트는 입력에 따라 논리표에 맞게 동작 하는 회로를 말합니다.

```

1  Const Device = cb210
2
3  Set Ladder On
4
5  Ramclear
6
7  Usepin 0, In, IN_PORTA
8  Usepin 1, In, IN_PORTB
9  Usepin 2, Out, LED_OUT
10
11 Do
12
13 Loop
  
```



# AND 회로의 심볼과 진리표



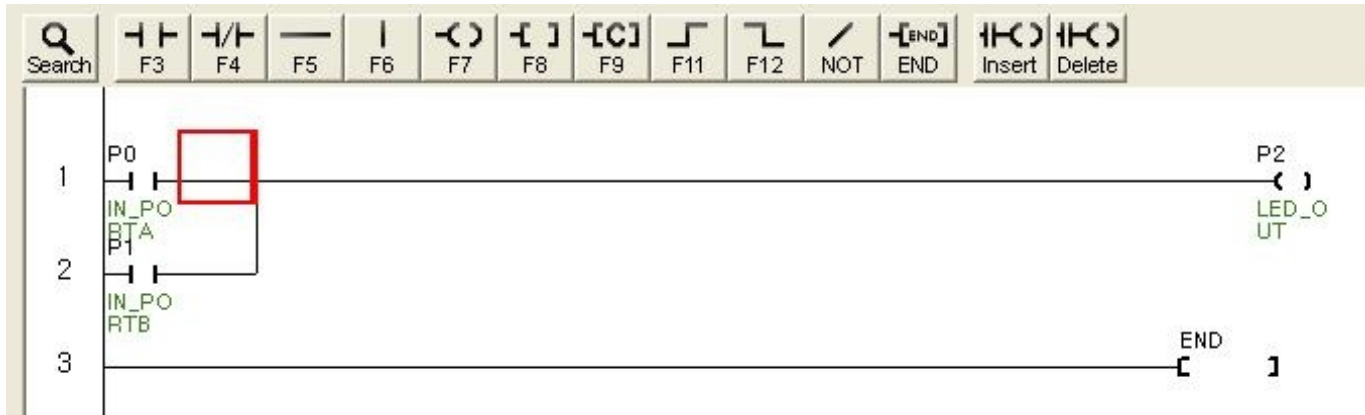
P0	P1	P2
OFF	OFF	OFF
OFF	ON	OFF
ON	OFF	OFF
ON	ON	ON



AND 회로는 모든 입력 포트에 ON이 되어야 동작을 하는 회로 입니다.

진리표를 보게 되면 P0와 P1이 동시에 ON이 되었을 경우에만, P2(LED)가 ON 되도록 되어있습니다.

## OR 회로의 심볼과 진리표



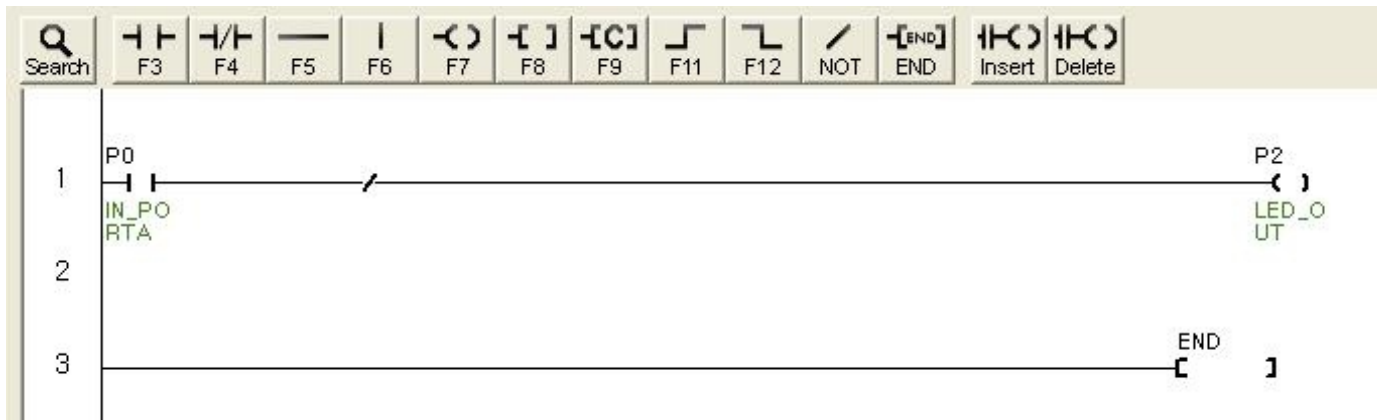
P0	P1	P2
OFF	OFF	OFF
OFF	ON	ON
ON	OFF	ON
ON	ON	ON



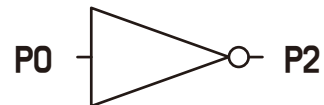
OR 회로는 모든 입력 포트 중 어떤 포트라도 ON이 되면 동작을 하는 회로입니다.

진리표를 보게 되면 P0과 P1 중 어떤 포트가 ON이 되더라도, P2(LED)가 ON 되도록 되어있습니다.

# NOT 회로의 심볼과 진리표



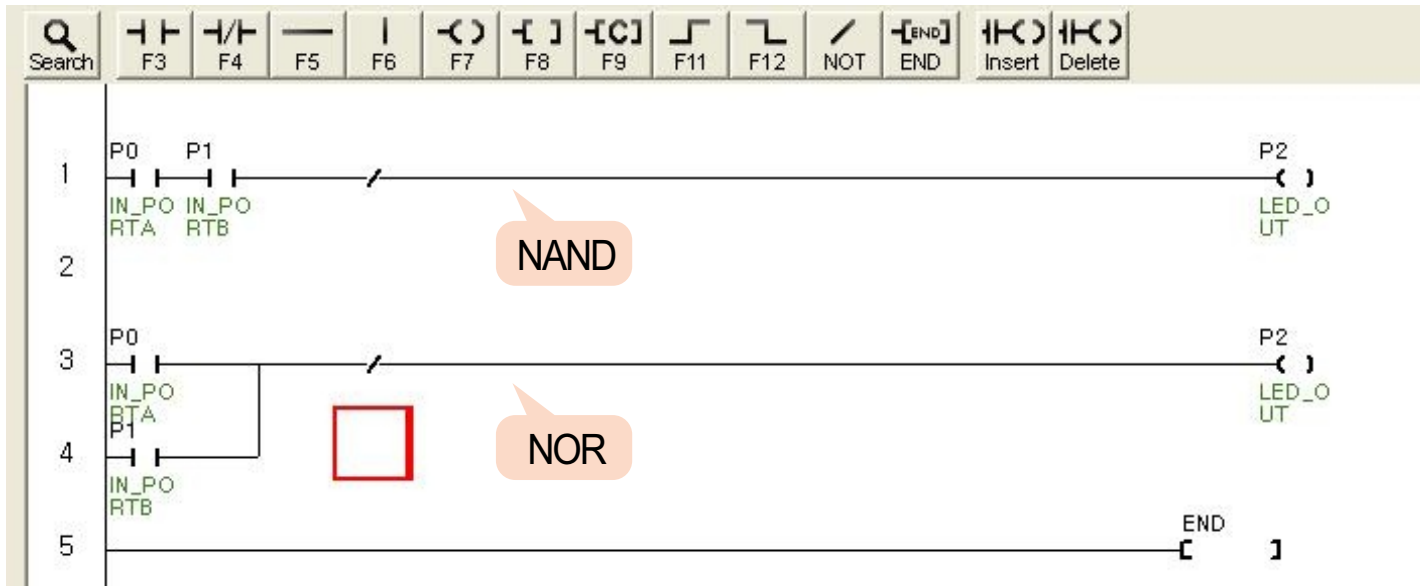
P0	P2
OFF	ON
ON	OFF



NOT 회로는 모든 입력을 반대로 실행하는 회로입니다.

P0에 입력이 OFF일 경우 P2(LED)가 ON이 되고, P0에 입력이 ON일 경우 P2(LED)가 OFF가 됩니다.

# NAND, NOR 회로의 심볼과 진리표



[NAND 회로]

P0	P1	P2
OFF	OFF	ON
OFF	ON	ON
ON	OFF	ON
ON	ON	OFF



[NOR 회로]

P0	P1	P2
OFF	OFF	ON
OFF	ON	OFF
ON	OFF	OFF
ON	ON	OFF



NAND와 NOR는 AND회로와 OR회로에 NOT회로를 추가한 회로입니다.

AND와 OR회로와 반대로 동작을 하는 회로입니다.

MEMO



# 자기유지 회로



## 자기유지 회로란?

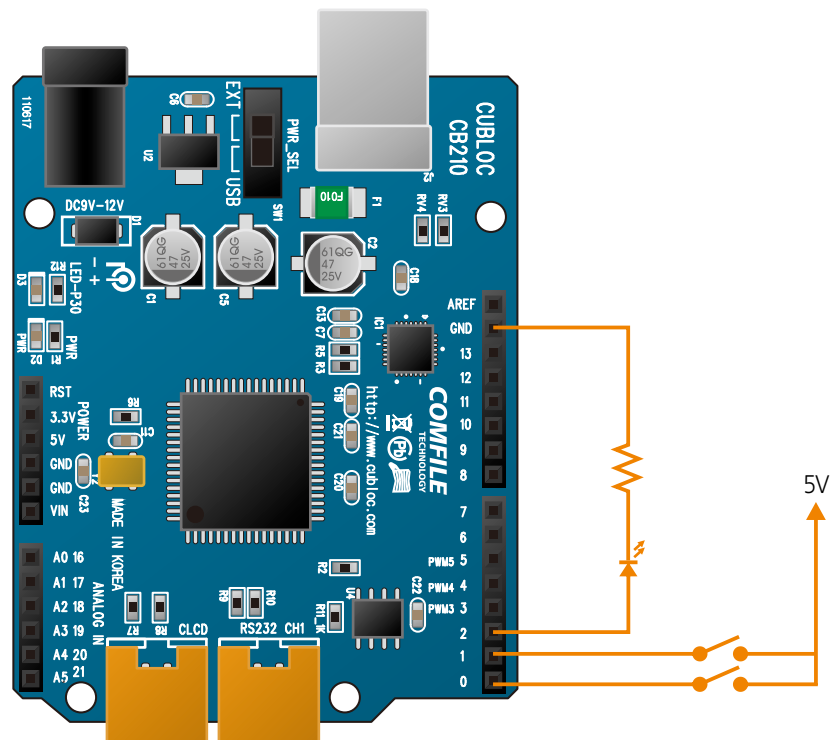
■ 입력 받은 신호를 지속적으로 유지시켜 주는 회로를 말합니다.

예로, 시작 스위치를 만들었다고 한다면,

이 스위치를 한번 누르면 지속적으로 동작을 하고, 정지 스위치를 누르게 되면 멈추는 회로를 자기유지라고 할 수 있습니다.

```

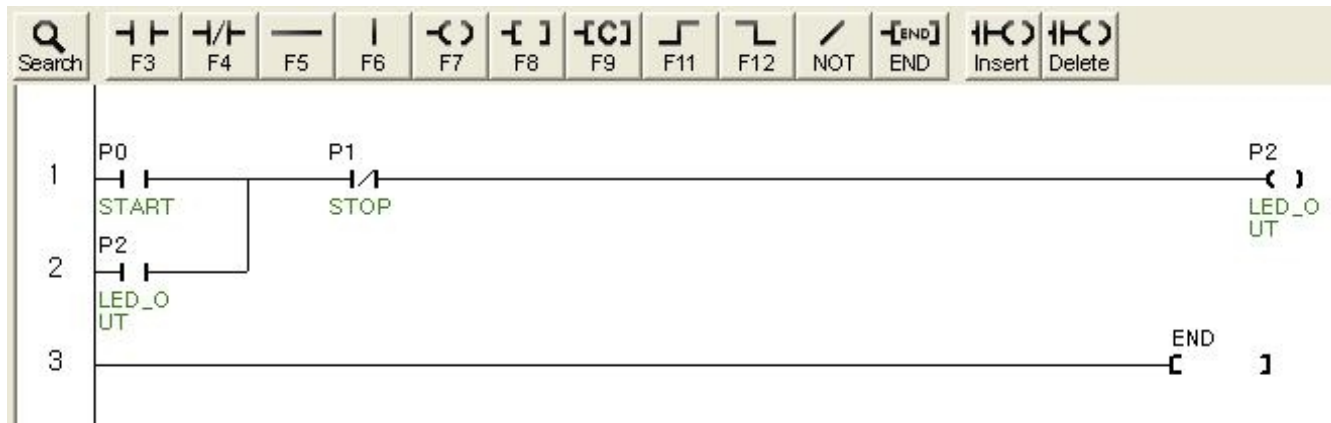
1  Const Device = cb210
2
3  Set Ladder On
4
5  Ramclear
6
7  Usepin 0, In, START
8  Usepin 1, In, STOP
9  Usepin 2, Out, LED_OUT
10
11 Do
12
13 Loop
  
```



P0(START)에 입력이 들어오게 되면, B점점인 P1(STOP)를 지나 P2(LED)가 동작 하게 됩니다.

P2(LED)포트가 동작을 하게 되면, P0(START)와 P2(LED)의 OR로 인하여 P0(START)가 입력이 들어오지 않아도 P2(LED)는 지속적으로 동작하게 됩니다.

B점점인 P1(STOP)에 입력이 들어오게 되면, P0(START)와 P2(LED)이 연결되어있던 중간 부분이 단절 되면서 동작을 멈추게 됩니다.



# 인터록 회로

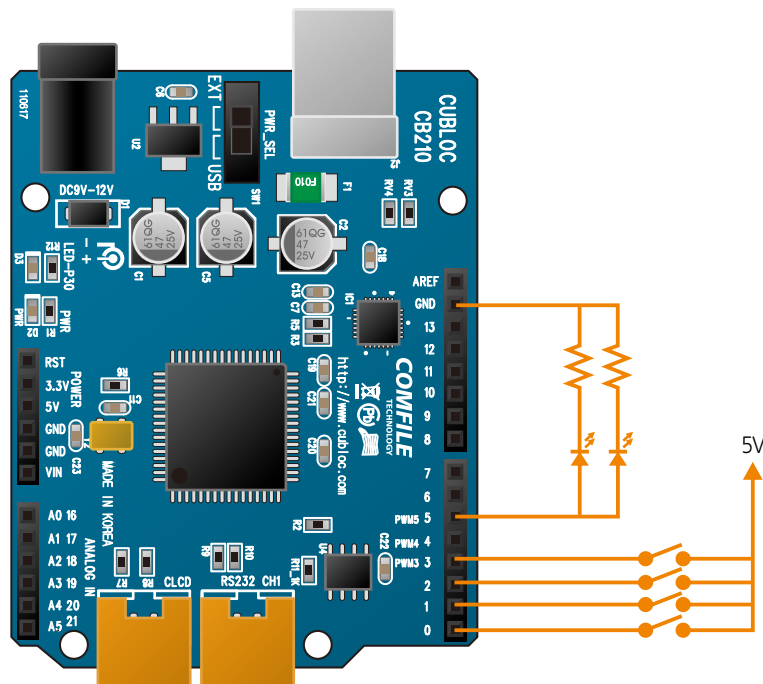


인터록 회로는...

- 2가지 동작 중 한가지 동작을 하고 있을 경우, 다른 동작은 하지 못하도록 막아 놓는 회로입니다.
- 인터록 회로를 사용 하는 이유는,  
가령 모터를 업/다운 동작을 시킬 경우, 업 스위치를 눌러 모터가 정 방향으로 돌아가고 있는 상태에서  
다운 키를 같이 눌렀을 경우, 두 동작이 충돌이 나지 않도록 하는 부분에 사용됩니다.  
모터 외에도 어떤 동작이 중복으로 동작 될 경우 제품의 충돌 위험이나 안전상의 위험이 있는 부분에 많이 사용됩니다.

```

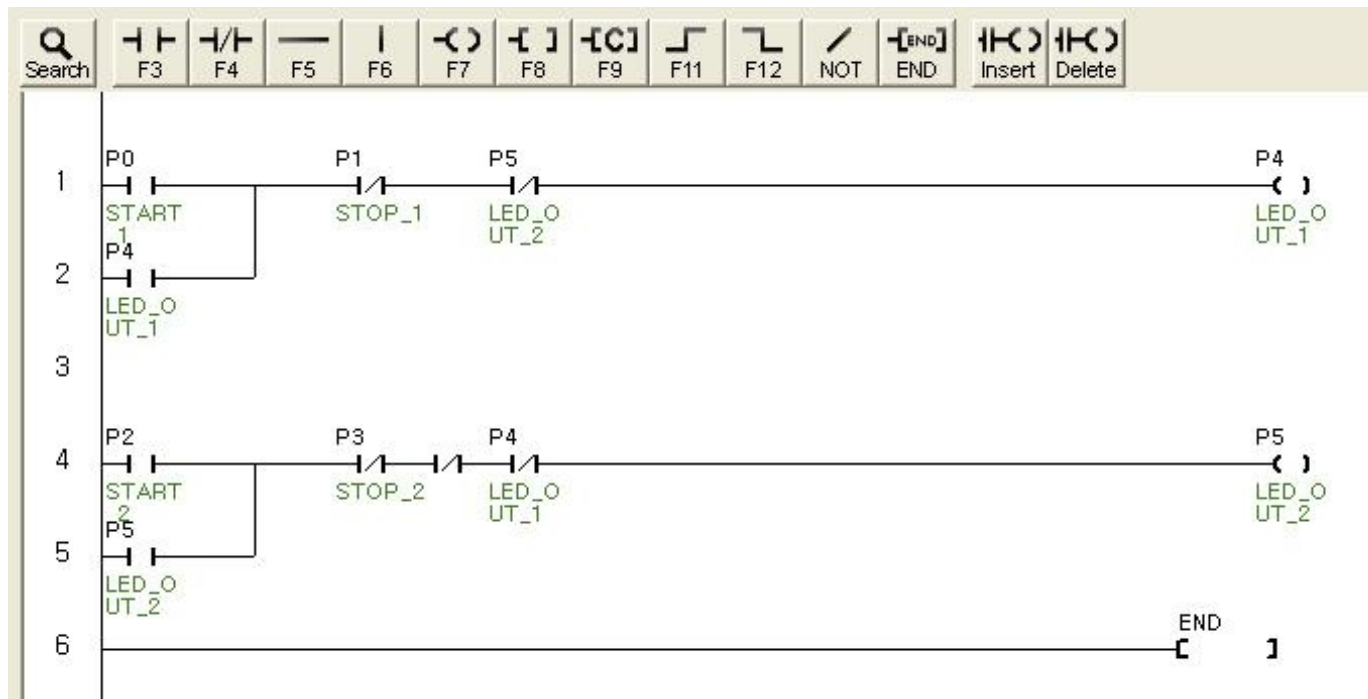
1  Const Device = cb210
2
3  Set Ladder On
4
5  Ramclear
6
7  Usepin 0,In,START_1
8  Usepin 1,In,STOP_1
9  Usepin 2,In,START_2
10 Usepin 3,In,STOP_2
11 Usepin 4,Out,LED_OUT_1
12 Usepin 5,Out,LED_OUT_2
13
14 Do
15
16 Loop
  
```



2개의 자기 유기 회로에 각각 출력에 해당되는 부분을 B접점으로 마지막 부분에 입력해 줍니다.

처음 어느 접점도 동작하지 않을 경우에는 두 접점은 동작이 가능하지만,

한 접점이 동작하게 되면, 동작 시키지 않은 접점에는 B접점으로 되어있는 출력 때문에 START를 눌러도 동작하지 않게 됩니다.



# 타이머 회로



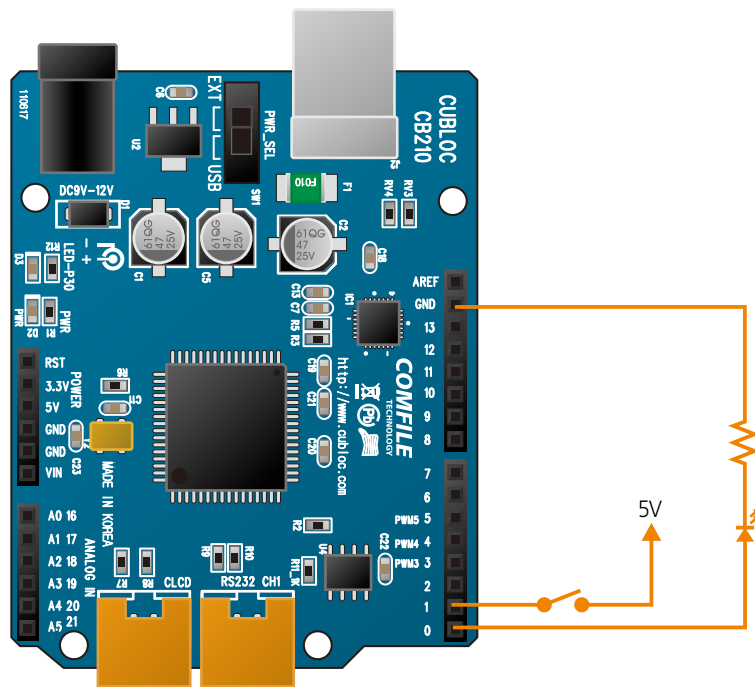
타이머 회로란?

CUBLOC 에는 내부 타이머 영역이 존재 합니다.

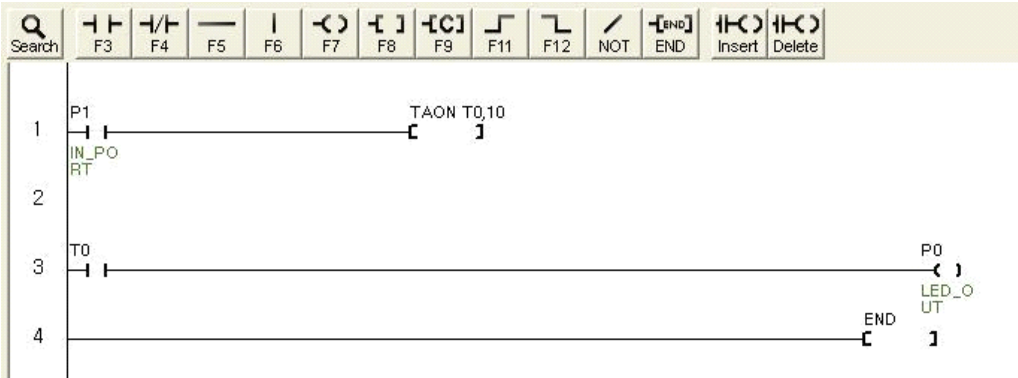
이 타이머 영역은 CUBLOC의 클럭에 의해 만들어지며, 이를 이용하여 시간을 정해놓고 동작하는 타이머 회로를 만들 수 있습니다.

```

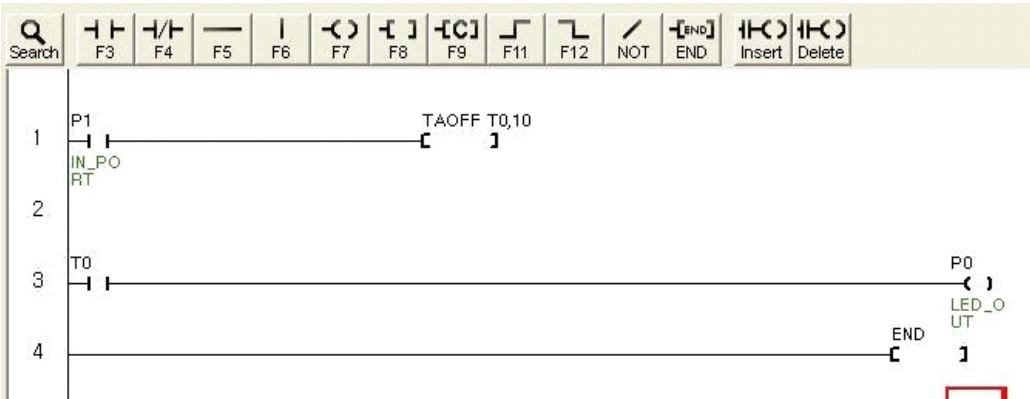
1  Const Device = cb210
2
3  Set Ladder On
4
5  Ramclear
6
7  Usepin 0,Out,LED_OUT
8  Usepin 1,In,IN_PORT
9
10 Do
11
12 Loop
  
```



CUBLOC에는 2가지의 타이머가 있습니다.  
ON타이머와 OFF타이머로, 입력을 받아 동작할 때 일정 시간이 지나면 ON되는 것과 일정 시간이 지나면 OFF되는 동작을 합니다.



ON타이머 입니다.  
1초동안 P1에 입력을 하게 되면 -[ ] 안의 숫자가 증가 됩니다.  
이 숫자가 10이 되면 T0가 켜지게 되고,  
이에 연결되어있는 P0가 동작 하게 됩니다.



OFF타이머 입니다.  
P1에 입력을 하면 P0가 동작 하며 -[ ] 안의 숫자가 감소되며,  
숫자가 0이 되었을때 P0이 꺼지게 됩니다..

# 카운터 회로

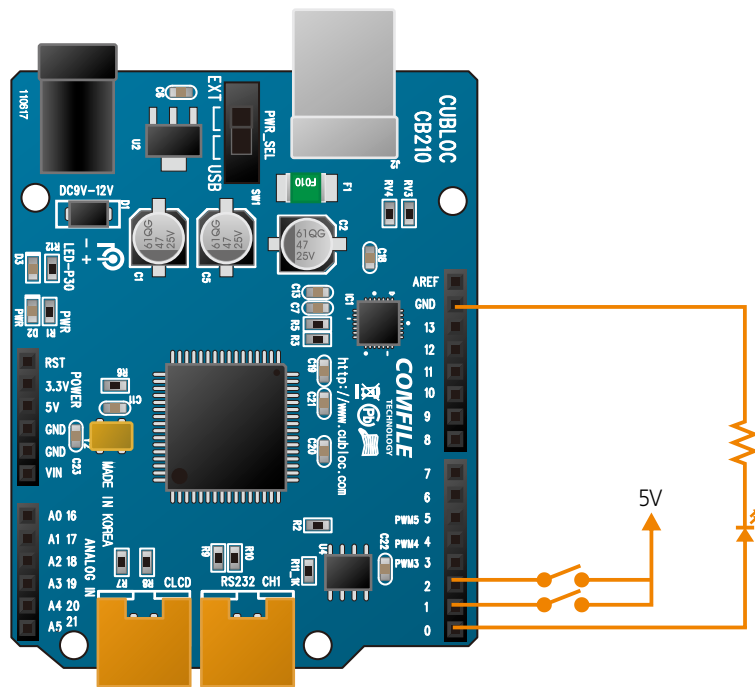


## 카운터 회로란?

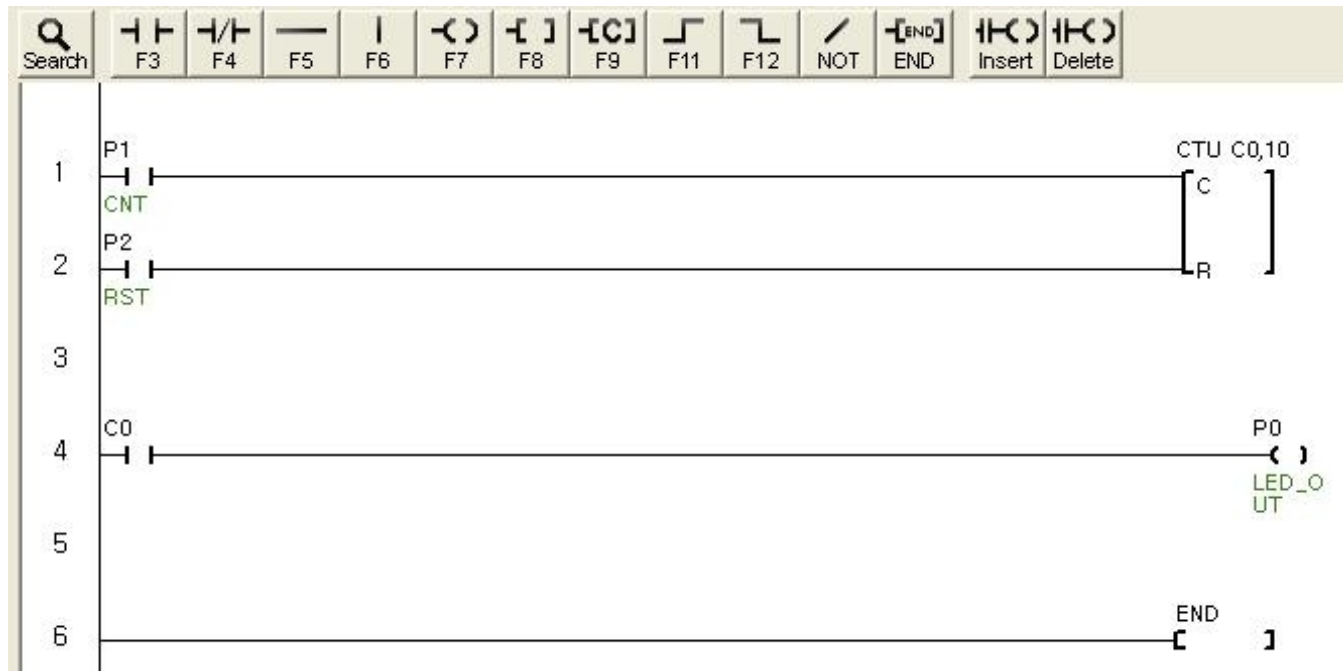
- 카운터 회로는 입력 받은 값을 축척하는 회로를 말합니다.  
모터의 회전수 또는 제품의 생산 개수 등을 측정할 때 사용합니다.

```

1  Const Device = cb210
2
3  Set Ladder On
4
5  Ramclear
6
7  Usepin 0, Out, LED_OUT
8  Usepin 1, In, CNT
9  Usepin 2, In, RST
10
11 Do
12
13 Loop
  
```



P1(CNT)스위치를 누르면 카운터가 올라가며 목표치 값이 될 때 까지 카운터가 됩니다.  
 목표치를 채우게 되면 C0가 켜지게 되며 C0에 연결된 P0(LED)가 켜지게 됩니다.  
 P2(RST)스위치는 리셋 스위치로 초기 값으로 돌아가게 됩니다.





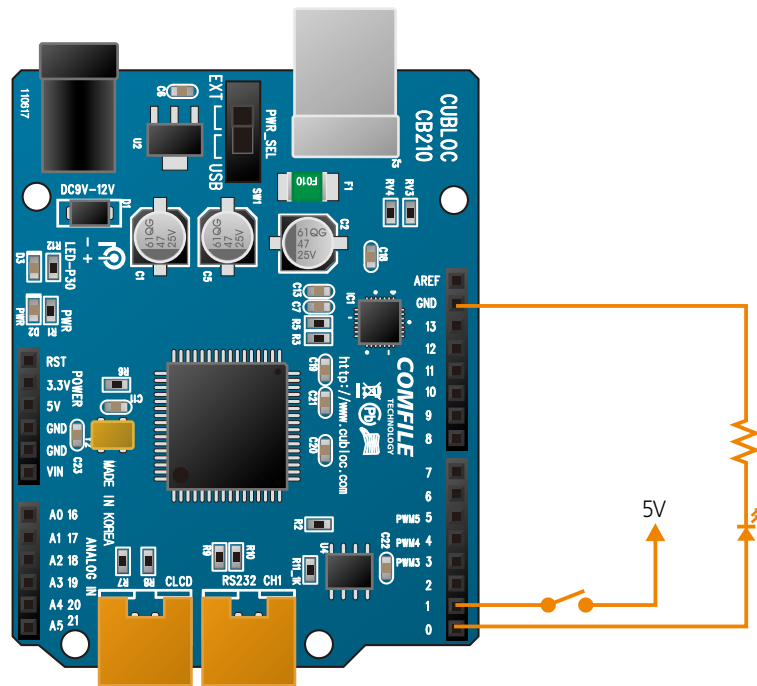
# 베이직과 레더의 연동



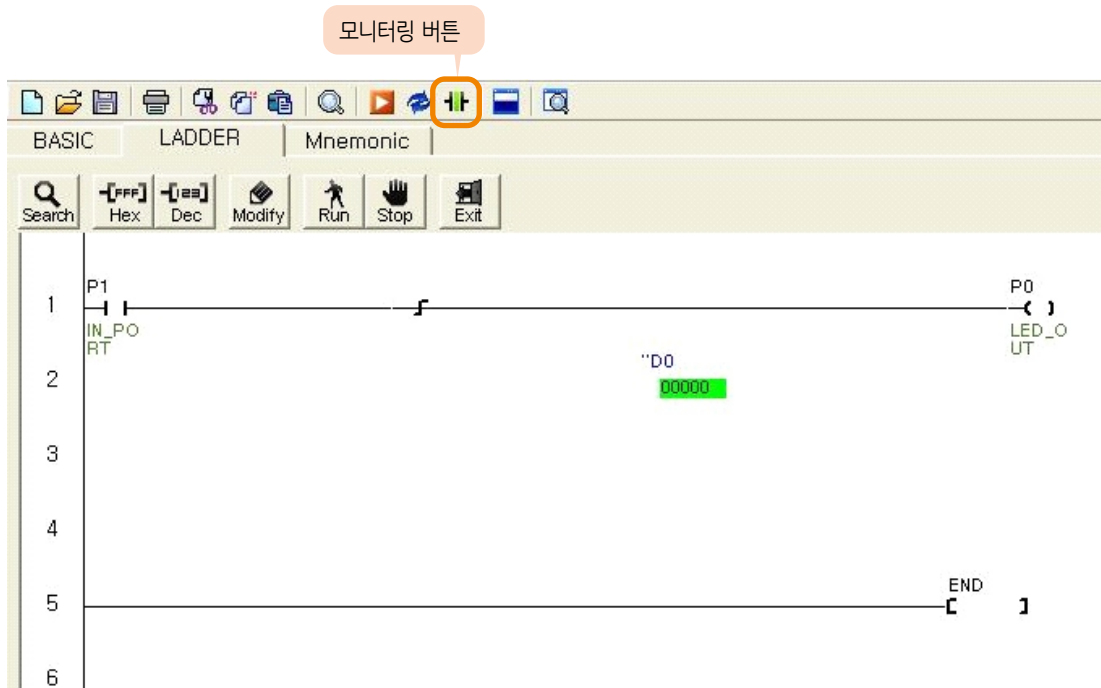
- CUBLOC은 베이직과 레더를 연동하여 사용이 가능합니다.  
레더에서 사용한 메모리를 베이직으로 가져와 사용 할 수 있으며, 반대로 베이직 데이터를 레더 영역으로도 옮겨갈 수 있습니다.
- 단, 레더에서 사용되는 I/O 포트는 레더에서 사용하도록 미리 정해 놨기 때문에,  
베이직에서 사용하려면 \_P()로 변경하여 사용해야 합니다.

```

1  Const Device = cb210
2
3  Set Ladder On
4
5  Dim DATA As Integer
6
7  Ramclear
8
9  Usepin 0, Out, LED_OUT
10 Usepin 1, In, IN_PORT
11
12 Do
13
14
15     If _P(0) = 1 Then
16         Incr DATA
17         _D(0) = DATA
18     Endif
19
20
21 Loop
  
```



P1을 누르게 되면 레더에 연결되어 있는 P2(LED)가 ON 되며, 베이직의 DATA 변수에 값이 올라가게 됩니다.  
이를 레더의 D0영역에 넣어 레더에서 모니터링 할 수 있게 되어있습니다.



레더 모니터링이란, 현재 레더의 상태를 볼 수 있는 기능입니다.

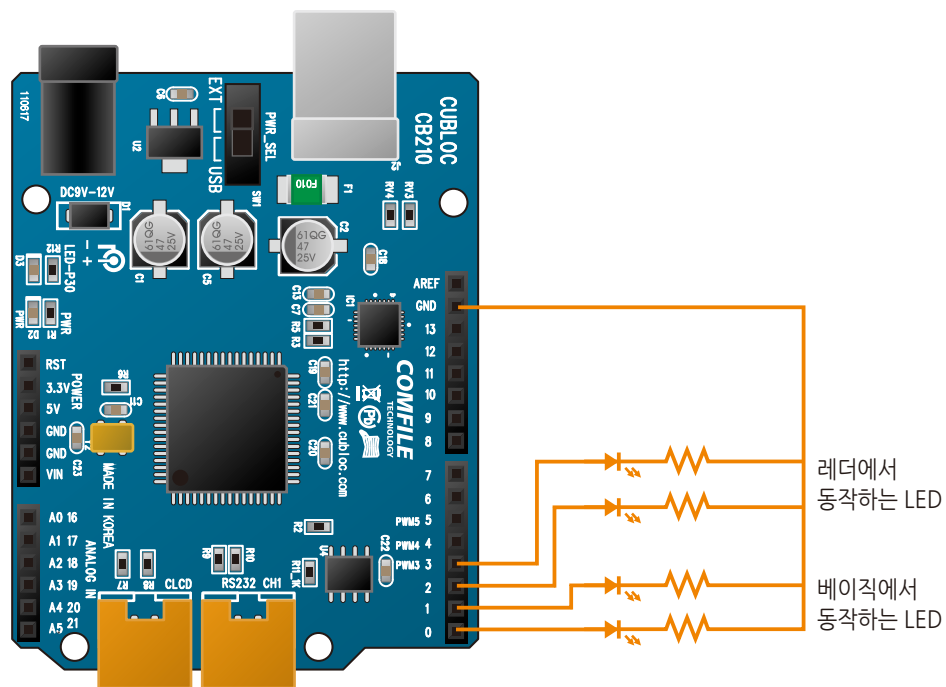
D영역과 같은 워드 영역을 모니터링 할 경우, ' ' 를 써주고 뒤에 영역을 써넣어 주시면 됩니다.

\* 레더의 모니터링과 베이직의 디버그는 같은 통신 채널을 사용하기 때문에 중복으로 사용할 수 없습니다.

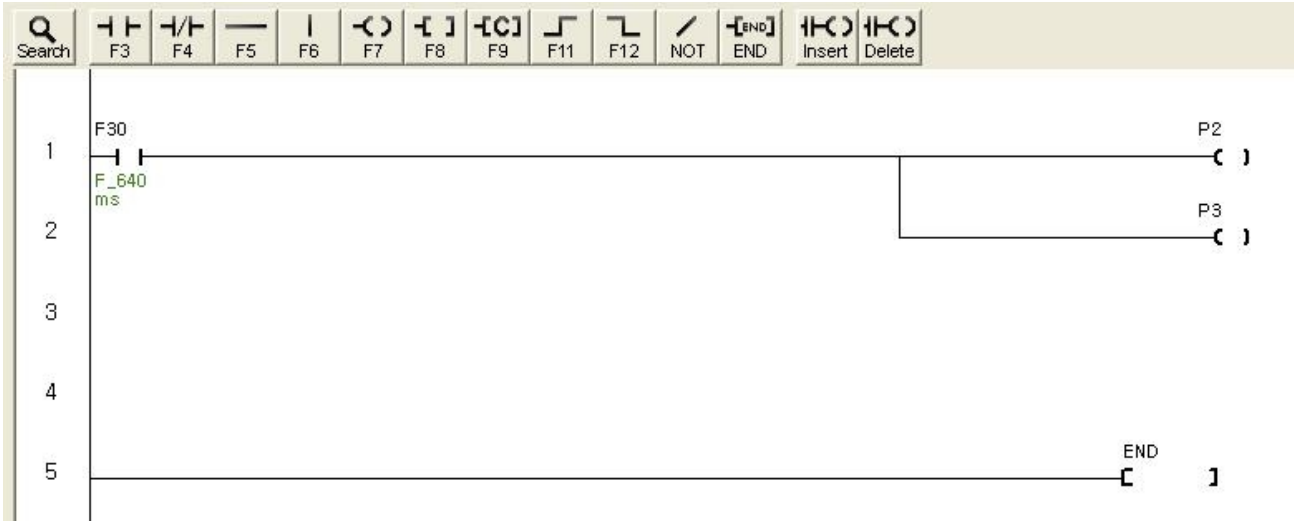
모니터링을 사용 중 에는 디버그 명령을 OFF해주셔야 합니다.

CUBLOC에서는 베이직과 레더는 멀티테스킹으로 동작합니다.

베이직과 레더가 충돌 없이 동작하기 때문에, 베이직의 영향을 받지 않고 레더에 작성된 기능들을 수행 할 수 있습니다.



```
1  Const Device = cb210
2
3  Set Ladder On
4
5  Dim DATA As Integer
6
7  Ramclear
8
9  Usepin 0,Out,LED_OUT
10 Usepin 1,In,IN_PORT
11
12 Do
13
14
15     If _P(0) = 1 Then
16         Incr DATA
17         _D(0) = DATA
18     Endif
19
20
21 Loop
```



베이직의 경우 P0과 P1이 100mS간격으로 ON/OFF되는 동작을 하며,  
레더의 경우 P2와 P3이 동시 640mS간격으로 ON/OFF 되는 동작을 하게 됩니다.  
두 동작은 별개로 동작을 하며 멀티테스킹 동작을 한다는 것을 확인 할 수 있습니다.

종류	릴레이명칭	사용범위	기능	단위	BASIC에서의 참조명칭
입출력 릴레이	P	P0~P127	외부 포트	1비트	_P(0)~_P(127)
내부릴레이	M	M0~M511	내부상태 보존	1비트	_M(0)~_M(511)
특수기능 릴레이	F	F0~F127	시스템 상태	1비트	_F(0)~_F(127)
타이머	T	T0~T99	타이머 진행값	16비트	_T(0)~_T(99)
카운터	C	C0~C49	카운터 진행값	16비트	_C(0)~_C(49)
데이터 영역	D	D0~D99	데이터 값	16비트	_D(0)~_D(99)

베이직과 레더의 연동시 변수와 메모리의 사용 방법의 차이가 생기게 됩니다.  
이를 위해 표를 이용하여 각각 연동시 사용되는 방법을 알아보도록 하겠습니다.



# 05

## 응용 예제

CB210을 활용한 실전 응용사례를 가지고,  
소스 프로그램 작성 테크닉과 활용 방법에 대하여 공부해 보도록 하겠습니다.

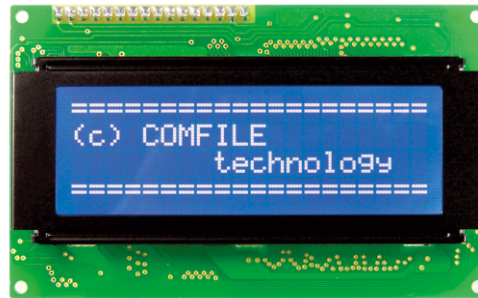
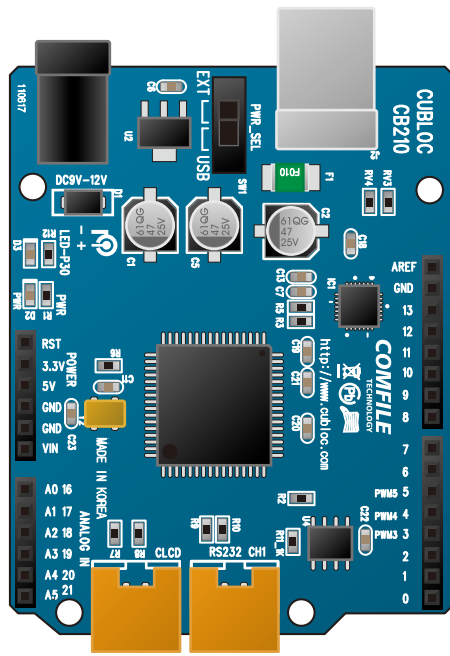
- 키패드를 이용한 계산기
- PWM을 이용한 전자 피아노
- AD를 이용한 RC서보모터 제어
- 스위치 입력을 이용한 릴레이 동작
- 스텝 모터 컨트롤
- DC 팬 모터를 이용한 선풍기



# 키패드를 이용한 계산기

키패드와 CUBLOC의 연산 기능을 이용하여, 전자 계산기를 만들 수 있습니다.

키패드의 입력을 받아 LCD에 출력 후, +, -, \*, / 의 연산을 합니다.



## 소스

```

1  Const Device = CB210
2
3  Dim KEY_PAD As Integer      '키패드의 실시간 데이터 변수
4  Dim KEY_DATA As Byte       '실제로 키패드 데이터 저장
5
6  Dim REG As Long            '4자리로 입력된 값이 저장되는 변수
7  Dim DATA As Long          'REG변수로 들어가기전 키패드 값이 임시로 저장되는 변수
8
9  Dim DATA_A As Long         '연산을 위한 임시 저장 변수
10 Dim DATA_B As Long         '연산을 위한 임시 저장 변수
11 Dim F As Single             '나눗셈 연산을 위한 소수점 변수
12 Dim F_A As Single           '나눗셈 연산을 위한 소수점 변수
13 Dim F_B As Single           '나눗셈 연산을 위한 소수점 변수
14
15 Dim SUM As Byte             '연산식을 확인 할수 있도록 나눠주는 변수
16
17 Set Display 0,0,0,50        'LCD 설정 명령어
18
19 Const Byte KEY_TABLE = (1,4,7,11,2,5,8,0,3,6,9,12,13,14,15,16) '키매트릭스 테이블
20
21 Ramclear
22
23 Cls                          'LCD 초기화
24 Delay 20                     '초기화 딜레이
25 CsrOff                       '커서 OFF
26 CLCDOUT 0,0,"KEY DATA = "  '최초 화면 표시
27
28 Do
29
30 KEY_PAD = Keypad(0)          '키매트릭스 값을 입력 받습니다.
31
32 Select Case KEY_TABLE(KEY_PAD) ' 키 값을 받아 어떤 연산을 할것인지 결정 합니다.
33     Case 13
34         SUM = 1                'SUM 변수가 연산을 결정하는 역할을 합니다.
35         DATA_A = REG
36         CLCDOUT 0,2,Dec3 DATA_A," + "

```



```

37     Do While Keypad(0) < 255
38     Loop
39     DATA = 0
40     REG = 0
41     Case 14
42     SUM = 2
43     DATA_A = REG
44     CLCDOUT 0,2,Dec3 DATA_A," - "
45     Do While Keypad(0) < 255
46     Loop
47     DATA = 0
48     REG = 0
49     Case 15
50     SUM = 3
51     DATA_A = REG
52     CLCDOUT 0,2,Dec3 DATA_A," * "
53     Do While Keypad(0) < 255
54     Loop
55     DATA = 0
56     REG = 0
57     Case 16
58     SUM = 4
59     DATA_A = REG
60     CLCDOUT 0,2,Dec3 DATA_A," / "
61     Do While Keypad(0) < 255
62     Loop
63     DATA = 0
64     REG = 0
65 End Select

66
67
68 If KEY_PAD < 255 And KEY_TABLE(KEY_PAD) < 10 Then      ' 키값을 0~9 사이의 값만 받아 레이터에 저장합니다.
69
70     KEY_DATA = KEY_TABLE(KEY_PAD) '키 매트릭스로 받은 값을 사용하기 쉽도록 테이블을 참조한다.
71     DATA = DATA << 4             '키 값을 4bit 좌측으로 쉬프트 합니다.
72     DATA = DATA + KEY_DATA      '쉬프트한 값에 새로 들어온 키값을 더합니다.
73
74     Do While Keypad(0) < 255      '키가 눌렀다가 떨어질때 까지 대기 합니다.
75     Loop

```

```

76
77 REG = Bcd2bin(DATA)          '키값을 워프드 한것을 2진수 값에서 10진수 값으로 변경 합니다.
78
79 If REG > 999 Then            '키 값이 999를 넘었을 경우 초기화를 합니다.
80     REG = 0
81     DATA = 0
82 Endif
83
84 CLCDOUT 0,0,"KEY DATA = ",DEC3 REG '키의 입력된 값을 표시 합니다.
85
86 ElseIf KEY_TABLE(KEY_PAD) = 11 Then '엔터를 눌러 최종 연산을 합니다.
87     DATA_B = REG
88     CLCDOUT 6,2,DEC3 DATA_B
89     If SUM = 1 Then CLCDOUT 9,2," = ",Dec6 DATA_A + DATA_B," "
90     If SUM = 2 Then CLCDOUT 9,2," = ",Dec6 DATA_A - DATA_B," "
91     If SUM = 3 Then CLCDOUT 9,2," = ",Dec6 DATA_A * DATA_B," "
92
93     If SUM = 4 Then          ' 나눗셈의 경우 소수점 연산을 위하여 정수를 소수점으로 변경해 줍니다.
94         F_A = DATA_A + 0.0
95         F_B = DATA_B + 0.0
96         F = F_A / F_B
97         CLCDOUT 9,2," = ",Fp(F,3,3) '소수점 표시 정수 3자리 소수점 3자리
98     Endif
99     Do While Keypad(0) < 255
100     Loop
101     DATA = 0
102     REG = 0
103
104 Endif
105 Loop

```

## 소스 설명

Select Case 명령을 이용하여 연산식의 종류를 고르도록 한 소스입니다.

키패드의 +, -, \*, / 의 종류에 따라 Select Case의 SUM변수의 값이 다르게 저장되도록 합니다.

처음 입력된 변수를 DATA\_A라는 변수에 따로 저장을 하고, 후에 입력되는 연산에 따라 다음 저장되는 DATA\_B의 변수를 연산하도록 합니다.

나눗셈의 경우 실수가 사용됨으로 SINGLE변수를 이용하여 처음부터 실수로 만들어 연산을 해야만 정상으로 데이터를 얻을 수 있습니다

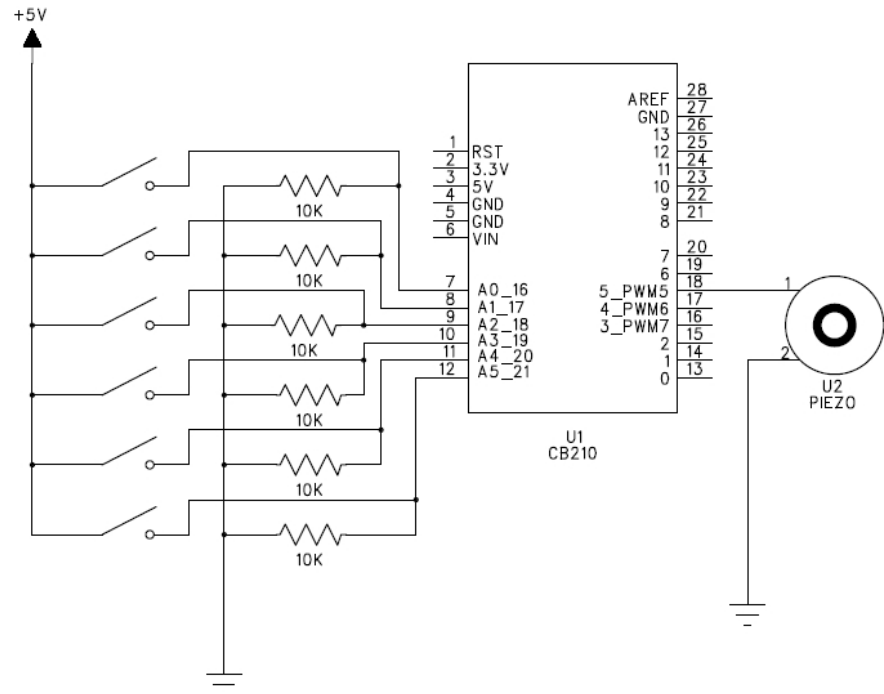
# PWM을 이용한 전자 피아노

주파수의 세기에 따라 소리를 달리 내어 줄 수 있는 PIEZO라는 부품을 이용하여 소리를 만들어 낼 수 있습니다.  
각 음계에는 고유 주파수가 있으며, 이 주파수대로만 파형을 만들어 줄 수 있다면 피아노처럼 버튼만으로 음계를 만들어 줄 수 있습니다.



피에조

## 회로도



```

1  Const Device = CB210
2
3  Dim DATA As Integer
4
5  Ramclear
6
7  Low 5
8
9  Do
10
11  If Keyin(16,100) = 1 And DATA = 0 Then
12      DATA = 1
13      Freqout 5,4403
14  ElseIf Keyin(17,100) = 1 And DATA = 0 Then
15      DATA = 1
16      Freqout 5,3923
17  ElseIf Keyin(18,100) = 1 And DATA = 0 Then
18      DATA = 1
19      Freqout 5,3495
20  ElseIf Keyin(19,100) = 1 And DATA = 0 Then
21      DATA = 1
22      Freqout 5,3299
23  ElseIf Keyin(20,100) = 1 And DATA = 0 Then
24      DATA = 1
25      Freqout 5,2939
26  ElseIf Keyin(21,100) = 1 And DATA = 0 Then
27      DATA = 1
28      Freqout 5,2618
29  Else
30      DATA = 0
31      Pwmoff 5
32  Endif
33
34  Loop

```

## 소스 설명

소스에선 PWM 명령 대신 Freqout 명령을 사용하였습니다.  
Freqout은 일정 주파수를 지속적으로 발생 시켜주는 명령으로  
듀티값이 50%인 주파수를 발생 시켜 줍니다.



### 새로운 명령어

#### Freqout 채널번호, Freq

"Freq"에 따라 발생 주파수를 결정할 수 있습니다.

"Freq"에 주파수를 직접 써주는 것은 아니고,  
어떤 공식에 의해서 계산된 값을 써주어야 합니다.

$$\text{주파수} = \frac{230400}{\text{Freq}}$$

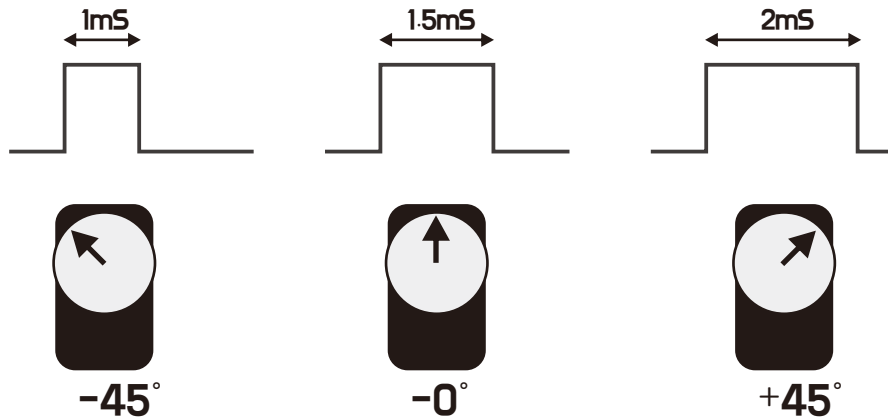
이 공식은 Freq의 값이 100 이상일 경우에만 올바른 값을 산출해냅니다.

Freqout 0, 5236 ' 440Hz의 주파수를 발생

- 위의 예제에는 한 블록만큼의 스위치를 이용하여 음계를 만들었습니다.  
하지만, 스위치를 더 늘린다면 더 많은 음계를 만들수 있습니다.
- 음계 외에도 여러 음계를 한 조건에 연속으로 넣는다면,  
여러 소리를 연속으로 출력하는 연주도 가능하게 됩니다.

# AD를 이용한 RC 서보모터 제어

- RC 서보 모터는 취미용 무선 자동차, 비행기 등에서 핸들 각도 조절을 하기 위해서 사용되는 모터를 말합니다.  
최근에는 로봇의 관절 제어에 많이 사용되는 모터입니다.
- RC 서보모터의 경우 주파수의 듀티에 따라 움직이는 각도가 달라지게 됩니다.  
이를 이용하여 AD값을 듀티에 적용하게 되면 RC 서보모터를 AD로 제어 할 수 있게 됩니다.
- RC 서보 모터에는 3가닥의 선이 있는데,  
이 중 빨간 선이 전원선, 검은 선은 GND, 노란 선은 PWM 펄스가 들어가는 선입니다.  
펄스는 대략 1초에 60회 정도의 펄스를 넣어주면 구동 할 수 있으며, 펄스의 듀티에 따라 각도가 변하게 됩니다.



- AD를 PWM에 적용하기 위해서는 우선 PWM의 듀티값의 시간을 알아야 합니다.

듀티값은 최소 1이상이어야 하며, PWM의 듀티가 1일 경우 약 0.9uS의 시간이 나오며, 이후의 1씩 증가함에 따라 0.4uS의 시간이 증가 합니다.

이를 이용해 RC 서보모터의 위치를 제어 합니다.

- RC 서보의 경우, 모터의 스펙이상의 각도의 펄스를 주게 되면 이상 동작을 하거나 모터에 무리가 되 수 있으니 AD의 값을 최소, 최대 값보다 조금 높은 값을 주어 안정적으로 동작을 할 수 있도록 해주는 것이 좋습니다.

```

1  Const Device = CB210
2
3  Dim AD_1 As Integer
4  Dim AD_OLD As Integer
5  Dim Pwm_CH As Integer
6
7  Ramclear
8
9  Low 5
10
11 Do
12     AD_OLD = AD_1           'AD값의 흔들림에 따라 PWM의 듀티값도
13     AD_1 = Tadin(0)         '값이 흔들리기 때문에 이를 막기 위해
14     AD_1 = 1750 + (AD_1 * 3.5) '한 루턴의 AD값을 이전 AD 와 비교하여
15                               '설정 한 이상의 값이 나왔을때에만 적용
16                               '하도록 합니다.
17     If Abs(AD_1 - AD_OLD) < 1 Then
18         Debug DEC6 Pwm_CH,Cr '디버깅의 내용을 보면 AD 값이 10이상 변경
19                               '될때만 PWM의 듀티 값을 적용 합니다.
20         Pwm 5,Pwm_CH,10000
21
22     ElseIf Abs(AD_1 - AD_OLD) > 10 Then
23
24         Pwm_CH = AD_1
25
26     Endif
27 Loop
  
```

- AD값의 흔들림에 따라 PWM의 값이 같이 흔들리기 때문에 이를 막기 위한 루틴이 필요 합니다.

- 한번 스캔한 값을 변수에 저장했다가 다음 스캔 값과 비교하여 일정 수치 이상 변화였을 때만, PWM값에 적용 루틴을 만들어 주어야 안정적인 동작을 할 수 있습니다.

# 스위치 입력을 이용한 릴레이 동작

큐블럭에서는 5V레벨의 입력만을 받을 수 있습니다.

하지만, 다른 제품들이 꼭 5V의 전압으로 신호를 주지는 않기 때문에 이를 받을 수 있는 회로를 만들어야 합니다.

전압 분배를 통하여 전압을 나눈다면 쉽게 전압을 5V를 만들어 줄 수 있지만, 노이즈나 채터링까지 잡아 줄 수는 없습니다.

이를 막기 위한 것이 포토커플러라는 IC 입니다. 이 포토커플러를 통하여 입력을 받아 릴레이를 동작 시키는 회로를 만들어 보겠습니다.

## 채터링

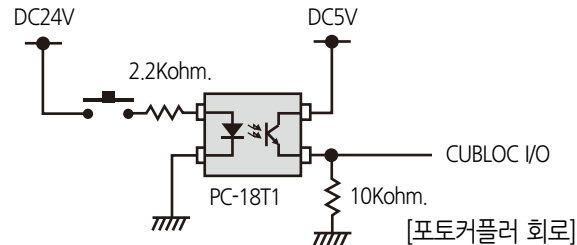


[채터링 현상]

### 채터링이란?

채터링은 스위치 입력이나 외부 입력을 받을 경우, 입력 펄스가 안정화가 되기 전까지 펄스의 흔들림으로 인한 필요없는 입력이 인식되는 현상을 말합니다.

## 포토커플러



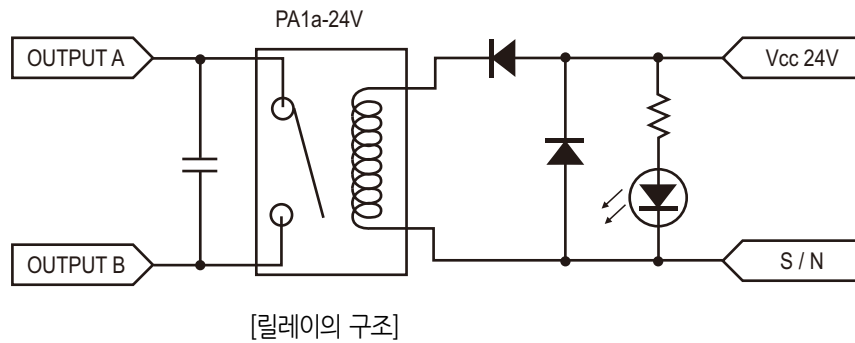
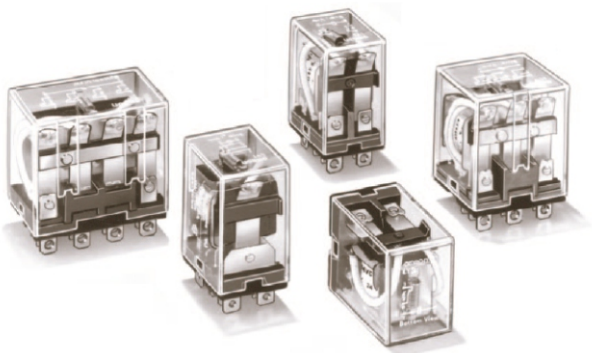
[포토커플러 회로]

### 포토커플러란?

외부 입력을 받을 경우 생기는 노이즈 및 채터링을 방지하기 위해 만들어진 IC로, 광 LED와 TR의 구조로 되어 있어 입력 전압을 스위칭 할 경우에도 많이 쓰입니다.

## 릴레이란?

릴레이는 점점 출력을 할 수 있는 부품으로,  
전압이 직접적으로 출력이 되는 것이 아니라 코일의 점점을 이용하여 연결해 주는 역할을 합니다.  
이는 AC,DC 모두 사용이 가능하며 일반적인 전압 보다 많은 양의 전압을 흘려 줄 수 있습니다.





## 소스

소스는 일반 입력을 받아 출력을 하는 소스와 같습니다.

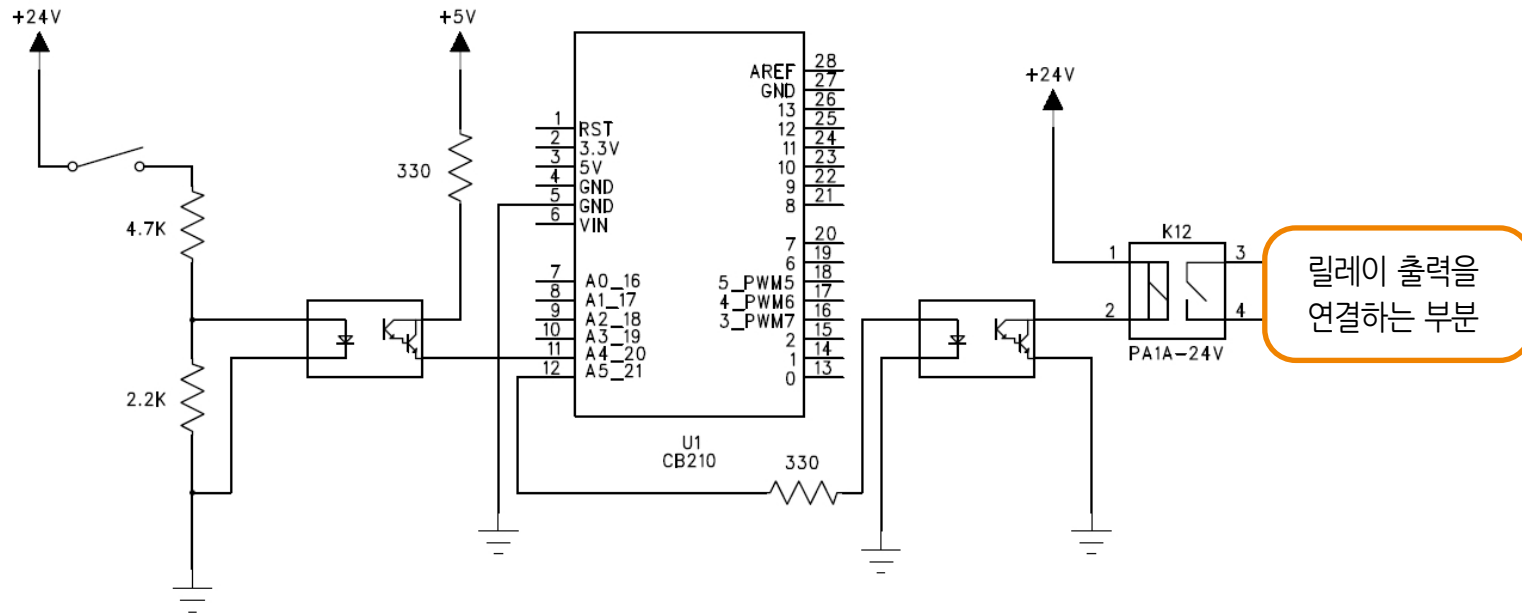
```

1  Const Device =CB210
2
3  Ramclear
4
5
6  Do
7
8  If In(21) = 1 Then
9      Out 20,1
10 Else
11     Out 20,0
12 Endif
13
14
15 Loop

```

- 포트 21번의 입력을 받아 포트 20의 출력을 ON 합니다.  
포트의 입력이 OFF되면 출력포트도 같이 OFF 됩니다.

# 회로도



스위치 입력을 받게 되면 릴레이가 동작하게 됩니다.

릴레이는 동작을 하게 되면 “딸깍”하는 접점이 붙는 소리가 나게 됩니다.

실제 동작을 확인 하려면, 릴레이 출력 단에 전압을 흘려주면 ON시 전압이 흐르게 되고, OFF시 단선 되는 것을 볼 수 있습니다.

# 스텝모터 컨트롤

스텝 모터란 DC 모터와는 다르게 위치 제어를 할 수 있는 모터입니다.  
 펄스의 수량을 체크하여 움직이고 싶은 만큼의 회전을 할 수 있습니다.  
 스텝모터는 지속적인 회전이 아닌 제한적인 움직임을 필요로 하는 곳에 사용 됩니다.



스텝모터를 컨트롤 하기 위해서는 펄스의 세기와 카운터를 동시에 해야 하며, CB210 제품의 PWM기능을 이용하여 사용이 가능합니다.  
 큐볼렉에서 제공되는 스텝펄스 컨트롤 명령을 이용하여 속도 및 펄스의 개수를 설정할 수 있습니다.

■ Steppulse 0,5,300,50000

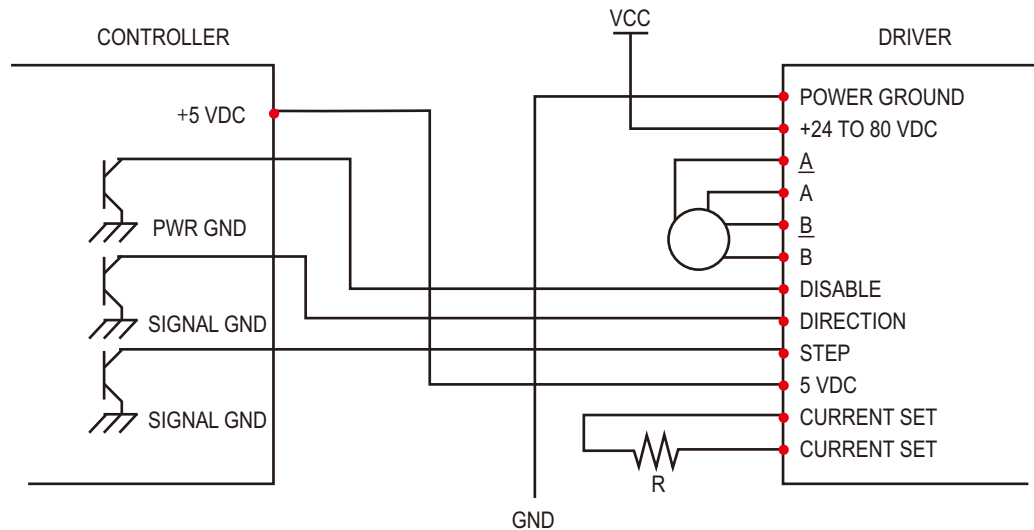
0 = 스텝펄스 채널 (CB210에서는 0CH만 사용 가능합니다.)

5 = 스텝펄스를 출력 할 포트 번호 입니다.

300 = 펄스의 속도 300HZ의 펄스를 출력 합니다.(최대 15000까지 가능)

50000 = 펄스의 수량 (최대 2147483647까지 가능)

스텝모터를 구동하기 위해서는 스텝모터 드라이버를 사용하여 하는데, 보통 스텝모터 드라이버의 구성은 아래와 같습니다.  
 큐블록에서 연결되는 선은 3선으로,  
 펄스가 연결되는 STEP포트와 제품의 상태를 on해주는 DISABLE, 방향을 정해 주는 DIRECTION을 연결해주면 동작이 가능 합니다.



[스텝모터 드라이버 구성]

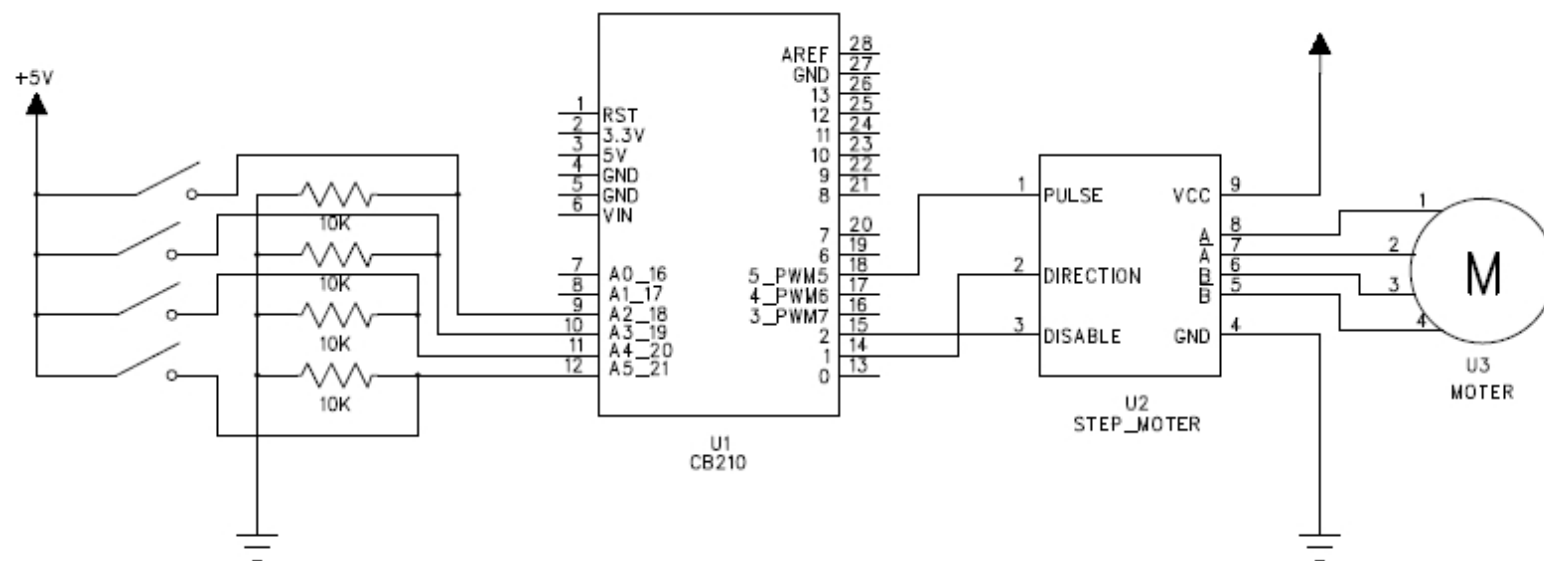
## 소스

```

1  Const Device =CB210
2
3  #define step_dir 1
4  #define step_en 2
5
6  Ramclear
7
8  Set Display 0,0,0,100
9
10 Out step_dir,1
11 Out step_en,1
12
13 Cls
14 Delay 20
15 Csoff
16
17 CLCDOUT 0,0,"STEP TEST!!"
18
19 Do
20
21     If In(21) = 1 Then Steppulse 0,5,200,50000 '시작 버튼을 누르면 200Hz의 속도로 5만개의 펄스출력
22     If In(20) = 1 Then Out step_dir,0         ' 방향을 변경 할수 있는 S/W
23     If In(19) = 1 Then Out step_dir,1
24     If In(18) = 1 Then Stepstop 0              '정지 S/W
25
26     CLCDOUT 0,1,"PULSE = ",DEC5 Stepstat(0)    '현재 펄스의 수량을 다운카운터 합니다.
27     CLCDOUT 0,2,"MOTER STAT = ",DEC2 Outstat(step_dir) '현재 회전 방향을 표시 합니다.
28
29     Delay 100
30 Loop

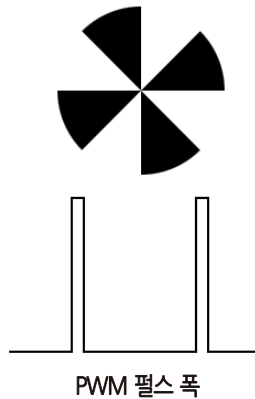
```

# 회로도



# DC 팬모터를 이용한 선풍기

앞의 스텝 모터와는 달리, DC 모터의 경우 해당 펄스를 입력하면 빠르기가 변경되는 모터입니다.  
주로 선풍기의 모터나 자동차 모터 등에 사용 됩니다.



PWM펄스 폭에 따라  
모터의 회전 속도가  
조정 됩니다.

```

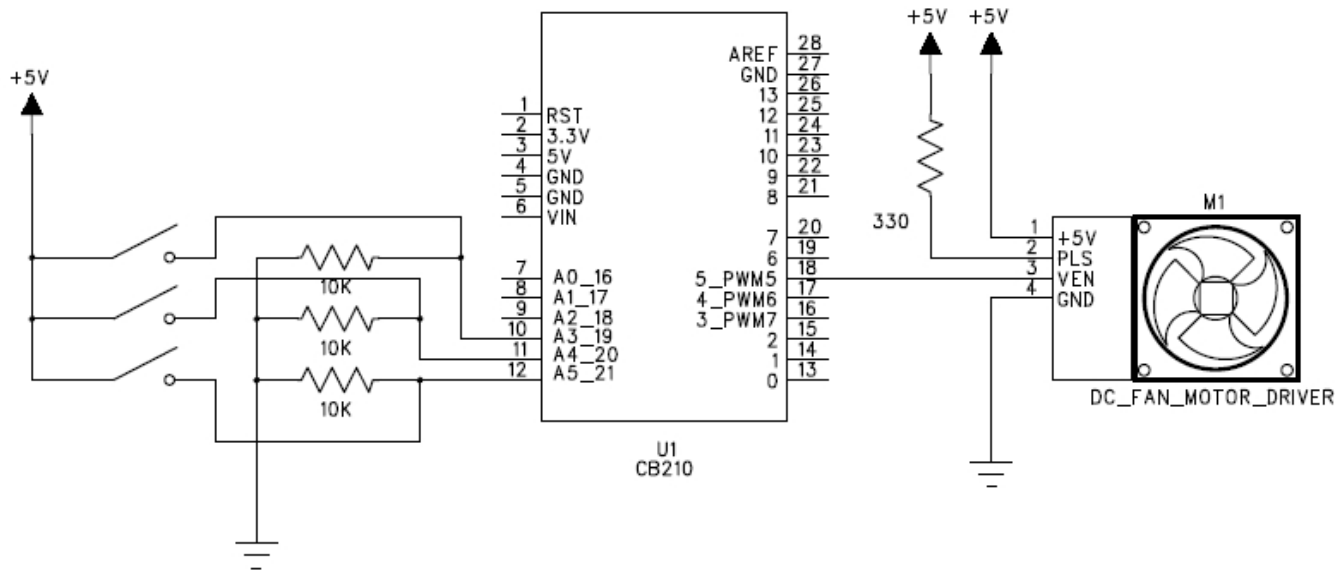
1  Const Device =CB210
2
3  #define FAN_VEN 5
4
5  Dim I As Integer
6
7  Ramclear
8
9  Set Display 0,0,0,100
10
11 Cls
12  Delay 20
13  CsrOff
14
15  CLCDOUT 0,0,"DC FAN TEST!!"
16
17  Low FAN_VEN
18
19  CLCDOUT 0,2,"MOTER SPEED = STOP  "
20  Do
21
22      I = Adin(5)
23      If In(21) = 1 Then          '버튼의 입력에 따라 회전 속도가
24                                  '달라지게 됩니다.
25          Pwm FAN_VEN,700,1024    '초기 모터 스타트시 최소 속도를 내줌
26          Delay 500
27          I = 400
28          Pwm FAN_VEN,I,1024
29          CLCDOUT 0,2,"MOTER SPEED = Low  "
30      ElseIf In(20) = 1 Then
31          I = 800
32          Pwm FAN_VEN,I,1024
33          CLCDOUT 0,2,"MOTER SPEED = Medium"
34      ElseIf In(19) = 1 Then
35          I = 1023
36          Pwm FAN_VEN,I,1024
37          CLCDOUT 0,2,"MOTER SPEED = High  "
38      ElseIf In(18) = 1 Then      '정지 버튼
39          PwmOff 5
40          CLCDOUT 0,2,"MOTER SPEED = STOP  "
41      Endif
42
43
44      Delay 100
45  Loop

```



- PWM을 이용한 DC모터의 속도 제어 소스로 PWM의 명령어 경우, 연속적인 명령어 실행시 오작동을 할 수 있습니다.  
이를 방지하기 위해 정해진 펄스 값을 지정하여 해당 스위치를 눌렀을 때 적용 되도록 하였습니다.
- 처음 시작되는 펄스의 경우, 모터의 최초 동작시 필요한 전압의 양이 부족하기 때문에,  
처음 시작시 최소한의 동작 전압량을 주기 위해 높은 펄스를 주고, 안정 동작 후 펄스의 세기를 줄여 주는 방식을 사용해야 합니다.

## 회로도



[THE END]