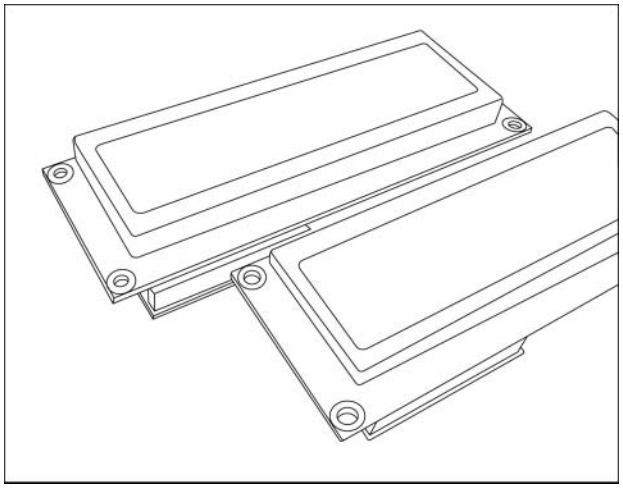
향상된 시리얼 영문 LCD 디스플레이 모듈

ALCD Series

Enhanced Serial English LCD Module



1. 제품의 개요

본 제품은, 시리얼 방식으로 LCD 를 제어할 수 있는, “시리얼 영문 LCD 모듈”입니다.

기존의 LCD 모듈은 많은 컨트롤라인이 필요하고, 복잡한 제어 방식을 이해해야만 사용할 수 있는 “병렬 접속”방식이었으나, 본 제품은 1~2선 만으로 시리얼 방식의 데이터를 전송하여, LCD상에 글자를 표시할 수 있는 방식으로, LCD모듈에 대한 구체적인 이해없이 사용자가 손쉽게 사용할 수 있게 만든 제품입니다.

2. 제품의 특징

- 16 By 2 라인부터, 40 By 4 라인까지의 다양한 사이즈.
- PICBASIC / 일반 마이컴 (PC 포함) 3선 / 4선의 선택적 인터페이스. (최적의 연결 선택)
- 5V레벨/+-15V레벨의 RS232를 모두 사용 가능. (포트에 따라 선택적으로 자동 적용)
- 자체에 RS232 드라이버 장착.
- 8종류의 보레이트(전송률) 지원.
- Insert/OverWrite 편집 모드 지원.
- 방향키,Space, Enter, Del, Home, End, BackSpace 키 작동 지원.
- 백라이트(**-BL 기종)를 프로그램적으로 컨트롤 가능.
- 56 BYTE 의 “링버퍼”를 내장.
- **마이컴 및 PC 등에서 고속으로 시간 지연 없이 컨트롤 할 수 있음.**
- 위치 조정, 커서 관리, 화면 클리어 등, 수많은 기능 내장. (제어 코드로 쉽게 컨트롤)
- ALCD 에뮬레이터 프로그램제공.(홈페이지 참고)
- **다양한 전송 모드 지원.(일반 명령SET, ESC 코드 SET, BYTE 단위 문자 전송)**

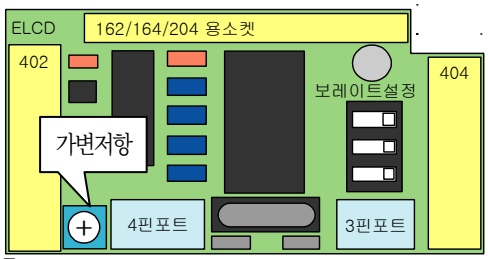
- 제품별 소비전류

제품명	소비전류
ALCD162-BL	0.10A
ALCD164-BL	0.09A
ALCD204-BL	0.29A
ALCD402-BL	0.12A
ALCD404-BL	0.14A
ALCD164-BLUE	0.09A
ALCD204-BLUE	0.12A

3. 제품의 종류

모 델 명	표시 가능 수	백라이트 방법
ALCD-162-BL	16 BY 2	LED
ALCD-164-BL	16 BY 4	LED
ALCD-204-BL	20 BY 4	LED
ALCD-402-BL	40 BY 2	LED
ALCD-404-BL	40 BY 4	LED
ALCD-164-BLUE	16 BY 4	CCFL
ALCD-204-BLUE	20 BY 4	CCFL

4. ALCD 구동모듈 각 부의 설명



▌LCD 소켓

해당되는 규격의 LCD를 연결함.(생산 시 해당 소켓에 결합되어 있음)

▌보레이트 설정

8종류의 보레이트 (전송 속도)를 설정할 수 있음.

BAUD	설정값 (DIP 설정)
2400	0
4800	1
9600	2
14400	3
19200	4
38400	5
57600	6
115200	7

▌가변저항

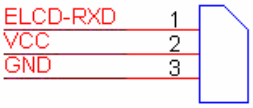
LCD 구동 시 글자가 흐리거나 잘 보이지 않을 경우 가변 저항을 조정하시기 바랍니다.

▌3핀 콘넥터

PICBASIC이 가지고 있는 PICBUS를 사용하여, PICBASIC에서 간단히 컨트롤 하고자 할 때 사용함. 일반 마이컴에서도 연결할 수 있으나, 이 경우 "READY CODE(0x88)"를 받을 수 없음.

PICBASIC의 PICBUS 사용시 19200BPS로 전송률 설정해 주어야 함.

RXD, VCC, GND로 구성되어 있고, 5V 레벨의 RS232를 사용하게 됨.



3핀 콘넥터

▌4핀 콘넥터

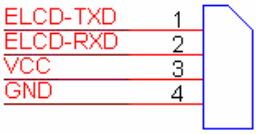
마이컴 및 PC 에서 직접 컨트롤 하고자 할 때 사용.

이때는 마이컴에서는 MAX232를 사용해서 전송해 주어야 함.(레벨을 맞추어 주기 위한 것임)

+-15V정도의 RS232 본래의 규격을 직접 사용할 수 있음.

내부적으로 MAX232 라고 하는 RS232 드라이버와 연결되어 있음.

TXD, RXD, VCC, GND의 4선으로 구성되어 있고, TXD로는 "READY CODE(0x88)"이 제어 코드 수행시 마다 전송됨.



4핀 콘넥터

5. ALCD 제어 코드 표

▌기본 명령

처리가 완료되면, "READY CODE"0x88(다음 처리 준비 됐음)을 ALCD 컨트롤러가 전송함.

(정확한 포맷의 명령을 전송한 경우)

제어 코드	전송 BYTE 수	작동 내용
A0	1	LCD 를 완전히 초기화. 약 50ms(0.05 초) 정도의 시간이 소요됨. -커서보이기. -커서 빰박이지 않음. -0,0 위치에 커서 보이기. -OVERWRITE MODE 로 설정. -백라이트 켜기. 의 과정을 수행함.
A1 X Y	3	표시위치를 지정. X 는 열, Y 는 행을 말함. ALCD 종류에 따라 표시 범위가 다름.
		지정된 위치에 커서가 위치하게 됨.
A2 문자열 00	Free	문자열을 출력. 문자열 다음의 0x00 코드는 문자열 "끝" 임을 알리는 NULL 코드. (C 언어 스타일 사용) 문자열은 전달 되는대로 바로 바로 출력됨. CR LF 코드가 문자열 사이에 끼어 있는 경우, 이도 자체적으로 자동 처리됨. 수십 바이트의 문자열을 일괄 처리하는 경우, 버퍼 용량(56 바이트)를 초과하는 문자열의 경우는 INSERT MODE 에서 사용하지 않기 바람.(INSERT MODE 는 처리 속도가 느리므로 OverFlow 될 수 있음.) 버퍼 용량 이상의 다량 문자열 처리의 경우는, 반드시, OVERWRITE MODE 로 설정 후, 문자열을 출력하기 바람. 버퍼 용량을 초과하지 않는 문자열의 경우는, 일단 버퍼에 일괄 저장된 후 처리되므로, INSERT MODE 를 사용해도 정확하게 처리됨. 키 입력 처럼 문자를 하나씩 처리하는 경우에도, 반드시 포맷을 갖추어서 보내야 함. (ALCD 컨트롤러 내부적으로, 이 0x00 을 읽어 들여야만, 모든 처리가 완료된 것으로 간주하고, 0x88 코드를 전송함.) (예) A2 문자 00 A2 문자열 00 56BYTE 의 버퍼 크기보다 작은 길이의 문자열을 일괄 출력하는 경우에는, 115200BPS 를 사용하는 초고속 마이컴 등에서도 Delay(시간지연) 없이 일괄전송이 가능하지만, 56BYTE 크기를 넘는 다량의 문자열을 일괄 출력하는 경우에는, 버퍼의 내용을 ALCD 가 실시간으로 실행하며 계속 버퍼를 비워줄 수 있도록, 약간의 여유 시간을 BYTE 전송 사이사이에 넣어 주면 좋습니다. PICBASIC 유저는 아무 걱정 없이 그냥 제한 없이 문자열을 전송해 주어도 상관없습니다.(BYTE 간 DELAY 는 전혀 필요 없음)
A3 01	2	LCD 화면 지우기. 표시위치는 0,0 으로 설정됨.
A3 0C	2	커서 감추기.
A3 0E	2	커서 보이기. ALCD 기본설정.
A3 0A	2	커서 깜박임 정지. ALCD 기본설정.
A3 0B	2	커서 깜박임.
A4 CODE 8BYTE	10	CGRAM 에 사용자정의 비트맵전송-저장. CODE 는 저장되는 비트맵의 호출 번호,

		1~8 까지 지정할 수 있음.(다른 코드 사용시 에러) 8BYTE 는 5*8 로 지정된 비트맵을 의미하며, 8Byte 의 길이를 가짐. 저장된 비트맵은 저장할 때 사용한 CODE 번호를 사용하여 호출할 수 있음. 예)A2 문자열... CODE 문자열... 00 로 문자열 전송 시, 문자열 사이에 CODE 를 집어 넣어 주면, 문자열 사이에 표시됨.
A5 01	2	백라이트 켜기.
A5 02	2	백라이트 끄기.
A6 08	2	BackSpace 키 구현. 커서가 위치해 있는 현재 라인만 적용됨.
A6 0D	2	Enter 키 구현. CR LF 코드 구현.
A6 23	2	End 키 구현. 현재 라인의 제일 끝 칸으로 이동.
A6 24	2	Home 키 구현. 현재 라인의 제일 첫째 칸으로 이동.
A6 25	2	Left 키 구현. 커서를 한칸 왼쪽으로 이동.
A6 26	2	Up 키 구현. 커서를 현재 라인 한줄 위로 이동.
A6 27	2	Right 키 구현. 커서를 한칸 오른쪽으로 이동.
A6 28	2	Down 키 구현. 커서를 현재 라인 한줄 아래로 이동.
A6 2E	2	Del 키 구현. 커서가 위치해 있는 현재 라인만 적용됨.
A6 2D	2	편집모드를 INSERT MODE 로 설정. Insert 모드는 문자 하나를 처리하는데 많은 시간이 소요되므로, 일괄적인 문자열 출력에서는, OverWrite 모드로 전환 후 사용할 것. 주로 문자를 키보드 입력방식으로 처리하는 경우 사용하면 좋음.
A6 00	2	편집모드를 OVERWRITE MODE 로 설정. 초기화 후에 기본으로 설정되어 있음. 일괄적인 문자열 출력에 사용할 것을 권장함. 문자 처리 속도가 대단히 고속으로 처리되고, 115200 BPS 에서도 Delay 없이, 무한의 문자열을 버퍼의 OverFlow 걱정 없이 사용할 수 있음.(단, 56BYTE 버퍼 용량을 초과 하지 않는 일괄 전송의 경우)
A7 01 HB LB	4	전달된 16BIT 값을 그대로 16 진수 문자로 LCD 에 표시. 현재 커서가 위치한 곳부터 표시됨. HB : 상위바이트 LB : 하위바이트 무조건 4 자리로 표시하고, 빈 자리는 0 으로 채워서 표시됨.
A7 02 HB LB	4	전달된 16BIT 값을 그대로 10 진수 문자로 LCD 에 표시. 현재 커서가 위치한 곳부터 표시됨. HB : 상위바이트 LB : 하위바이트 무조건 5 자리로 표시하고, 빈 자리는 0 으로 채워서 표시됨.

▌ESC 코드 확장 명령

ESC 코드 명령이 필요한 이유는, VISUAL BASIC 같은 BASIC 계열의 언어에서는 시리얼 포트로 80h ~ FFh 사이의 값을 전송하기 어렵기 때문입니다.

이로 인하여, ESC 라는 제어코드를 사용하고, 모든 전송 값은 00h ~ 7Fh 사이의 값만으로도 제어를 수행할 수 있도록 명령포맷을 확장한 것입니다.

ESC 는 10진수로는 27을 16진수로는 1Bh 의 값을 갖습니다.

'A' 는 문자 A 를 의미합니다.

처리가 완료되면, "READY CODE" 0x88(다음 처리 준비 됐음)을 ALCD 컨트롤러가 전송함.

(정확한 포맷의 명령을 전송한 경우)

제어 코드	전송 BYTE 수	작동 내용
ESC 'A'	2	[A0] 코드와 동일.
ESC 'B' X Y	4	[A1 X Y] 코드와 동일.
ESC 'C' 문자열 00	Free	[A2 문자열 00] 코드와 동일.
ESC 'D' 01	3	[A3 01] 코드와 동일.
ESC 'D' 0C	3	[A3 0C] 코드와 동일.
ESC 'D' 0E	3	[A3 0E] 코드와 동일.
ESC 'D' 0A	3	[A3 0A] 코드와 동일.
ESC 'D' 0B	3	[A3 0B] 코드와 동일.
ESC 'E' CODE 8BYTE	11	[A4 CODE 8BYTE] 코드와 동일.
ESC 'F' 01	3	[A5 01] 코드와 동일.
ESC 'F' 02	3	[A5 02] 코드와 동일.

ESC ‘ G ’ 08	3	[A6 08] 코드와 동일.
ESC ‘ G ’ 0D	3	[A6 0D] 코드와 동일.
ESC ‘ G ’ 23	3	[A6 23] 코드와 동일.
ESC ‘ G ’ 24	3	[A6 24] 코드와 동일.
ESC ‘ G ’ 25	3	[A6 25] 코드와 동일.
ESC ‘ G ’ 26	3	[A6 26] 코드와 동일.
ESC ‘ G ’ 27	3	[A6 27] 코드와 동일.
ESC ‘ G ’ 28	3	[A6 28] 코드와 동일.
ESC ‘ G ’ 2E	3	[A6 2E] 코드와 동일.
ESC ‘ G ’ 2D	3	[A6 2D] 코드와 동일.
ESC ‘ G ’ 00	3	[A6 00] 코드와 동일.
		[A7 01 HB LB] 와 대응되는 코드 없음. VISUAL BASIC 계통에서는 00h ~ FFh 를 전부 사용할 수 없으므로.
		[A7 02 HB LB] 와 대응되는 코드 없음. VISUAL BASIC 계통에서는 00h ~ FFh 를 전부 사용할 수 없으므로.

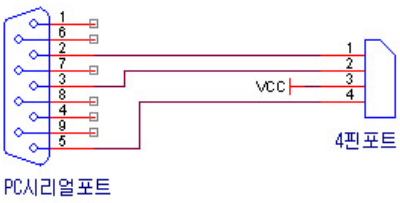
▌BYTE 단위 직접 출력

이 기능은, 단순한 문자 출력의 경우, 제어 코드 없이 BYTE 단위의 한 개 문자를 ALCD 상에 바로 바로 표시할 수 있습니다.
표시되는 문자의 코드 범위는, 20h ~ 7Eh 까지 입니다.
ALCD 안의 문자가 윈도우상의 문자와 다를 수 있습니다.(LCD의 자체 특성 때문임)
처리가 완료되어도 제어 코드를 수행한 것이 아니므로, “READY CODE ”0x88(다음 처리 준비 됐음)을 전송하지 않습니다.

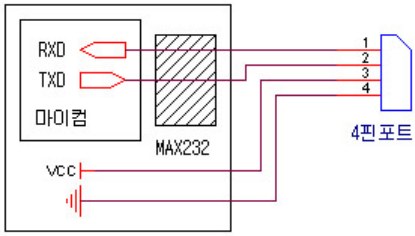
전송	전송 BYTE 수	작동 내용
1BYTE 문자 코드	1	코드에 해당하는 문자가 현재 캐럿(커서)의 위치에 바로 출력됨.

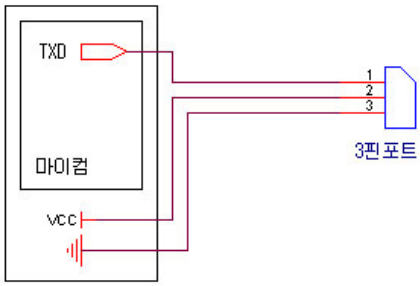
6. 다양한 연결 예

▌PC에 연결 할 경우

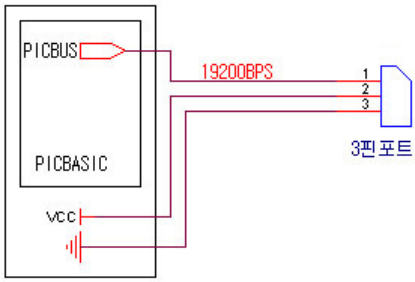


▌마이컴에 연결 할 경우



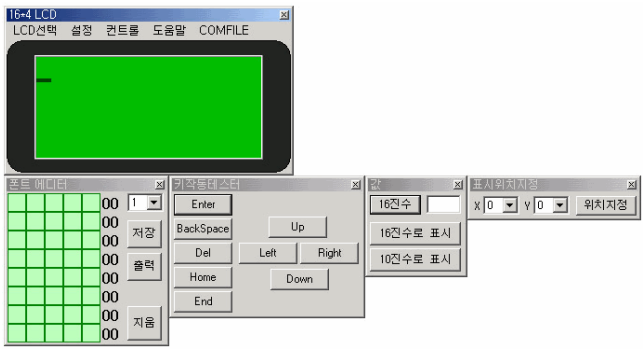


▌PICBASIC에 연결 할 경우



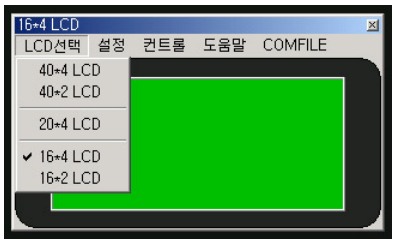
7. ALCD PC용 에뮬레이터 사용 방법

PC와 ALCD를 직접 연결한 뒤 본 프로그램을 이용하여 ALCD의 각종 기능을 테스트해볼 수 있습니다.



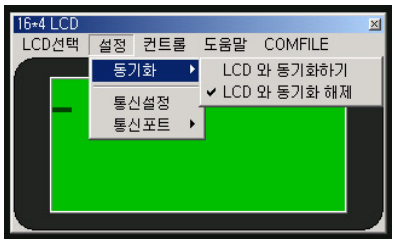
LCD 테스터는 아주 간단한 ALCD 테스용 프로그램입니다. ALCD 테스터로서의 기능 이외에, ALCD의 기능을 쉽게 학습할 수 있는 기능도 가지고 있습니다. 사용하실 때는 PC와 연결하기 위한 통신 케이블이 필요합니다. 통신 케이블은 LCD 전용의 통신 케이블을 별도 판매하고 있으니 이를 사용하시면 됩니다. 테스트 전에 ALCD를 PC 와 케이블로 연결한 상태에서 제어를 수행하고자 할 때는, 먼저 " 설정 -> 통신 포트 -> 열기" 수행해서, 통신 포트를 열어야 합니다.

LCD 선택 메뉴



에뮬레이션 할 LCD 를 선택 합니다. 현재 생산되는 모든 기종을 설정할 수 있습니다. 케이블로 연결된 LCD 와 같은 종류를 설정해주면 됩니다.

설정 메뉴



통신설정(BAUD-통신속도)을 수행하거나, COM 포트 선택을 할 수 있고, 통신 포트를 열거나 닫을 수 있습니다. 동기화는 에뮬레이션용 LCD 화면과 LCD 의 자동을 완전히 동기화하는데 사용됩니다. 키 작동은 물론이고, 캐럿의 위치, 표시되는 문자열 등도 완전히 동일하게 표시됩니다.

컨트롤 메뉴

LCD 를 제어하는데 사용되는 제어코드를 모두 실행해 볼 수 있습니다. 문자열 출력(Print)의 경우는 현재 에뮬레이션용 LCD 화면에 표시된 문자열을 전부 읽어서 ELCD 로 출력해 줍니다. LCD 초기화, LCD 화면 지우기 의 두 가지 메뉴는, 동기화가 설정된 경우, 에뮬레이션용 LCD 도 완전히 초기화 되며, 화면도 완전히 지워집니다. 편집모드 설정은 PC 에디터와 같은 Insert/Overwrite Mode 의 전환을 가능하게 만들어 줍니다.

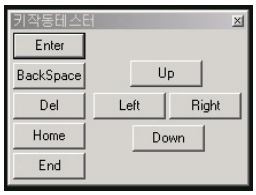
폰트 에디터

LCD 의 CGRAM 에 사용자가 만든 BITMAP 을 저장하고, 출력할 수 있습니다. CODE 는 1~8 까지 8 개를 사용합니다. 지움 버튼은 현재 창을 초기화 합니다. 마우스로 점을 찍으면, 검은색으로 표시된 곳은 논리 1, 다른 부분은 논리 0 이되고, 옆에 현재 설정 값이 그대로 표시됩니다. LCD 가 연결 되있지 않아도, 저장 버튼을 누르면, 개발 환경 내부적으로 비트맵을 저장해 주며, 콤보 박스 코드를 선택하면, 저장된 비트맵이 다시 보여집니다.

표시 위치 지정

Location 제어코드를 수행합니다. 콤보박스에서 위치를 설정후, 위치지정버튼을 누르면, 해당 포맷의 명령이 LCD 로 전송되어, 지정된 위치에 캐럿이 위치하게 됩니다.

키 작동 테스터



제어코드 중 키 작동 관련 코드를 수행합니다. Location 을 하는 것이 아니라, 실제 그대로 키 코드 제어를 수행합니다. 키 작동은 ELCD 의 특성상, PC 에디터의 작동과 완전히 동일하지는 않습니다. LCD 의 특성에 맞추었으므로, 주의 하시기 바랍니다.

값 -> 문자열

Edit 에 사용자가 입력한 값을 16 진수나, 10 진수 포맷으로 ALCD 에 표시하게 합니다. "16 진수로 표시" 버튼을 누르면, 입력된 값을 ALCD 가 16 진수 문자로 표시해 주고, "10 진수로 표시" 버튼을 누르면, 입력된 값을 ALCD 가 10 진수 문자로 표시해줍니다. 실제로, 제어코드와 16BIT 값을 전송해주며, 문자열 처리는 ALCD 내부적으로 처리됩니다.

PC통신 케이블

ALCD 와 PC를 연결할 수 있는 전용의 케이블을 별도 판매하고 있습니다. 이 통신 케이블이 있어야, 이 프로그램을 사용하여 ALCD를 컨트롤 할 수 있게 됩니다. http://www.comfile.co.kr 을 참고 하시기 바랍니다.

8. 영문 ALCD 모듈을 사용하기 위한 예제 소스

▌PICBASIC에서 사용 할 경우

```
'================================================
'         Advanced LCD 버전 테스트 프로그램
'================================================

  DIM I AS BYTE

  SET PICBUS HIGH   ' HIGH는 19200, LOW는 4800으로 설정됨
  LCDINIT
  CSROFF

' BUSOUT &HA0    'LCD 초기화 (LCD를 클리어 시키는 명령 포함)
' BUSOUT &HA1;&H00;&H00    ' 위치 표시
' BUSOUT &HA2;&H30;&H00    ' 문자열 표시
' BUSOUT &HA3;&H01         ' LCD 화면 지우기
' BUSOUT &HA3;&H0C         ' 커서 감추기
' BUSOUT &HA3;&H0E         ' 커서 보이기
' BUSOUT &HA3;&H0A         ' 커서 깜박임 정지
' BUSOUT &HA3;&H0B         ' 커서 깜박임
' BUSOUT &HA5;&H01         ' 백라이트 켜기
' BUSOUT &HA5;&H02         ' 백라이트 끄기
' BUSOUT &HA6;&H08         ' 백스페이스
' BUSOUT &HA6;&H0D         ' 엔터키
' BUSOUT &HA6;&H23         ' END 키 구현
' BUSOUT &HA6;&H24         ' HOME 키 구현
' BUSOUT &HA6;&H25         ' LEFT 키 구현
' BUSOUT &HA6;&H26         ' UP 키 구현
' BUSOUT &HA6;&H27         ' RIGHT 키 구현
' BUSOUT &HA6;&H28         ' DOWN 키 구현
' BUSOUT &HA6;&H2E         ' DEL 키 구현
' BUSOUT &HA6;&H2D         ' 편집 모드를 인서트 모드로 설정
' BUSOUT &HA6;&H00      ' 편집 모드를 오버라이트 모드로 설정
' BUSOUT &HA7;&H01;상위;하위    ' 전달된 16비트 값을 그대로
16진수로 표시
' BUSOUT &HA7;&H02;상위;하위    ' 전달된 16비트 값을 그대로
10진수로 표시

   I = 255

' LOCATE 0,0
' PRINT "TEST ELCD "

  LOCATE 0,1
  PRINT DEC(I,3,0)
```

▌PIC 마이컴에서 C로 사용 할 경우

BAUD	설정값(DIP 설정)
2400	0
4800	1
9600	2
14400	3
19200	4
38400	5
57600	6
115200	7

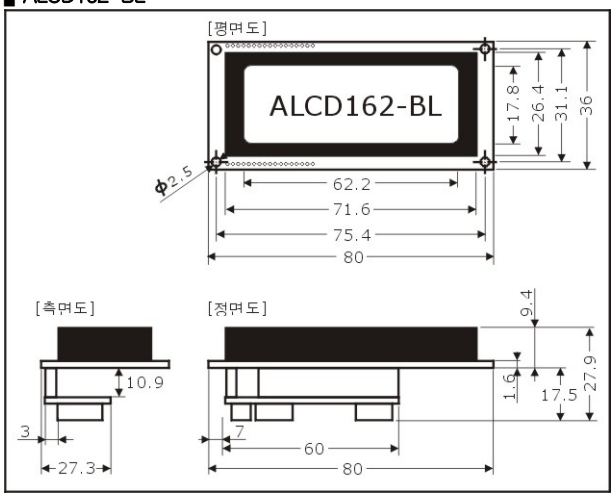
```
/*
lcd test2
*/
#include<16F876.h>
#use delay(clock=4000000)
#use rs232(baud=19200,xmit=pin_B0,rcv=pin_B1)

void main()
{
   delay_ms(200);  // LCD 초기화를 위한 딜레이 타임

   printf("%c%c",0xa3,0x1);
// printf(“%c”,0xA0);    ' LCD 초기화
                          (LCD를 클리어 시키는 명령 포함)
// printf(“%c%c%c”,0xA1,0,0);     ' 위치 표시
// printf(“%c%c%c”,0xA2,0x30,0);  ' 문자열 표시
// printf(“%c%c”,0xA3,0x01);      ' LCD 화면 지우기
// printf(“%c%c”,0xA3,0x0C);      ' 커서 감추기
// printf(“%c%c”,0xA3,0x0E);      ' 커서 보이기
// printf(“%c%c”,0xA3,0x0A);      ' 커서 깜박임 정지
// printf(“%c%c”,0xA3,0x0B);      ' 커서 깜박임
// printf(“%c%c”,0xA5,0x01);      ' 백라이트 켜기
// printf(“%c%c”,0xA5,0x02);      ' 백라이트 끄기
// printf(“%c%c”,0xA6,0x08);      ' 백스페이스
// printf(“%c%c”,0xA6,0x0D);      ' 엔터키
// printf(“%c%c”,0xA6,0x23);      ' END 키 구현
// printf(“%c%c”,0xA6,0x24);      ' HOME 키 구현
// printf(“%c%c”,0xA6,0x25);      ' LEFT 키 구현
// printf(“%c%c”,0xA6,0x26);      ' UP 키 구현
// printf(“%c%c”,0xA6,0x27);      ' RIGHT 키 구현
// printf(“%c%c”,0xA6,0x28);      ' DOWN 키 구현
// printf(“%c%c”,0xA6,0x2E);      ' DEL 키 구현
// printf(“%c%c”,0xA6,0x2D);      ' 편집 모드를 인서트 모
드로 설정
// printf(“%c%c”,0xA6,0);         '편집 모드를 오버라이트
모드로 설정
// printf(“%c%c”,0xA7,0x01,상위,하위);   ' 전달된 16비트
값을 그대로 16진수로 표시
// printf(“%c%c”,0xA7,0x02,상위,하위);   ' 전달된 16비트
값을 그대로 10진수로 표시

   while(1){
            printf("%c%c%c",0xa1,0,0);
            printf("%cTEST ALCD%c",0xa2,0);
          }
}
```

VISUAL BASIC 으로 사용 할 경우

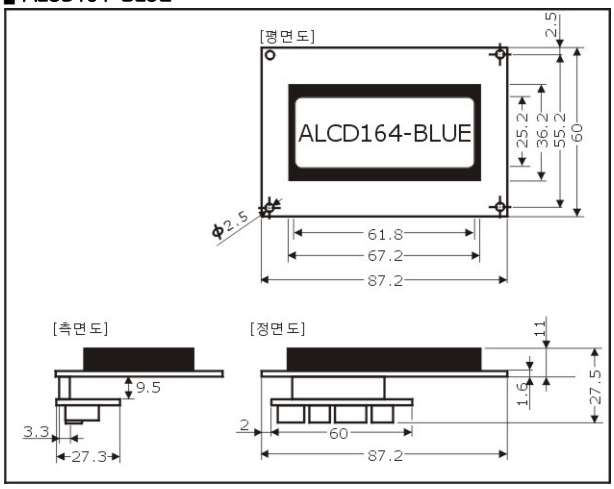
```
Private Sub Command10_Click()
' 엔터키
    Dim TX As Variant
    TX = Chr(27) + "G" + Chr(&HD)
    MSComm1.Output = TX
End Sub
Private Sub Command11_Click()
' END KEY
    Dim TX As Variant
    TX = Chr(27) + "G" + Chr(&H23)
    MSComm1.Output = TX
End Sub
Private Sub Command12_Click()
' HOME KEY
    Dim TX As Variant
    TX = Chr(27) + "G" + Chr(&H24)
    MSComm1.Output = TX
End Sub
Private Sub Command13_Click()
' 삽입 모드
    Dim TX As Variant
    TX = Chr(27) + "G" + Chr(&H2D)
    MSComm1.Output = TX
End Sub
Private Sub Command14_Click()
' 문자 지우기
    Dim TX As Variant
    TX = Chr(27) + "G" + Chr(&H2E)
    MSComm1.Output = TX
End Sub
Private Sub Command15_Click()
' 덮어쓰기 모드
    Dim TX As Variant
    TX = Chr(27) + "G" + Chr(0)
    MSComm1.Output = TX
End Sub
Private Sub Command16_Click()
' 화면 지우기
    Dim TX As Variant
    TX = Chr(27) + "D" + Chr(1)
    MSComm1.Output = TX
End Sub
Private Sub Command17_Click()
' 커서 감추기
    Dim TX As Variant
    TX = Chr(27) + "D" + Chr(&HC)
    MSComm1.Output = TX
End Sub
Private Sub Command18_Click()
' 커서 보이기
    Dim TX As Variant
    TX = Chr(27) + "D" + Chr(&HE)
    MSComm1.Output = TX
End Sub
Private Sub Command19_Click()
' 블링크 ON
    Dim TX As Variant
    TX = Chr(27) + "D" + Chr(&HB)
    MSComm1.Output = TX
End Sub
Private Sub Command20_Click()
' 블링크 OFF
    Dim TX As Variant
    TX = Chr(27) + "D" + Chr(&HA)
    MSComm1.Output = TX
End Sub
Private Sub Command21_Click()
' 유저 정의 영역 사용
    Dim TX As Variant
    TX = Chr(27) + "E" + Chr(8) + Chr(0) + Chr(255) + Chr(0)
       + Chr(255) + Chr(0) + Chr(255) + Chr(0) + Chr(255)
    MSComm1.Output = TX
    TX = Chr(27) + "C" + Chr(8) + Chr(0)
    MSComm1.Output = TX
End Sub
Private Sub Command22_Click()
' 백라이트 ON
    Dim TX As Variant
    TX = Chr(27) + "F" + Chr(1)
    MSComm1.Output = TX
End Sub
Private Sub Command23_Click()
' 백라이트 OFF
    Dim TX As Variant
    TX = Chr(27) + "F" + Chr(2)
    MSComm1.Output = TX
End Sub
Private Sub Command3_Click()
    Dim TX As Variant
    TX = Chr(27) + "A" + Chr(27) + "F" + Chr(1) + Chr(27)
        + "B" + Chr(0) + Chr(0) + Chr(27) + "C" + "ABCDEFG"
        + Chr(0)
    MSComm1.Output = TX
End Sub
Private Sub Command4_Click()
    Dim TX As Variant
    TX = Chr(27) + "D" + Chr(1) + Chr(27) + "F"
         + Chr(2) + Chr(27) + "G" + Chr(24)
    MSComm1.Output = TX
End Sub
Private Sub Command5_Click()
    Dim TX As Variant
    TX = Chr(27) + "G" + Chr(&H26)
    MSComm1.Output = TX
End Sub
Private Sub Command6_Click()
    Dim TX As Variant
    TX = Chr(27) + "G" + Chr(&H28)
    MSComm1.Output = TX
End Sub
Private Sub Command7_Click()
    Dim TX As Variant
    TX = Chr(27) + "G" + Chr(&H25)
    MSComm1.Output = TX
End Sub
Private Sub Command8_Click()
    Dim TX As Variant
    TX = Chr(27) + "G" + Chr(&H27)
    MSComm1.Output = TX
End Sub
Private Sub Command9_Click()
' 백스페이스
    Dim TX As Variant
    TX = Chr(27) + "G" + Chr(8)
    MSComm1.Output = TX
End Sub
```

11. 외형 치수 (단위 : ㎜)

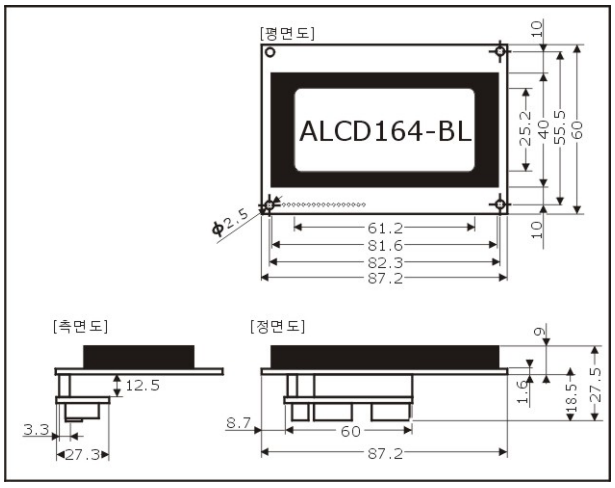
ALCD162-BL



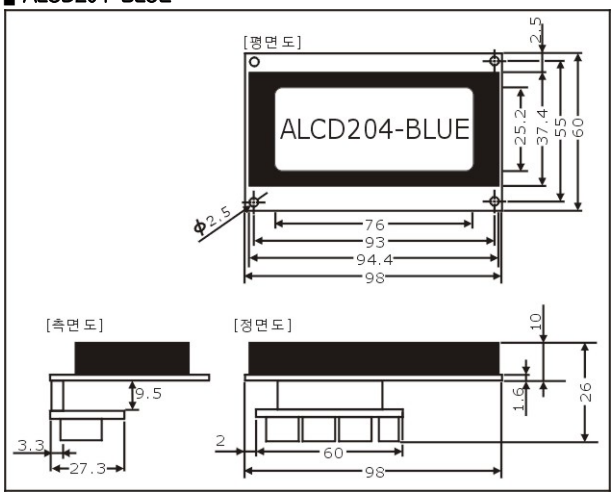
ALCD164-BLUE



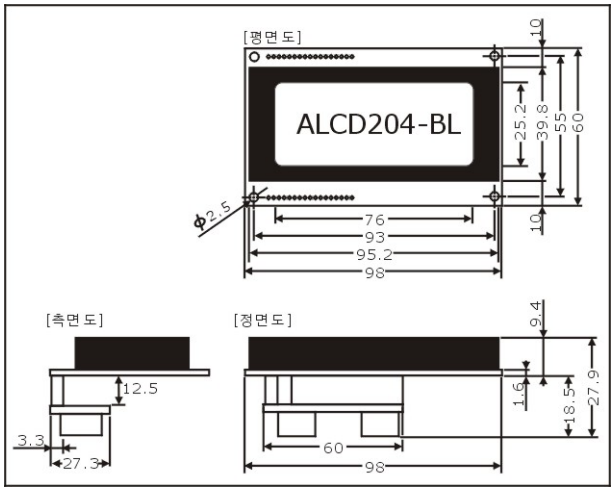
ALCD164-BL



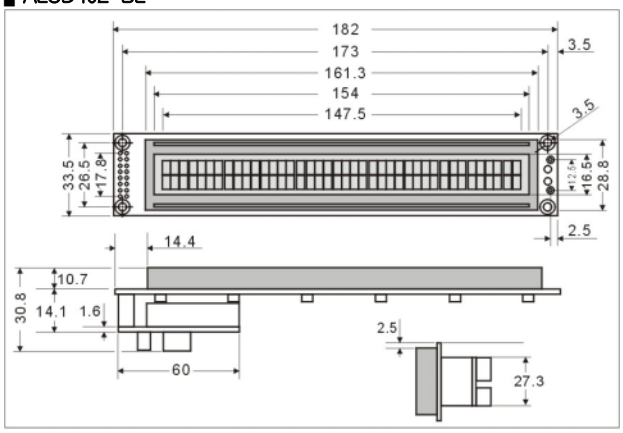
ALCD204-BLUE



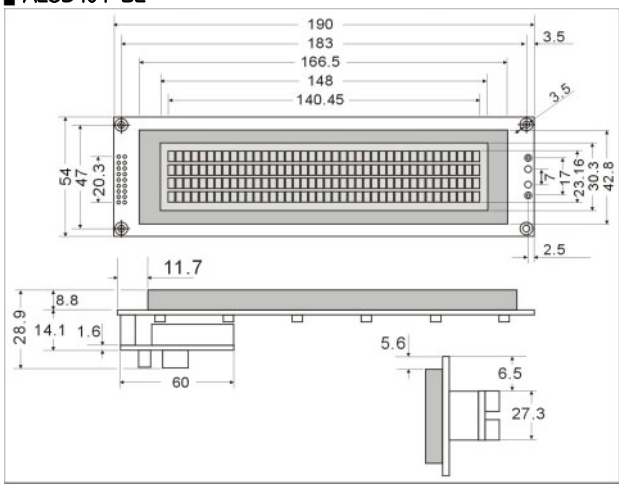
ALCD204-BL



ALCD402-BL



ALCD404-BL



ALCD 구동모듈