실험 5. SWITCH와 KEYPAD 사용하기, 부저 울리기, FND 표시하기

2019년 1학기

담 당 : 이인수교수

실험 5. SWITCH와 KEYPAD 사용하기, 부저 울리기

1. 개별 스위치 읽기

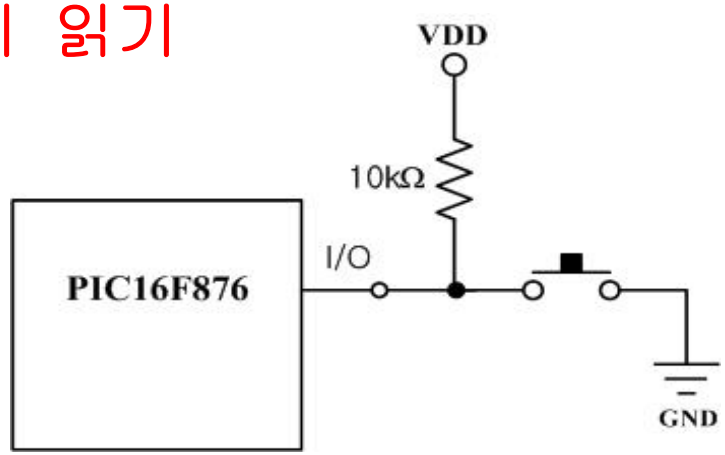


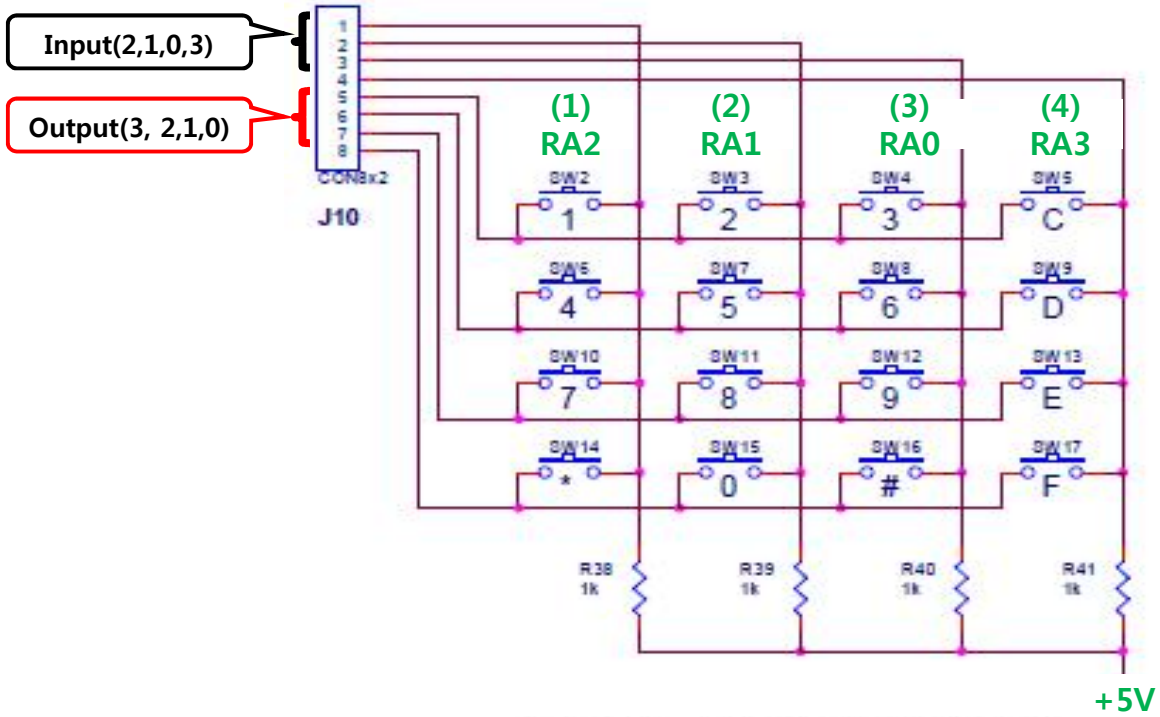
그림 E5-1의 회로에서 스위치가 눌러지지 않으면 +5 에 연결된 저항(pull up 저항이라 함)에 의해서 입력 신호는 +5가 되어 '1' 이 들어오며, 스위치가 눌러지면 0이 되어 로직으로는 '0' 이 들어온다.

* 16개의 Key를 사용하기 위해서는 16개의 I/O Pin이 필요. 22개의 Key 사용을 위해서는 22개의 I/O Pin이 필요. 그렇다면 출력장치는 어느 I/O Pin에 연결하나? 그러므로 효율적인 방법이 필요

-→ KEY I/O PAD : 위의 Key를 Key Matrix 구조로 만들어서 사용
(Key Matrix Scanning 방법)

2. Key Matrix Scanning 방법

- I/O 방향 설정방법, PORT 읽는 방법, KEY PAD의 구조, BUZZ 울리기



적은 수의 I/O 포트로 많은 수의 스위치 를 구현 하는 방법

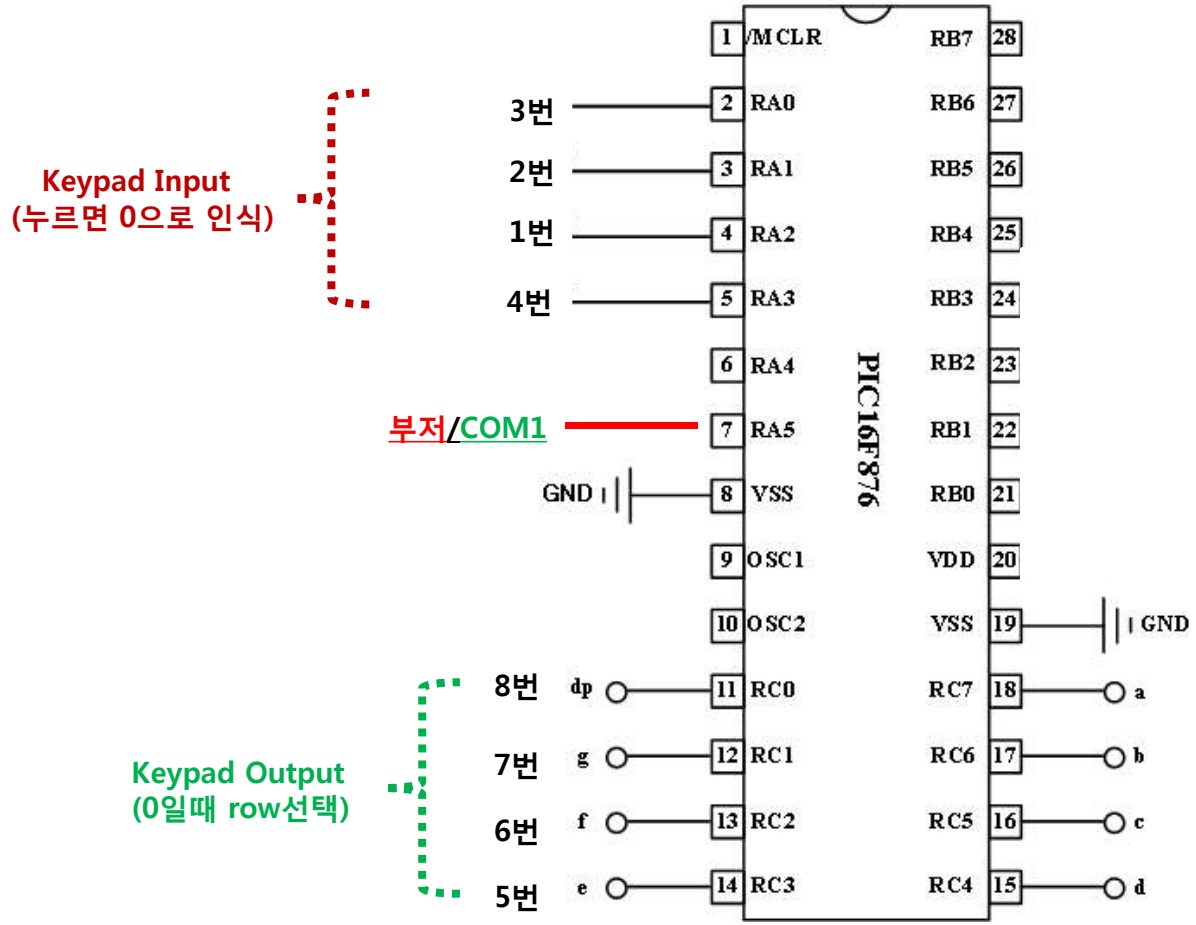
그림 E5-2. KEY MATRIX의 하드웨어 구조

INPUT pin 1개에 4개의 key 스위치가 병렬 연결된 것으로서 특정 key를 인식하기 위해서는 4개중 1개를 선택하기 위한 선택 신호가 필요하다. 이 신호가 OUTPUT PORT를 통해서 주어진다. 즉 OUTPUT pin에 '0' 을 출력하는 것만 선택되고, '1' 을 출력하는 것은 선택되지 않는다. 즉, key를 누를 경우 입력 측 단자가 '0' 으로 변화되는 것을 인식할 수 있음

-→ 16개의 key를 전부 인식하기 위해서는 OUTPUT을 4번 바꿔주고 INPUT을 4번. (I/O Pin 8개로 16개 Key 인식 가능)

실험 5. KEYPAD 사용하기, 부저 울리기, FND에 표시하기

실험 1-4 수행



I/O PORT DESCRIPTION

- PORTA
  - 6bits 양방향 포트 : [RA5~RA0]
    - **[RA3~RA0]동일 구조**
    - **[RA4] : Open Drain/Clock in**
    - **RA5**
  - Read-modify-write 동작
  - 전체허용 Source current : 60mA
    - **6개 모두 구동 → 10mA**
    - **핀 당 최대허용 source current : 20mA**
  - 전체허용 Sink current: 100mA
    - **핀 당 최대허용 Sink current → 25mA**

I/O PORT DESCRIPTION

◘ PORTA 관련 레지스터

TABLE 4-1: PORTA FUNCTIONS

Name	Bit#	Buffer	Function
RA0/AN0	bit 0	TTL	Input/output or analog input.
RA1/AN1	bit 1	TTL	Input/output or analog input.
RA2/AN2/VREF-/CVREF	bit 2	TTL	Input/output or analog input or VREF- or CVREF.
RA3/AN3/VREF+	bit 3	TTL	Input/output or analog input or VREF+.
RA4/T0CKI/C1OUT	bit 4	ST	Input/output or external clock input for Timer0 or comparator output. Output is open-drain type.
RA5/AN4/$\overline{SS}$/C2OUT	bit 5	TTL	Input/output or analog input or slave select input for synchronous serial port or comparator output.

Legend: TTL = TTL input, ST = Schmitt Trigger input

TABLE 4-2: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA

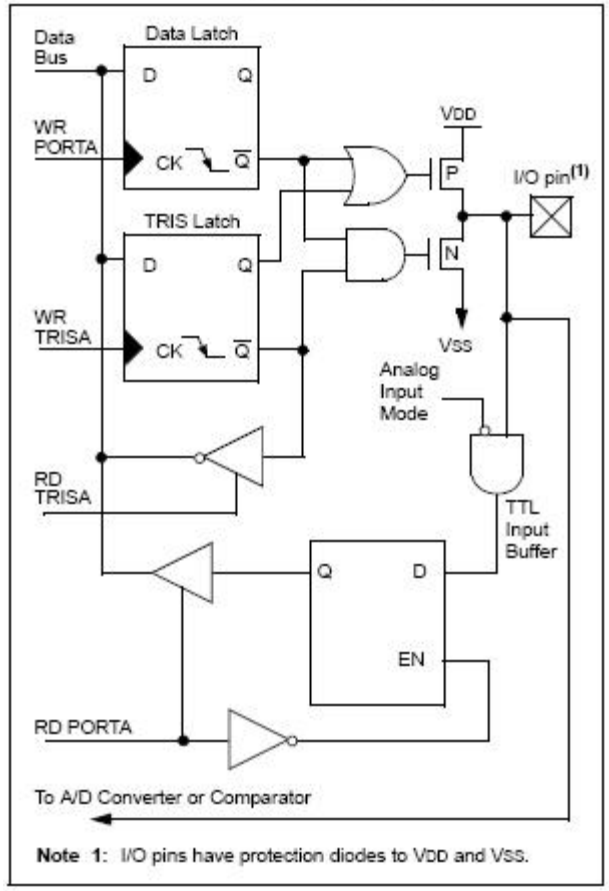
Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
05h	PORTA	—	—	RA5	RA4	RA3	RA2	RA1	RA0	--0x 0000	--0u 0000
85h	TRISA	—	—	PORTA Data Direction Register						--11 1111	--11 1111
9Ch	CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0111	0000 0111
9Dh	CVRCON	CVREN	CVROE	CVRR	—	CVR3	CVR2	CVR1	CVR0	000- 0000	000- 0000
9Fh	ADCON1	ADFM	ADCS2	—	—	PCFG3	PCFG2	PCFG1	PCFG0	00-- 0000	00-- 0000

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'. Shaded cells are not used by PORTA.

I/O PORT DESCRIPTION

- RA0~RA5
  - 포트 A의 양방향 I/O 포트, TTL레벨
  - RA0/AN0
  - RA1/AN1
  - RA2/AN2/VREF-
  - RA3/AN3/VREF+
  - RA4/T0CKI
  - RA5/AN4/(/SS)
    - 동기식 시리얼 포트에서 슬레이브 선택에 사용

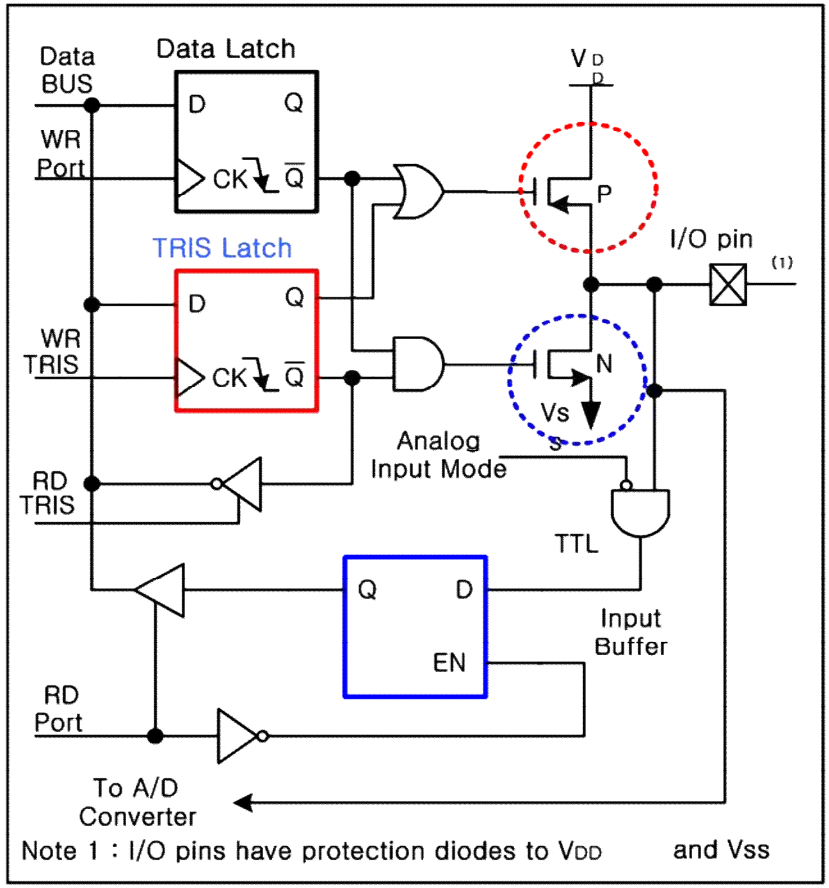
FIGURE 4-1: BLOCK DIAGRAM OF RA3:RA0 PINS



Note 1: I/O pins have protection diodes to VDD and VSS.

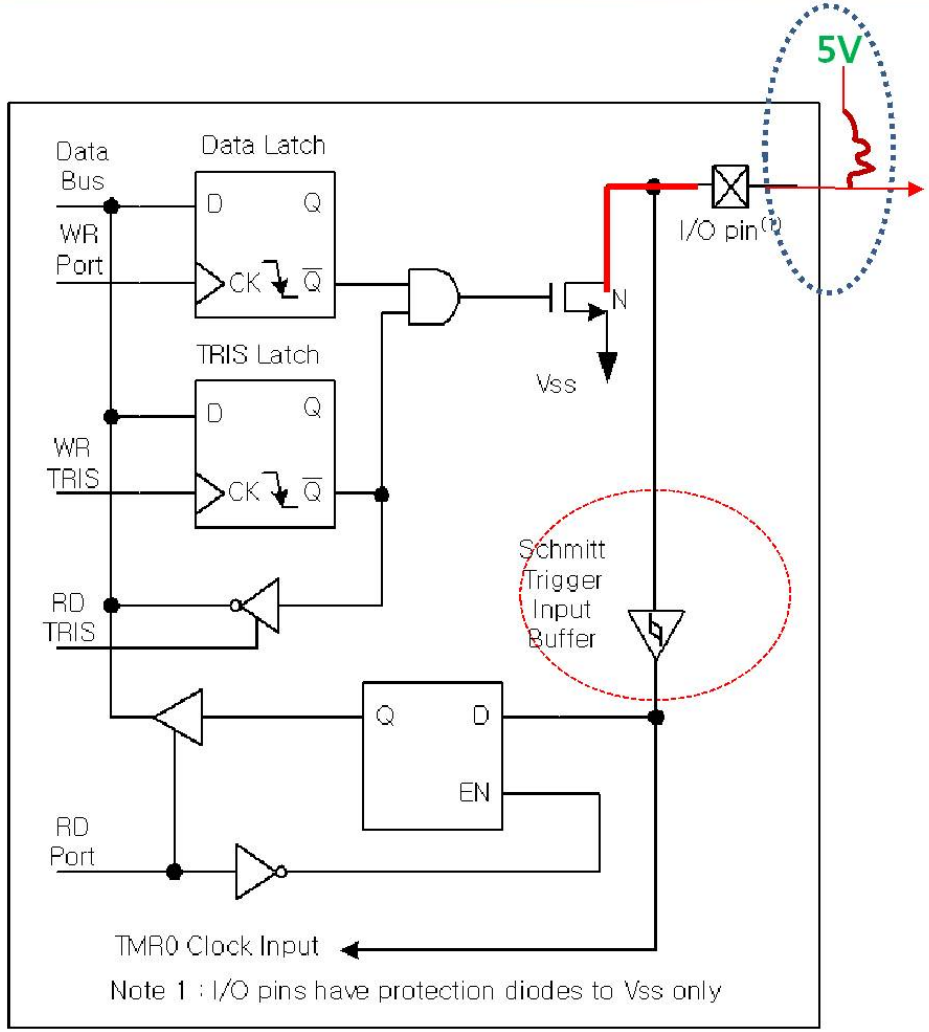
I/O PORT DESCRIPTION

- [RA3:RA0]
  - P-MOS,N-MOS로 흐르는
  - 허용 최대전류는 20mA,25mA



I/O PORT DESCRIPTION

- [RA4]
  - TOCKI
    - **RTCC(Real Time Clock Counter)**
  - Open Drain 구조
    - 높은 전압 사용가능
    - **사용상 주의사항**
      - 출력 단자에 저항을 반드시 연결
  - Schmitt Trigger Input
    - **Noise Margin 증가**
  - **Pull-up Resistor**
    - **2~6[Kohm] 정도의 낮은 저항 사용**



- MOS FET가 ON : 출력핀은 GND에 연결되므로 논리 "0"
  OFF : 출력핀은 Floatimg or high imdedance상태

→ 강제로 논리값이 "1"이 되게 해주어야 함

→ 풀업저항의 연결로 MOS FET OFF시에 "1"을 출력가능
(우리 실험키트에는 RA4에 풀업저항 연결되어 있음)

→ 그러나 풀업저항연결시 Power On 시에는 RA4는 high 상태가 되어서 Buzzer On

I/O PORT DESCRIPTION

◘ [RA4]

POWER ON RESET과 관련되어서…

초기 TRISA상태는 "1"인가 "0"인가?

➔1"로 설정 PORTA는 입력으로 설정되어짐 그리고

PORTA의 상태는 HIGH인가 LOW인가?

➔X, U의 값을 가짐(X :UNKNOWN, U:UNCHANGED)

결국 초기에 POWER ON RESET상태에서 RA4는 입력 포트로 설정 되어 있으므로 PULL UP 저항을 달아놓았다면 PIN 상태는 HIGH이며 BUZZ쪽에 연결되어 있는 TR이 ON 이 되어 부저가 울리게 된다.

- 어떻게 하면 이 문제를 해결 할 수 있을까?

① 트랜지스터의 베이스 쪽 저항 앞 즉 RA4 핀에 NOT 게이트를 달아 주면 초기에 BUZZ는 울리지 않을 것이다. 단, RA4번 동작은 반대로(BSF PORTA,4 일 경우 트랜지스터는 OFF되고, BCF PORTA,4의 경우 ON됨) 사용하면 된다.

② RA4를 사용하지 않으면 된다.

소프트웨어적으로 해결 할 방법은 없다. 이것은 구조적 문제이기 때문이다.

I/O PORT DESCRIPTION

◘ PORTB 관련 레지스터

TABLE 4-3: PORTB FUNCTIONS

Name	Bit#	Buffer	Function
RB0/INT	bit 0	TTL/ST(1)	Input/output pin or external interrupt input. Internal software programmable weak pull-up.
RB1	bit 1	TTL	Input/output pin. Internal software programmable weak pull-up.
RB2	bit 2	TTL	Input/output pin. Internal software programmable weak pull-up.
RB3/PGM(3)	bit 3	TTL	Input/output pin or programming pin in LVP mode. Internal software programmable weak pull-up.
RB4	bit 4	TTL	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up.
RB5	bit 5	TTL	Input/output pin (with interrupt-on-change). Internal software programmable weak pull-up.
RB6/PGC	bit 6	TTL/ST(2)	Input/output pin (with interrupt-on-change) or in-circuit debugger pin. Internal software programmable weak pull-up. Serial programming clock.
RB7/PGD	bit 7	TTL/ST(2)	Input/output pin (with interrupt-on-change) or in-circuit debugger pin. Internal software programmable weak pull-up. Serial programming data.

Legend: TTL = TTL input, ST = Schmitt Trigger input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.

2: This buffer is a Schmitt Trigger input when used in Serial Programming mode or in-circuit debugger.

3: Low-Voltage ICSP Programming (LVP) is enabled by default which disables the RB3 I/O function. LVP must be disabled to enable RB3 as an I/O pin and allow maximum compatibility to the other 28-pin and 40-pin mid-range devices.

TABLE 4-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB

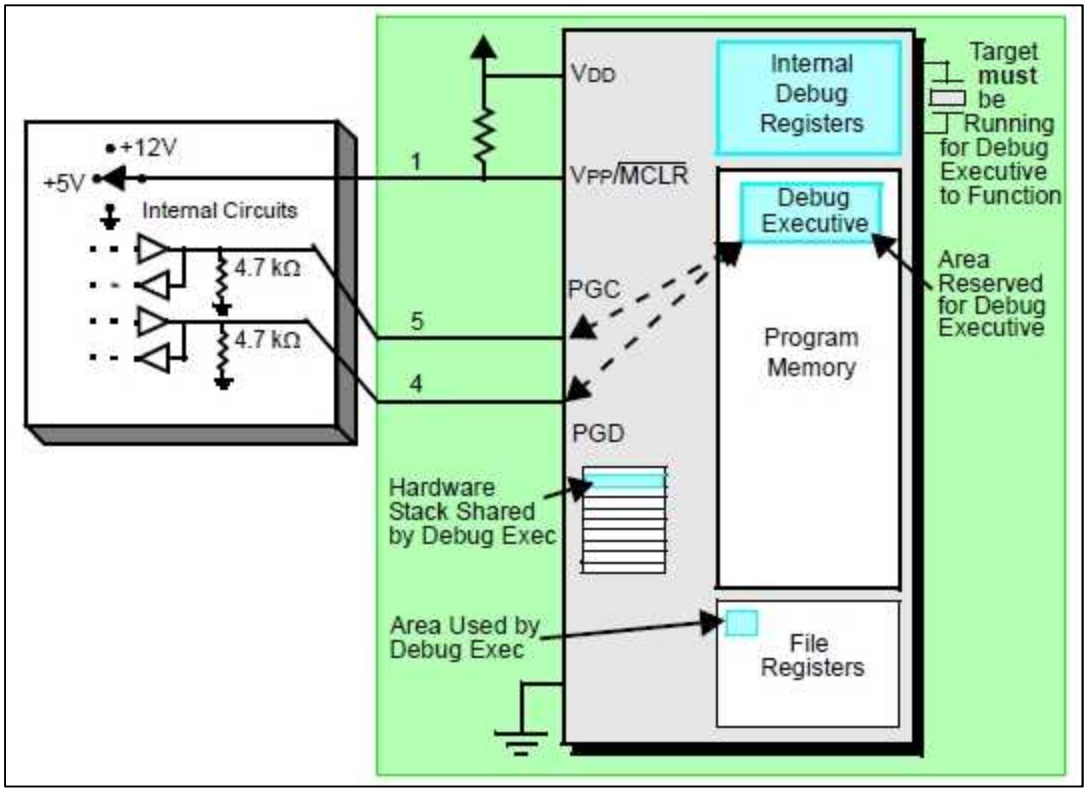
Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
06h, 106h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	uuuu uuuu
86h, 186h	TRISB	PORTB Data Direction Register								1111 1111	1111 1111
81h, 181h	OPTION_REG	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111

Legend: x = unknown, u = unchanged. Shaded cells are not used by PORTB.

이미 RB6와 RB7은 장비(REALICE)에서 사용하고 있으므로 I/O로 사용할 수 없음.
RB0-RB5까지는 I/O로 사용가능 함

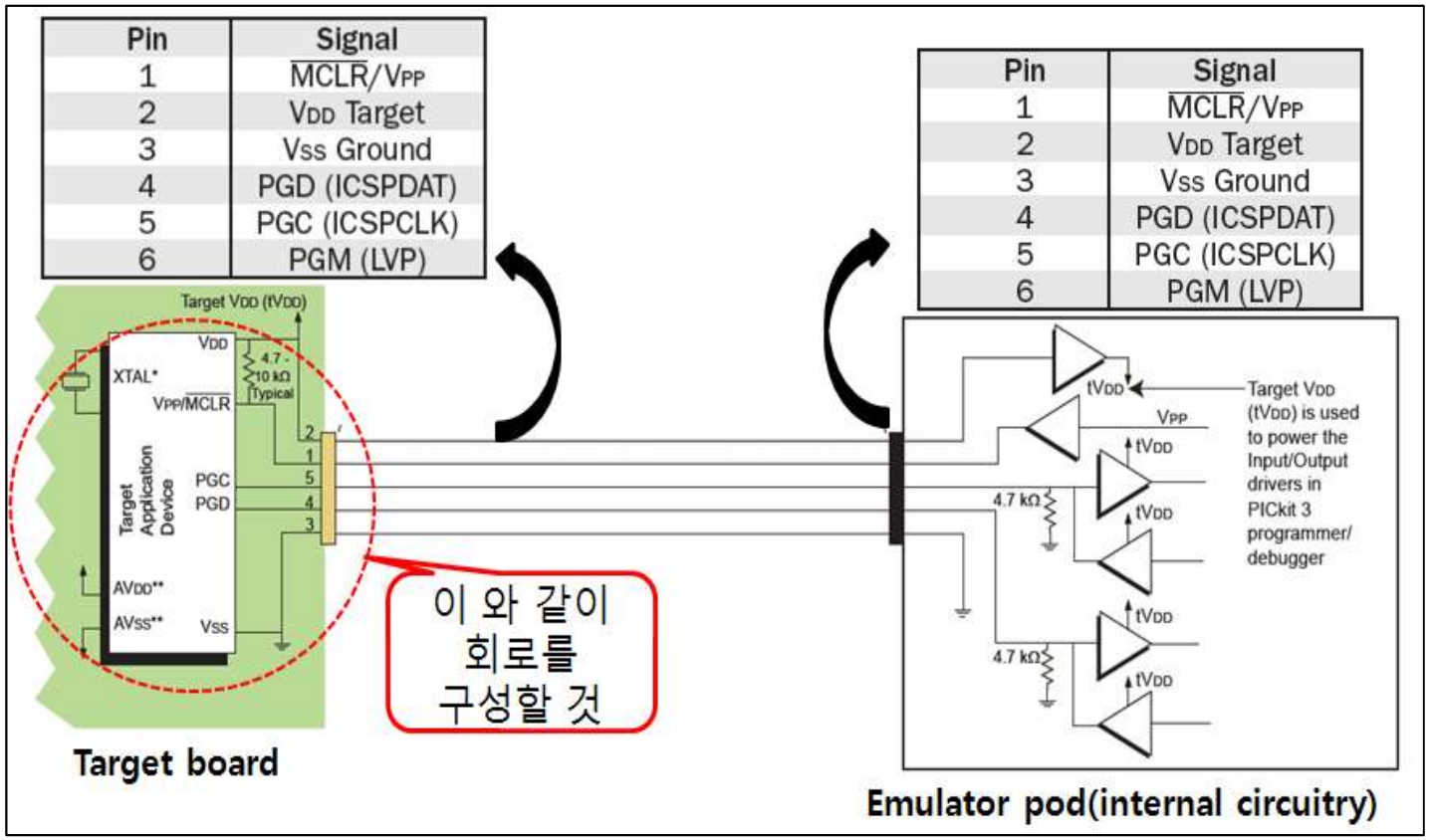
Development tools

- Real ICE
  - In-Circuit Emulator ready for debugging



Development tools

- Real ICE
  - Emulator pod 전기적 접속



Development tools

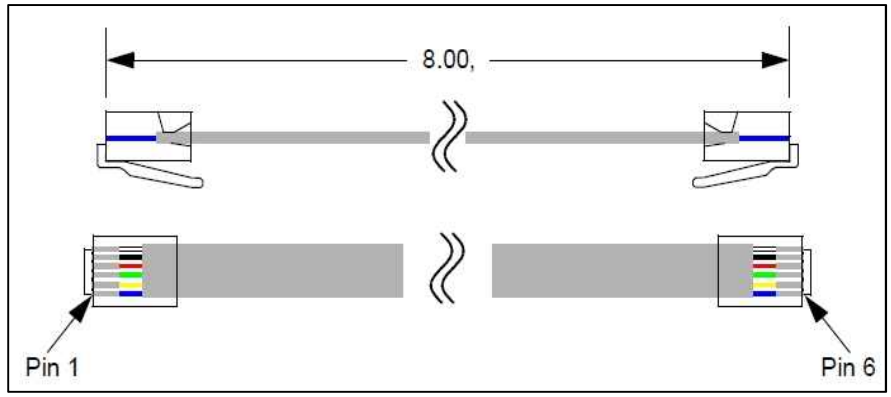
Real ICE

RJ-11 connect pin diagram

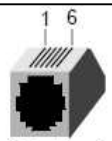
Front view of Modular Connector on Std Driver Board

Bottom view of Modular Connector Pinout on Std Driver Board

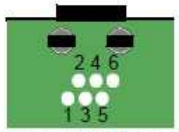
Modular Connector Pin	Microcontroller Pin
1	Not Used
2	RB6
3	RB7
4	Ground
5	VDD Target
6	VPP



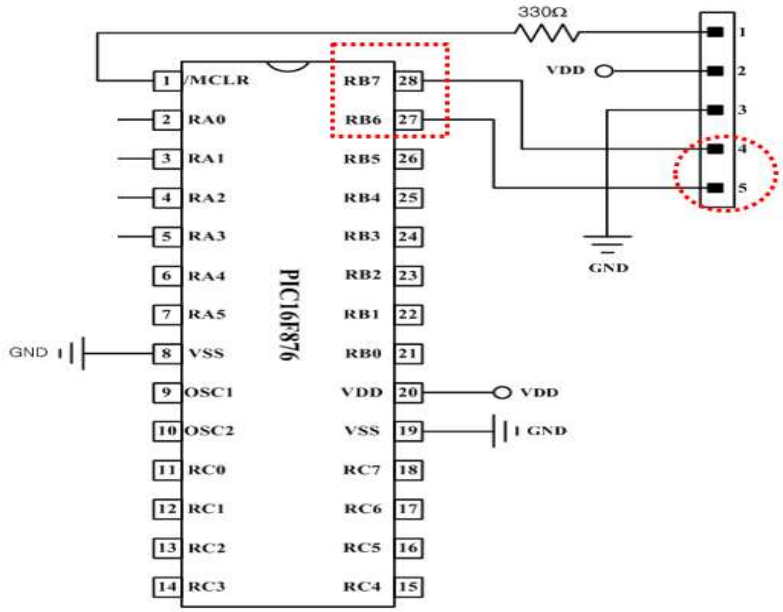
Modular Connector Pin	Microcontroller Pin
6	Not Used
5	RB6
4	RB7
3	Ground
2	VDD Target
1	VPP



Front view of Modular Connector on Target Board



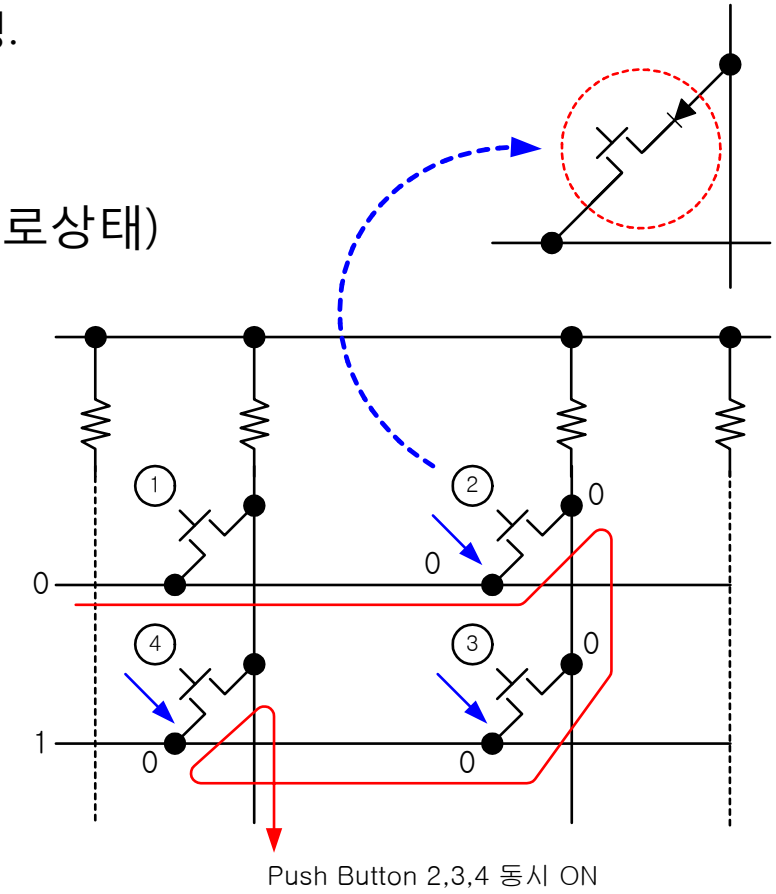
Bottom view of Modular Connector Pinout on Target Board



예비문제

- Roll over
  - 한 개 이상의 key가 동시에 눌러진 경우 발생.
  - Key의 scan 순서에 따라 key 값 결정.

- Sneak path (누설경로 : 바람직하지 않는 회로상태)
  - Diode 추가로 해결



- 참고.(관련 IC)
  - 8-key encoder → 74LS148
  - 20-key key board scan IC → MM74C923
  - MM5740
  - TTL(74LS147, 148, 348), CMOS(4532)
  - Intel 사 8279(Programmable Keyboard/Display Interface)

실험 5장

실험 1. Key 스위치 "2"가 눌러졌는가 아닌가를 구분하여 "2"가 눌러졌으면 Buzzer On시키는 프로그램 작성 (ex5-1)

Port A : 입력 4개 ➔ Key pad J10의 핀 4, 3, 2, 1에 연결(Intput 으로 사용) 즉, RA3-4번, RA0-3번, RA1-2번, RA2-1번
출력 1개 ➔ RA5를 Buzzer에 연결
Port C : 출력 5개 ➔ Key pad J10의 핀 5, 6, 7, 8에 연결(Output 으로 사용) 즉, RC3-5번, RC2-6번, RC1-7번, RC0-8번

* 부가실험 2)번 Key 4와 6을 동시에 누를때만 Buzzer On시키는 프로그램 작성

* "1", "2", "3", "4", "5", "6"중에서 어떤 Key가 눌러졌는지 찾는 P/G 105page (ex5-2)

실험 2. 실험 1에서 Look up Table를 이용하여 Key 스위치 "0"가 눌러졌는가 아닌가를 구분하여 "0"가 눌러졌으면 Buzzer On시키는 프로그램 작성 (ex5-3)

Port A : 입력 4개 ➔ Key pad J10의 핀 4, 3, 2, 1에 연결(Intput 으로 사용) 즉, RA3-4번, RA0-3번, RA1-2번, RA2-1번
출력 1개 ➔ RA5를 Buzzer에 연결
Port C : 출력 5개 ➔ Key pad J10의 핀 5, 6, 7, 8에 연결(Output 으로 사용) 즉, RC3-5번, RC2-6번, RC1-7번, RC0-8번

* 부가실험 5)번 Key 7와 8을 동시에 누를때만 Buzzer On시키는 프로그램 작성

실험 3. 실험 2의 내용에서 Key 번호를 FND에 표시하고 Buzzer On시키는 프로그램 작성 (ex5-4)

- 부가실험 7)번 프로그램 작성 (특정 Key 입력시 Buzzer On)
- Key 누르는거 FND 에 직접표시 (누르지 않으면 "F"표시)
- 다음 Key 누를때까지 유지

실험 1. Key 스위치 “2”가 눌러졌는가 아닌가를 구분하여 “2”가 눌러졌으면 Buzzer On시키는 프로그램 작성 (ex5-1)

Port A : 입력 4개 ➔ Key pad J10의 핀 4, 3, 2, 1에 연결(Intput 으로 사용) 즉, RA3-4번, RA0-3번, RA1-2번, RA2-1번

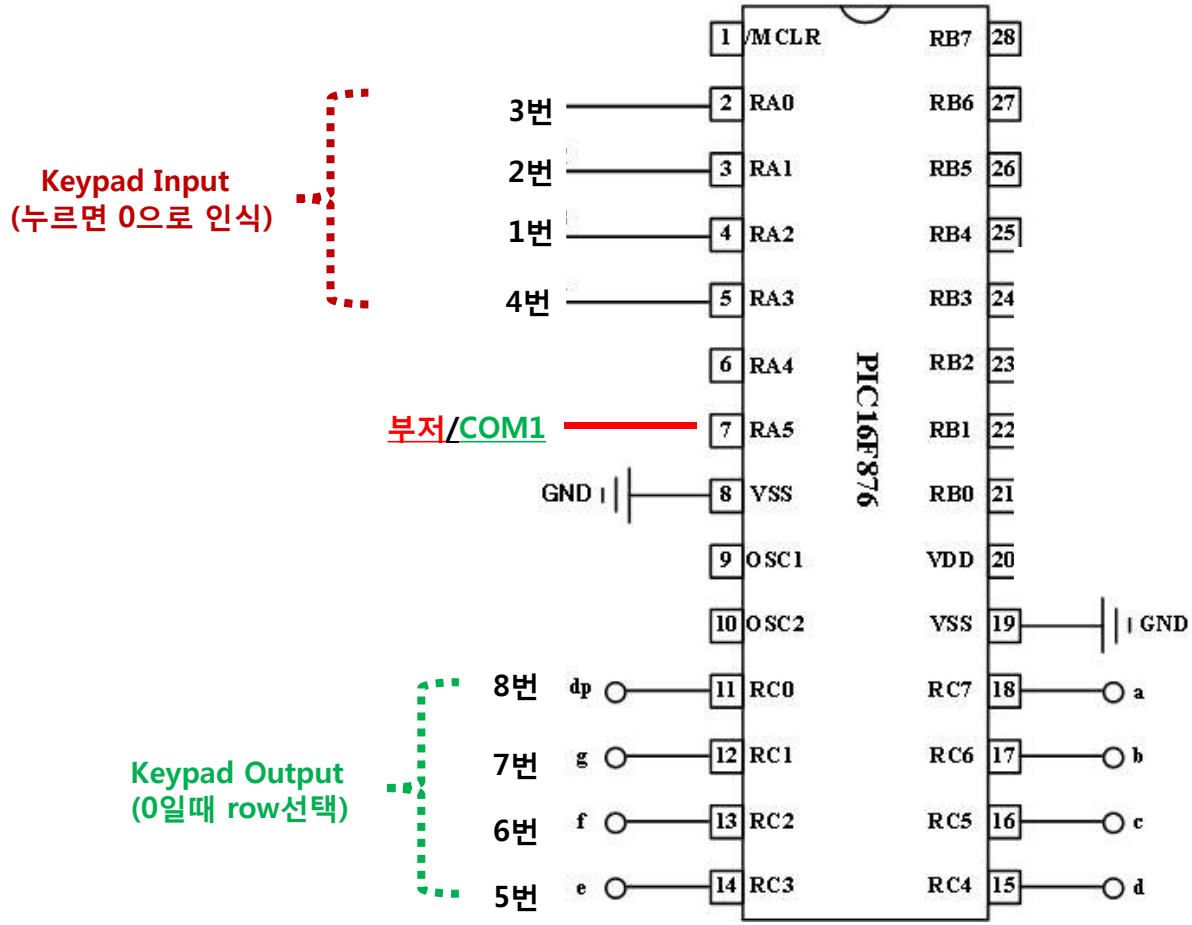
출력 1개 ➔ RA5를 Buzzer에 연결

Port C : 출력 5개 ➔ Key pad J10의 핀 5, 6, 7, 8에 연결(Output 으로 사용) 즉, RC3-5번, RC2-6번, RC1-7번, RC0-8번

* 부가실험 2)번 Key 4와 6을 동시에 누를때만 Buzzer On시키는 프로그램 작성

-→ 다양한 ROW선택과 스위치 선택 위한 실험 해볼것!!!

실험 1-4 수행회로


Keypad Input
(누르면 0으로 인식)
3번
2번
1번
4번
부저/COM1
GND
Keypad Output
(0일때 row선택)
8번
7번
6번
5번
dp
g
f
e
PIC16F876
1 /MCLR
2 RA0
3 RA1
4 RA2
5 RA3
6 RA4
7 RA5
8 VSS
9 OSC1
10 OSC2
11 RC0
12 RC1
13 RC2
14 RC3
28 RB7
27 RB6
26 RB5
25 RB4
24 RB3
23 RB2
22 RB1
21 RB0
20 VDD
19 VSS
18 RC7
17 RC6
16 RC5
15 RC4
GND
a
b
c
d

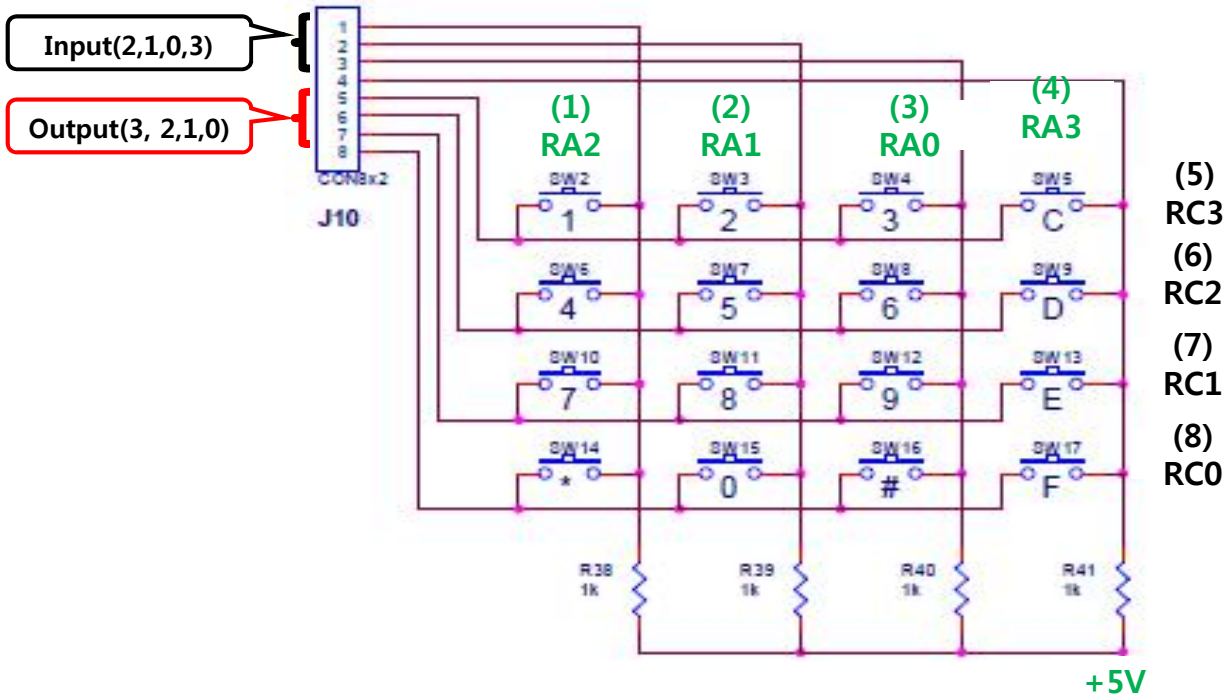
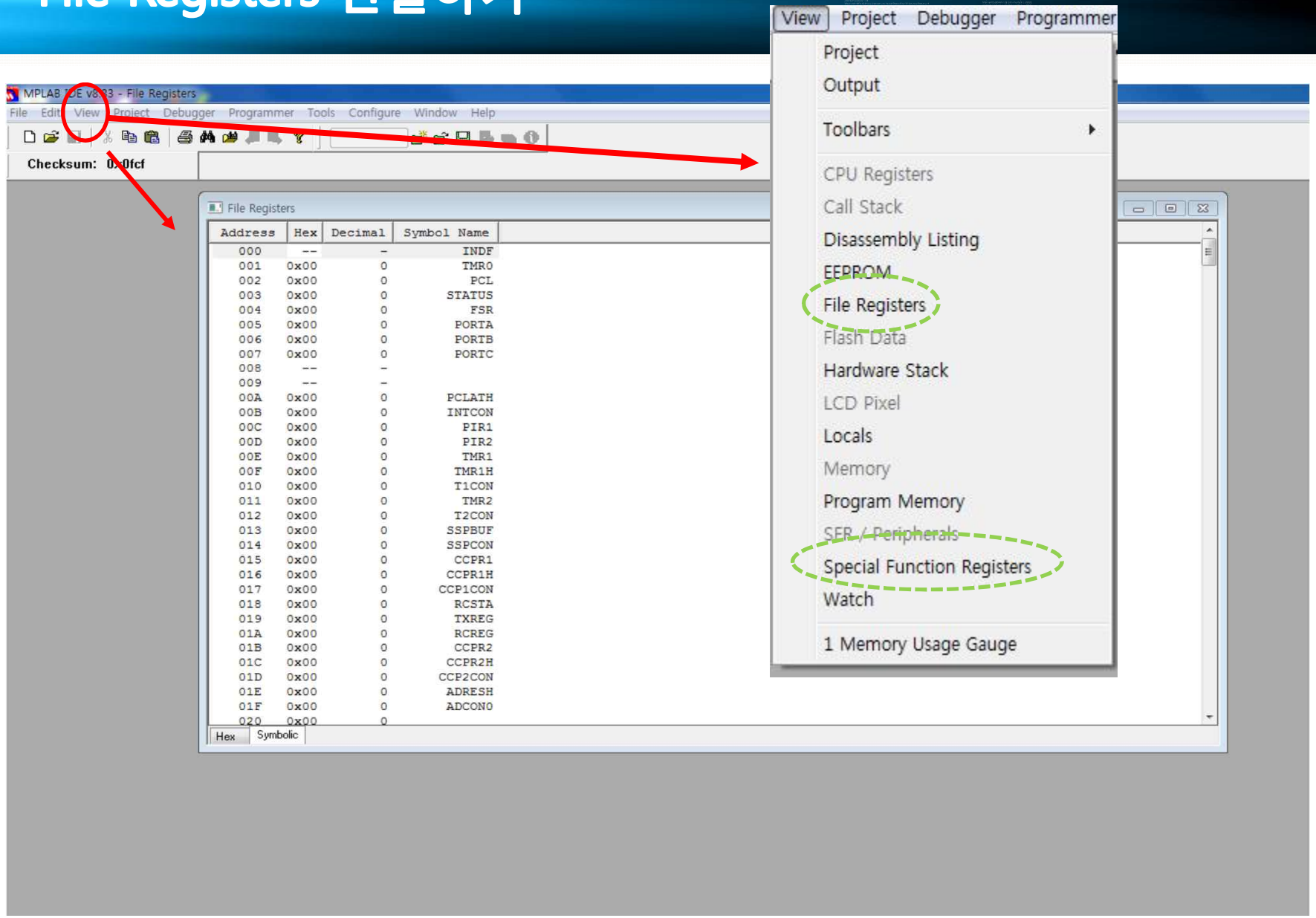


그림 E5-2. KEY MATRIX의 하드웨어 구조

File Registers 관찰하기



실험 5

```
; TEXT 실험 5 (104PAGE)
; Key pad와 부져 이용 실험 예1) 104page
; STANDARD HEADER FILE
각종 설정 부분 생략

;  MAIN PROGRAM
ORG    0000
                BSF                     STATUS,RP0 ;Bank1로 지정

                MOVLW       B'00001111'  ;PORTA RA4, RA5 출력, RA0 - 3은 입력으로 설정
                MOVWF       TRISA
                MOVLW       B'00000000'
                MOVWF       TRISB
                MOVLW       B'00000000' ;PORTC 출력으로 설정 FND
                MOVWF       TRISC ;PORTC의 4비트를 출력으로 설정
               MOVLW        B'00000111'
                MOVWF    ADCON1

        BCF                 STATUS,RP0 ; PORT에 입출력시에는 반드시 Bank0로 해야함
```

실험 5

```
;       I/O PORT 선택    (2번 스위치 관찰)
;       RC PORT OUTPUT
;       RA4 PORT OUTPUT, RA2,1,0 PORT INPUT

이 값바꾸면 다른 ROW 선택가능 -->      MOVLW     B'00000111'   ; RC3=0, RC2,1,0=1 (첫번째 Row 선택)
                                ;MOVLW     B'00001011'   ; RC3=1, RC2=0, RC1,0=1 (두번째 Row 선택)
                    MOVWF       PORTC
        LP          MOVF        PORTA,W                 ; 스위치 읽기 (2번이 ON되면 1101이 됨)
                    ANDLW       B ' 00001111 '          ; Mask
이 값바꾸면 다른 스위치 모니터링--> SUBLW   B'00001101 ' ; 2번이 ON되면 "00000000" 이 되어 ZF가 1이됨
                     ; SUBLW    B'00001011 ' ; 1번 모니터링시에는
                     ; SUBLW    B'00001010 ' ; 4번과 6번 동시에 모니터링시에는
                    BTFSC                   STATUS,ZF ; ZF가 0이면 GOTO BUZZOFF, 1이면 GOTO BUZZON
                    GOTO                    BUZZON
                    GOTO                    BUZZOFF
              BUZZON
                    BSF                     PORTA,5   ;BUZZER ON, PORTA의 6번째 PIN RA5에 1을 출력
                   ; BSF                    PORTA,4   ;BUZZER ON, PORTA의 5번째 PIN RA4에 1을 출력
                    GOTO                    LP
              ;CALL    DELAY

              BUZZOFF
                    BCF                     PORTA,5    ;BUZZER OFF
                    GOTO                    LP
```

```
;SUBROUTINE
DELAY
        MOVLW       .10
      MOVWF         DBUF1                   ; 130번을 확인하기 위한 변수
LP1     MOVLW       .255
        MOVWF       DBUF2
LP2     MOVLW       .255
        MOVWF       DBUF3                   ; 200번을 확인하기 위한 변수
LP3     NOP
        DECFSZ      DBUF3,F
        GOTO                LP3
        DECFSZ      DBUF2,F     ; 변수를 감소시켜 가면서 00이 되었나 확인
        GOTO                LP2                     ; ZERO가 아니면 여기에 들어옴.
      DECFSZ        DBUF1,F     ; 변수를 감소시켜 가면서 00이 되었나 확인
        GOTO                LP1                     ; ZERO가 아니면 여기에 들어옴.
        RETURN

        END
```

예제 2) 교과서 105PAGE

: 실험 1은 특정키가 눌러졌는가를 인식하고 BUZZER를 울림

-→ 이를 확장하여 **"1", "2", "3", "4", "5", "6"**중에서 어떤 **Key**가 눌러졌는지 찾는 **P/G (ex5-2)**

여기서는 **3*4**개 **Pad**만 고려 : **1 2 3, 4 5 6, 7 8 9, * 0 #**

최종적으로 KEY_DATA 값이 눌러진 key 값이 됨

-→ (View) **File Register 027H** 확인하면 **key번호 있음**

-→ (1-6번 외의 Key 누르면 KEY_DATA에 0F 저장)

예제 2) 교과서 105PAGE

```
KEY_IN
        MOVLW   0FH             ;초기값으로 key가 눌러지지 않음 설정
        MOVWF   KEY_DATA

LP      MOVLW   B'00000111' ; RC3=0, RC2,1,0=1    ; 첫번째 row
        MOVWF   PORTC
        ;
        MOVF    PORTA, W    ; key 읽기
        ANDLW   B'00000111‘ ; PortA 3bit Mask
        SUBLW   B'00000011' ; 1번 key가 눌러졌는가 확인  맞으면 00000000 가 되어 ZF =1
        BTFSC               STATUS,ZF ; ZF =1이면  GOTO SW_1,  ZF =0이면 다음수행
        GOTO                SW_1  ; Yes

        MOVF    PORTA,W
        ANDLW   B'00000111'
        SUBLW   B'00000101' ; 2번 key가 눌러졌는가 확인
        BTFSC               STATUS,ZF
        GOTO                SW_2
        MOVF                PORTA,W
        ANDLW   B'00000111'
        SUBLW   B'00000110' ; 3번 key가 눌러졌는가 확인
        BTFSC               STATUS,ZF
        GOTO                SW_3
```

```
MOVLW     B'00001011'  ; RC3,1,0=1, RC2=0  두번째 row 조사
MOVWF     PORTC
;
MOVF      PORTA,W      ; key 읽기
ANDLW     B'00000111'
SUBLW     B'00000011'  ; 4번 key가 눌러졌는가 확인
BTFSC                  STATUS,ZF
GOTO                   SW_4
MOVF                   PORTA,W
ANDLW     B'00000111'
SUBLW     B'00000101'  ; 5번 key가 눌러졌는가 확인
BTFSC                  STATUS,ZF
GOTO                   SW_5
MOVF                   PORTA,W
ANDLW     B'00000111'
SUBLW     B'00000110'  ; 6번 key가 눌러졌는가 확인
BTFSC                  STATUS,ZF
GOTO                   SW_6
GOTO                   LP
```

```
SW_1
        MOVLW   1
        GOTO            XLP
SW_2
        MOVLW   2
        GOTO            XLP
SW_3
        MOVLW   3
        GOTO            XLP
SW_4
        MOVLW   4
        GOTO            XLP
SW_5
        MOVLW   5
        GOTO            XLP
SW_6
        MOVLW   6
XLP
        MOVWF   KEY_DATA ; KEY_DATA 값이 눌러진 key 값이됨   -→ File Register 027H 확인
        GOTO            LP
```

예제 3) 교과서 107Page
: 예제 2에서 OUTPUT 반복부분만 간소화 해보자(PORTC)

-→ OUTPUT은 '0111'에서 '1011', '1101', '1110' 다시 '0111'로 반복하며, -→ row 선택 (1...4)

INPUT는 '011', '101', '110'으로 반복한다. -→ column 선택**(4번째는 사용 않함)**

먼저 **OUTPUT는 4bit를 '0111'에서 오른쪽으로 shift(rotate 명령어 사용)**하면 된다. 구체적으로는 rotate 명령어는 CARRY와 함께 동작하므로 CARRY에 대한 고려를 해야 한다.

그러나 **RC3, 2, 1, 0를 사용하고, B'11110111'을 초기값으로 두고 RRF 명령어를 사용하면 들어오는 CARRY는 고려할 필요가 없으며, 끝은 CARRY가 발생하지 않을 때이다.**

예제 3) 교과서 107PAGE

```
                MOVLW     0FH                    ; key가 눌러지지 않음 설정
                MOVWF     KEY_DATA
;
LP2             MOVLW     B'11110111'            ; RC3=0, RC2,1,0=1  초기치 임
                MOVWF     PORTC
;
LP1             CALL      READ_KEY   ; key가 눌러짐 확인
                RRF       PORTC,F
                BTFSC     STATUS, CF  ; -→ CF가 0 이면 4번째 Row 까지 탐색완료이므로 LP2로 가서 초기화
                                      ;   (첫번째 Row부터 다시 탐색)
                GOTO                  LP1   ; 마지막 LOOP --> 초기화
                GOTO                  LP2
; key가 눌러짐 확인 눌러진 key 확인
READ_KEY
                  …
RETURN
```

실험 2. 실험 1에서 KEY_DATA얻기위한 Look up Table를 이용하여 Key 스위치 "0"가 눌러졌는가 아닌가를 구분하여 "0"가 눌러졌으면 Buzzer On시키는 프로그램 작성 (ex5-3)

"0" 은 4번째 row, 2번째 col. 요소 : input은 101, output은 11(1110)

Port A : 입력 4개 ➔ Key pad J10의 핀 4, 3, 2, 1에 연결(Intput 으로 사용) 즉, RA3-4번, RA0-3번, RA1-2번, RA2-1번

출력 1개 ➔ RA5를 Buzzer에 연결

Port C : 출력 5개 ➔ Key pad J10의 핀 5, 6, 7, 8에 연결(Output 으로 사용) 즉, RC3-5번, RC2-6번, RC1-7번, RC0-8번

-→ 교과서 108page 예제 5)를 이용하여 작성

Table address: Input 3bit, Output 2bit로 재구성

1개의 s/w 누른경우만 고려

Input : 011, 101, 110 의 3가지상태만 유효, 나머지 001, 100, 010, 000, 111은 안눌린상태로 간주

Output: 0111→ 00, 1011→ 01, 1101→ 10, 1110→ 11

* 부가실험 5)번 Key 7와 8을 동시에 누를때만 Buzzer On시키는 프로그램 작성

예제 5) 교과서 108-110PAGE

```
; TEXT 실험 5 (108PAGE)
; Key pad와 부져 이용 실험 예5) 108page
; STANDARD HEADER FILE

      DBUF1   EQU     24H ;GPR레지스터에 저장 (중요)
      DBUF2   EQU     25H ;GPR레지스터에 저장 (중요)
      DBUF3   EQU     26H ;GPR레지스터에 저장 (중요)
      KEY_DATA    EQU     27H ;GPR레지스터에 저장 (중요)
      KEY_T     EQU     28H ;GPR레지스터에 저장 (중요)

;  MAIN PROGRAM
ORG             0000
                        BSF                     STATUS,RP0 ;Bank1로 지정

                        MOVLW       B'00001111'  ;PORTA RA4 입력, RA0 - 3은 입력으로 설정
                        MOVWF       TRISA
                        MOVLW       B'00000000'
                        MOVWF       TRISB
                        MOVLW       B'00000000' ;PORTC 출력으로 설정 FND
                        MOVWF       TRISC ;PORTC 출력으로 설정
                        MOVLW       B'00000111'
                        MOVWF    ADCON1
                        BCF   STATUS,RP0 ; PORT에 입출력시에는 반드시 Bank0로 해야함


;               I/O PORT 선택
;               RC PORT OUTPUT
;               RA4 PORT OUTPUT, RA2,1,0 PORT INPUT

;               KEY_IN
;               MOVLW       0FH             ;초기값으로 key가 눌러지지 않음 설정
;               MOVWF       KEY_DATA
```

```
LP          MOVLW     0FH
            MOVWF     KEY_DATA    ; 초기화 (최종적으로 누른스위치 번호가 들어있음)
LP2         MOVLW     B'11110111' ; RC3=0, RC2,1,0=1
            MOVWF     PORTC
            CLRF      KEY_T                ; SCAN 위치 (bit 0와 1)
;
LP1         CALL                  READ_KEY  ; key가 눌러짐 확인
            INCF                  KEY_T,F   ; PORTC 선택 00(0111)→01(1011)→10(1101)→11(1110)
            RRF                   PORTC,F   ; 다음 위치 선택
            BTFSC                 STATUS,CF ; 마지막 위치 확인
            GOTO                  LP1
;  마지막 위치 --> key 값에 따라서 주어진 일하기
            MOVF                  KEY_DATA,W

            SUBLW     0                    ; '0' key 인가 확인
; SUBLW     1                    ; '1' key 인가 확인
; SUBLW     2                    ; '2' key 인가 확인
; SUBLW     3                    ; '3' key 인가 확인
; SUBLW     4                    ; '4' key 인가 확인
; SUBLW     5                    ; '5' key 인가 확인
; SUBLW     6                    ; '6' key 인가 확인
; SUBLW     7                    ; '7' key 인가 확인
; SUBLW     8                    ; '8' key 인가 확인
; SUBLW     9                    ; '9' key 인가 확인
 ; SUBLW    46H                  ; '4'와 '6' key 동시에 ON인가 확인 010 01
 ; SUBLW    78H                  ; '7'와 '8' key 동시에 ON인가 확인 001 10

 BTFSS      STATUS,ZF
            GOTO                  LP3 ; '0' key가 아니면 OFF
;  특정시간 동안 소리내기
            BSF                   PORTA,5   ; BUZZER ON  '0' key이면 수행
            CALL      DELAY
LP3         BCF                   PORTA,5   ; BUZZER OFF
            GOTO                  LP                ;  --> 초기화
```

```
; 스위치가 눌러짐 확인
READ_KEY
        MOVF    PORTA,W   ; 스위치 읽기
        ANDLW   B'00000111'
        SUBLW   B'00000111'  ; 눌러졌는지 확인하기 위함. 눌러졌으면 ZF는 0
        BTFSC   STATUS,ZF    ; key 눌러짐 확인(눌러졌으면 ZF는 0 이므로 RETURN 생략)
        RETURN                          ;key가 눌러지지 않으면 그냥 return
; key 값을 얻기 위한 TABLE ADDRESS 만듦
        MOVF    PORTA,W
        MOVWF   KEY_DATA
        RLF             KEY_DATA,F   }
        RLF             KEY_DATA,W   }  (두 비트 이동시킴)
        ANDLW   B'00011100'
        IORWF           KEY_T,W      }  Table address bit (5개 bit를 만듦)
        ;ANDLW  B'00011111' ;코멘트로
        CALL            KEY_TABLE
        MOVWF   KEY_DATA            ; 들어온 스위치 값
        RETURN
```

실험 5

Key scan algorithm

RC3 RC0

	RB3	RB2	RB1	RB0	RA2	RA1	RA0	
key number	(OUTPUT)				(INPUT)			
1	0111 = 00				011			1
2					101			2
3					110			3
4	1011 = 01				011			4
5					101			5
6					110			6
7	1101 = 10				011			7
8					101			8
9					110			9
*	1110 = 11				011			*
0					101			0
#					110			#

1	2	3
4	5	6
7	8	9
*	0	#

```
KEY_TABLE
CLRF      PCLATH    ;사용상 주의
ANDLW     B'00011111'  ; 2^5=32 개의 LOOK-UP TABLE
ADDWF     PCL, F    ; PCL=PCL+W
RETLW     0FH       ; '000'+'00' 일 때  → 0
RETLW     0FH       ; '000'+'01' 일 때  → 1
RETLW     0FH       ; '000'+'10' 일 때  → 2
RETLW     0FH       ; '000'+'11' 일 때
RETLW     0FH       ; '001'+'00' 일 때
RETLW     0FH       ; '001'+'01' 일 때
RETLW     78H       ; '001'+'10' 일 때; 78H로 수정시 7, 8동시 ON

RETLW     0FH       ; '001'+'11' 일 때
RETLW     0FH       ; '010'+'00' 일 때
RETLW     0FH       ; '010'+'01' 일 때 ; 46H로 수정시 4, 6동시 ON
RETLW     0FH       ; '010'+'10' 일 때
RETLW     0FH       ; '010'+'11' 일 때 *+#
RETLW     01H       ; '011'+'00' 일 때  1
RETLW     04H       ; '011'+'01' 일 때  4
RETLW     07H       ; '011'+'10' 일 때  7
RETLW     10H       ; '011'+'11' 일 때  *
RETLW     0FH       ; '100'+'00' 일 때
RETLW     0FH       ; '100'+'01' 일 때
RETLW     0FH       ; '100'+'10' 일 때
RETLW     0FH       ; '100'+'11' 일 때
RETLW     02H       ; '101'+'00' 일 때  2
RETLW     05H       ; '101'+'01' 일 때  5
RETLW     08H       ; '101'+'10' 일 때  8
RETLW     00H       ; '101'+'11' 일 때  0
RETLW     03H       ; '110'+'00' 일 때  3
RETLW     06H       ; '110'+'01' 일 때  6
RETLW     09H       ; '110'+'10' 일 때  9
RETLW     11H       ; '110'+'11' 일 때  #
RETLW     0FH       ; '111'+'00' 일 때
RETLW     0FH       ; '111'+'01' 일 때
RETLW     0FH       ; '111'+'10' 일 때
RETLW     0FH       ; '111'+'11' 일 때  → 31
```

```
; KEY 값을 저장하는 TABLE  -- 32 개임 (2^5)
                KEY_TABLE
                        ADDWF     PCL,F
                        RETLW     0FH         ; '000'+'00' 일때
                        RETLW     0FH         ; '000'+'01' 일때
                        RETLW     0FH         ; '000'+'10' 일때
                        RETLW     0FH         ; '000'+'11' 일때
                        RETLW     0FH         ; '001'+'00' 일때
                        RETLW     0FH         ; '001'+'01' 일때
                        RETLW     0FH         ; '001'+'10' 일때
                         RETLW    0FH         ; '001'+'11' 일때
                        RETLW     0FH         ; '010'+'00' 일때
                        RETLW     0FH         ; '010'+'01' 일때
                        RETLW     0FH         ; '010'+'10' 일때
                        RETLW     0FH         ; '010'+'11' 일때
                        RETLW     01H         ; '011'+'00' 일때
                        RETLW     04H         ; '011'+'01' 일때
                        RETLW     07H         ; '011'+'10' 일때
                        RETLW     10H         ; '011'+'11' 일 때 -- '*' code
;
                        RETLW     0FH         ; '100'+'00' 일때
                        RETLW     0FH         ; '100'+'01' 일때
                        RETLW     0FH         ; '100'+'10' 일때
                        RETLW     0FH         ; '100'+'11' 일때
                        RETLW     02H         ; '101'+'00' 일때
                        RETLW     05H         ; '101'+'01' 일때
                        RETLW     08H         ; '101'+'10' 일때
                        RETLW     00H         ; '101'+'11' 일때
                        RETLW     03H         ; '110'+'00' 일때
                        RETLW     06H         ; '110'+'01' 일때
                        RETLW     09H         ; '110'+'10' 일때
                        RETLW     11H         ; '110'+'11' 일 때 -- '#' code
```

실험 3. 실험 2의 내용에서 Key 번호를 FND에 표시하고 Buzzer On시키는 프로그램 작성 (ex5-4) "0"을 입력시에 표시및 "0" 은 4번째 row, 2번째 col. 요소 : input은 101, output은 11(1110)

Port A : 입력 4개 ➔ Key pad J10의 핀 4, 3, 2, 1에 연결(Intput 으로 사용) 즉, RA3-4번, RA0-3번, RA1-2번, RA2-1번
출력 2개 ➔ RA5를 Buzzer에 연결,
RA5에 FND의 COM1을 연결

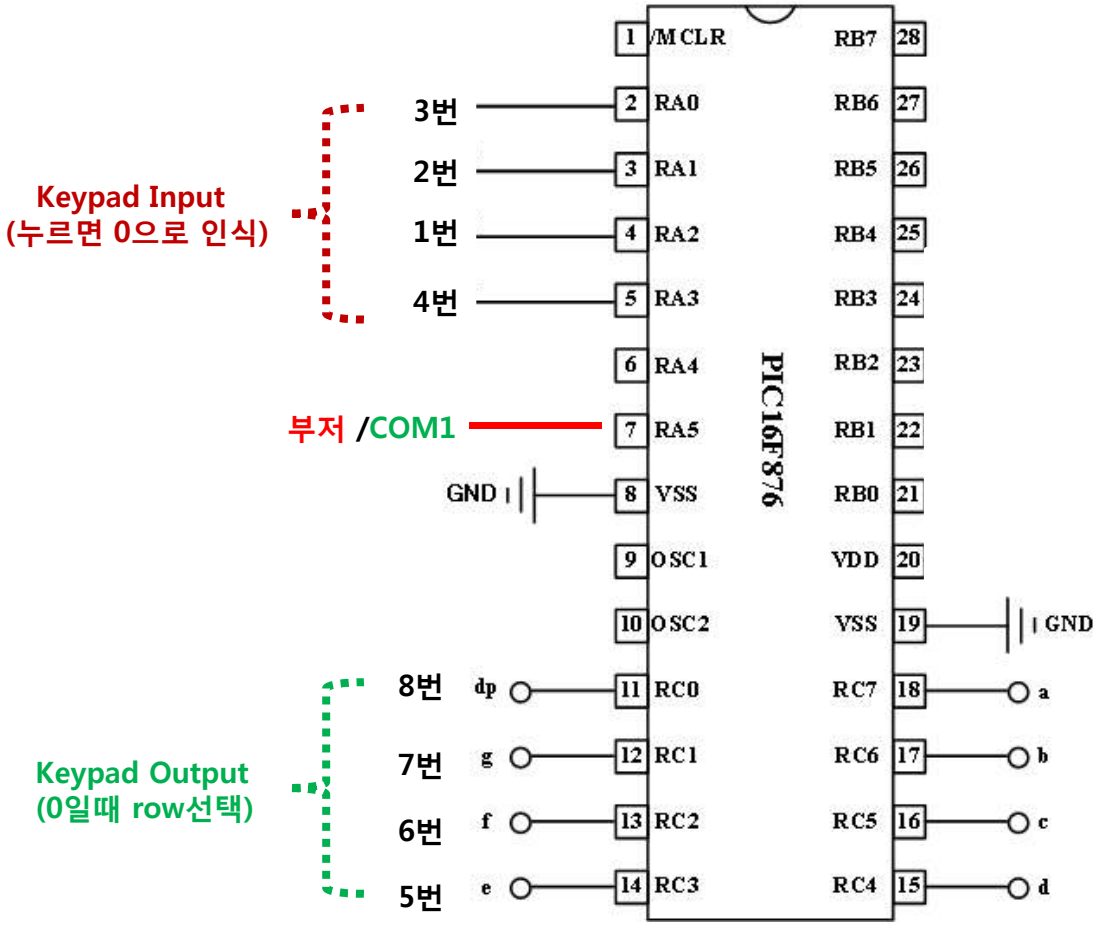
Port C : 출력 5개 ➔ Key pad J10의 핀 5, 6, 7, 8에 연결(Output 으로 사용) 즉, RC3-5번, RC2-6번, RC1-7번, RC0-8번
출력 8개 ➔ FND 8단자 연결

-→ 교과서 112page 예제 6)를 이용하여 작성

* 부가실험 7)번 프로그램 작성

Lab.2 I/O Port를 사용한 입출력-7segment 동작시키기

실험 5-4 수행



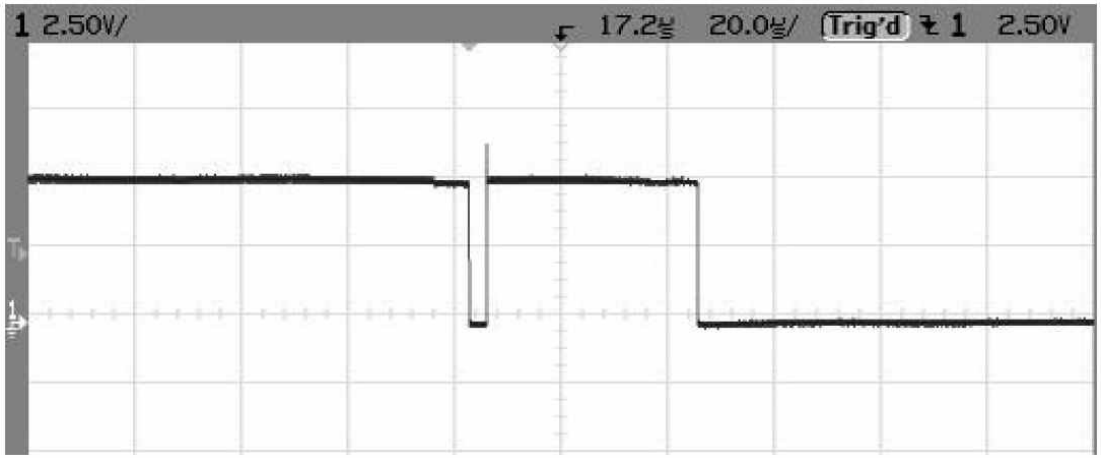
교과서 112page 예제 6) FND 출력

```
LP      CALL        KEY_IN
        MOVF        KEY_DATA, W
        CALL        CONV            ; 숫자를 7-segment 값으로 변경
        MOVWF       PORTC           ; 숫자 값 출력
        MOVLW       B'00100000'     ; COM 1선택 (RA5)
        MOVWF       PORTA ; 위치 결정
        GOTO        LP
```

4. Switch 사용시 주의할 점 - Chattering (Bouncing)

➢ 스위치 바운싱 현상

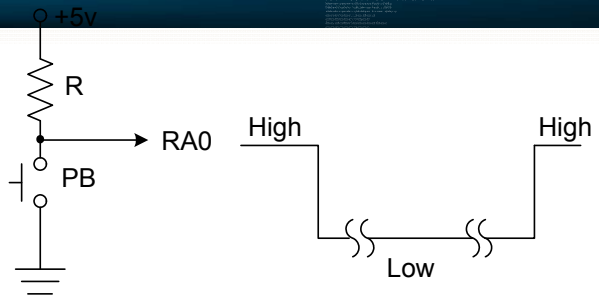
- 스위치를 여러 번 누른 동작으로 오동작할 수 있음
- 스위치의 기계적인 특성으로 발생 → 전압 변화 유발
- 소프트웨어적으로 **스위치 디바운싱** 역할 수행
  - ✓ 스위치 변화를 처음 감지 후, 일정시간 동안 변화를 무시함

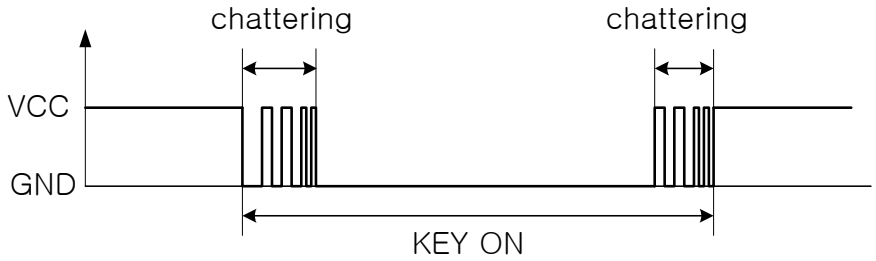


[스위치 바운싱에 의한 전압 변화]

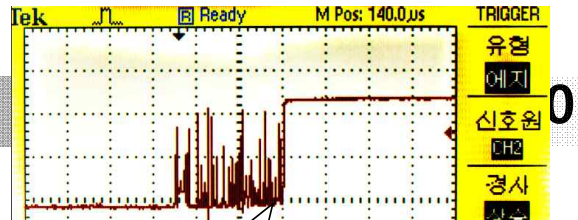
Chattering (Bouncing)

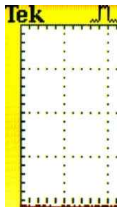
- Chattering 시간 → 5~20msec
- Hardware 에 의한 방법
  - 적분회로 + Schmitt trigger
  - RS Latch
  - 전용 IC(MC14490. 각자 찾아볼 것.)
- Software에 의한 방법
  - Hardware 부담 감소





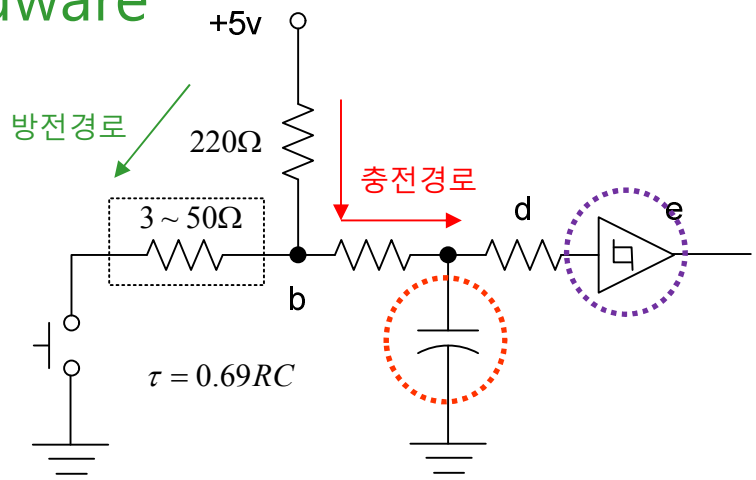
Chattering 제거

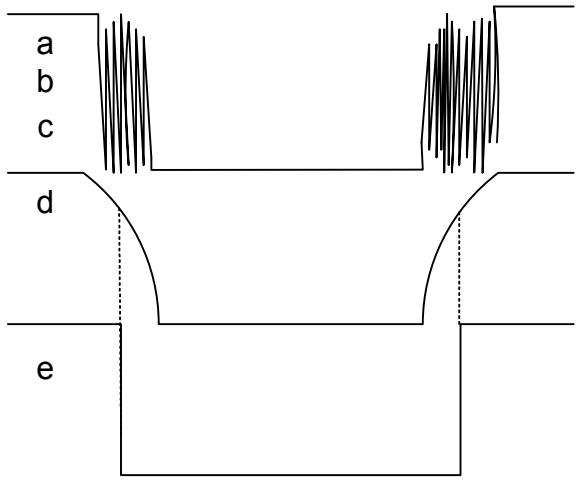




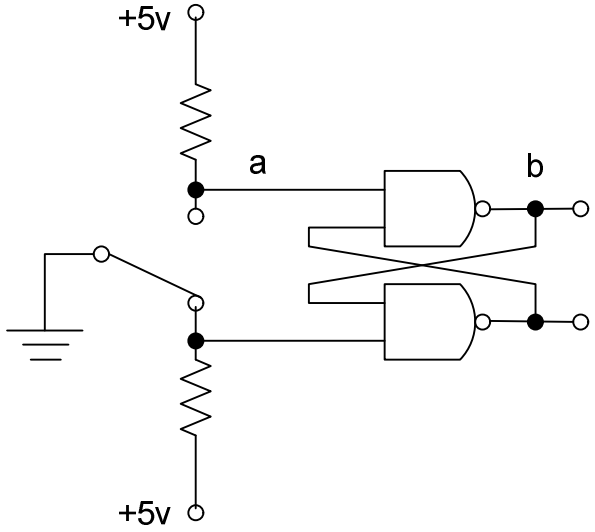
참고: Chattering(De-bouncing)

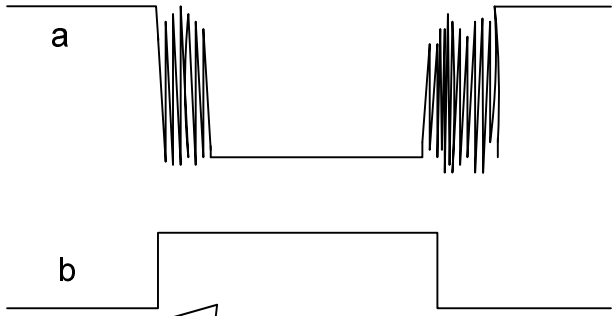
Hardware





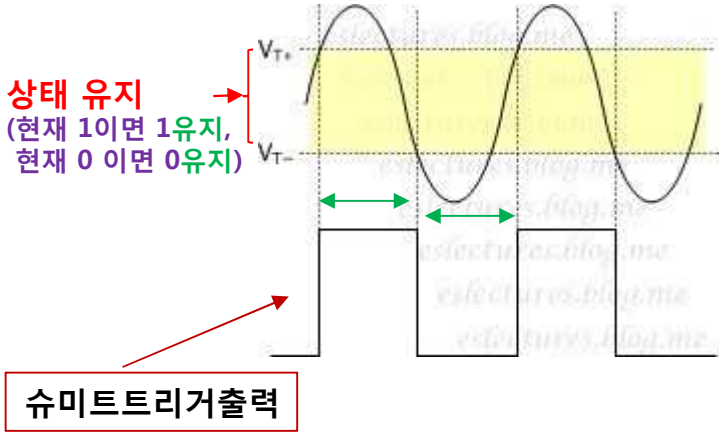
RC + Schmitt trigger



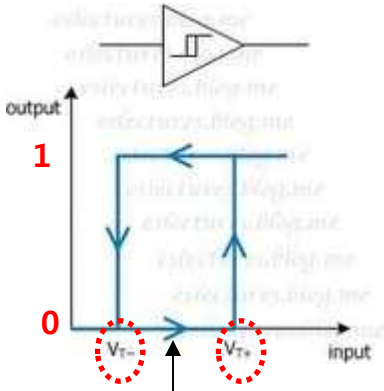


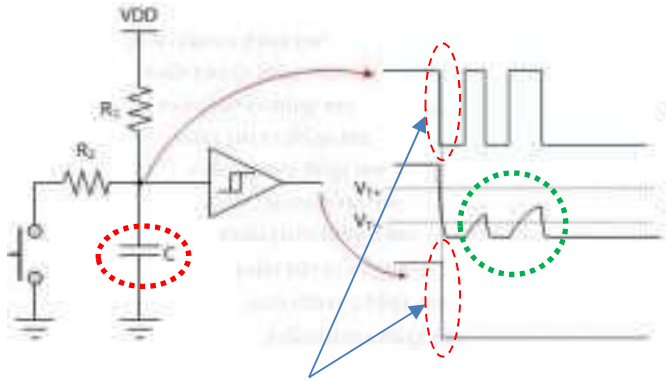
RS Latch

Schmitt trigger



Schmitt trigger





(바운스 현상 발생)

(시정수(R1C)를 조정하여 회로의 반응을 느리게 만들어 신호조정한
슈미트트리거를 이용하여 바운스현상 제거 가능) 시정수조정이 매

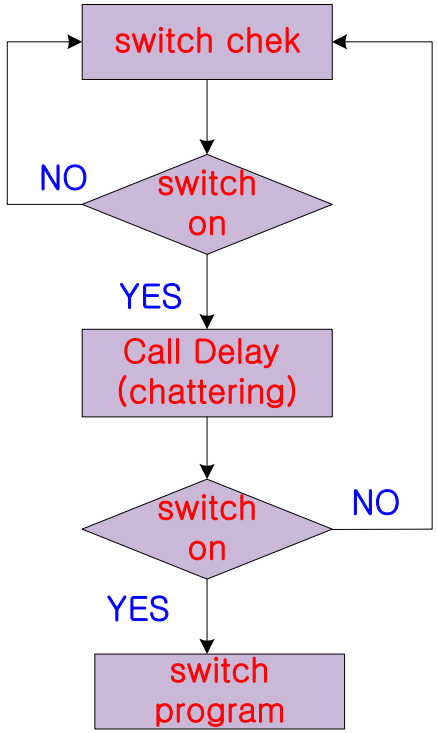
전압이 VT+를 넘으면 논리 1(High), 전압이 감소할때 VT-가 될때까지 1
전압이 VT-가 되면 논리 0(Low) 계속유지

(시간지연 발생, 시정수 크면 시간지연 발생가능,
너무 작으면 바운스에 의한 전압이 VT+ 이상되어 슈미트트리거 사용해도 바운스제거 못할 수 있음
R1이 R2보다 충분히 커야함)

* 히스테리시스(hysteresis): 출력이 현재의 상태에 의존하는 비선형적인 특성(두개의 문턱값을 이용)

Chattering

- Software
  - Delay 시간 이용 -→ Delay 명령어 수행시간으로 인해 프로그램 수행시간 지체되므로 타이머 인터럽트에 의한 방법을 이용하면(타이머로 시간 계수하면 됨) 다른 프로그램의 수행 지연 방지 가능
  - Typematic key(반복:키가 눌러진 회수체크)
  - 일반적으로 key의 scan은 인터럽트를 이용해서 일정시간 마다 체크 함.
  - PC의 경우 key보드와 직렬통신 방식으로 데이터를 주고 받으며, 이 역시도 인터럽트(INT09)를 이용하고 있음.



Thank you! (Grand Canyon, Arizona)

Thank you! (그랜드캐넌의 독수리)