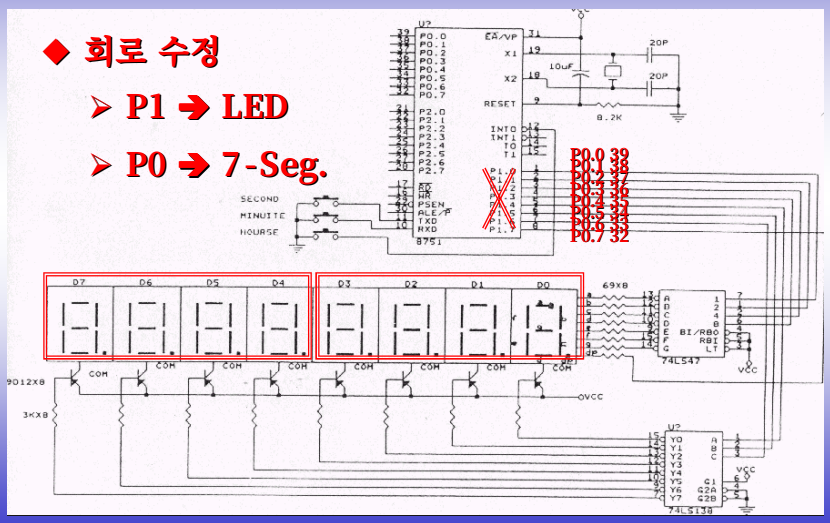
임베디드 기초 프로그래밍
(마이크로프로세서 실습)
ch6. 시계제작

정보네트워크공학전공
이훈재

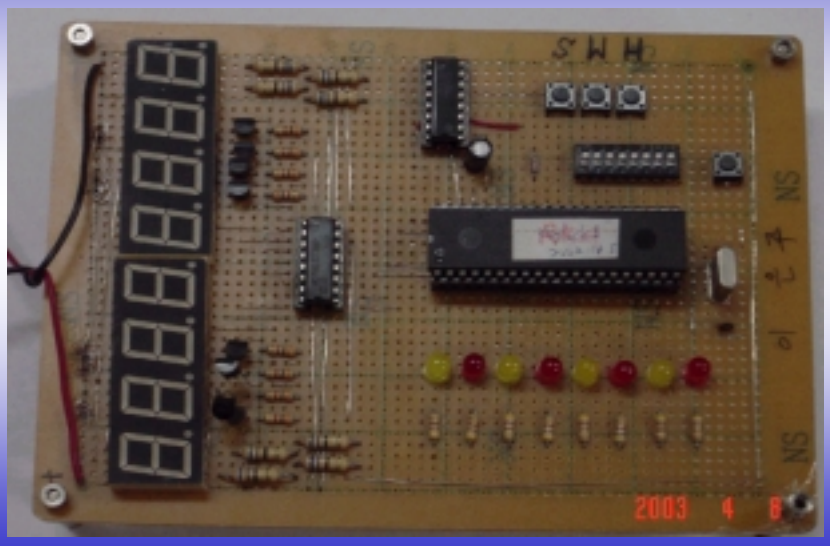
hjlee@dongseo.ac.kr
http://crypto.dongseo.ac.kr
http://kowon.dongseo.ac.kr/~hjlee

6.3 시계 제작
- 실험##. 회로도

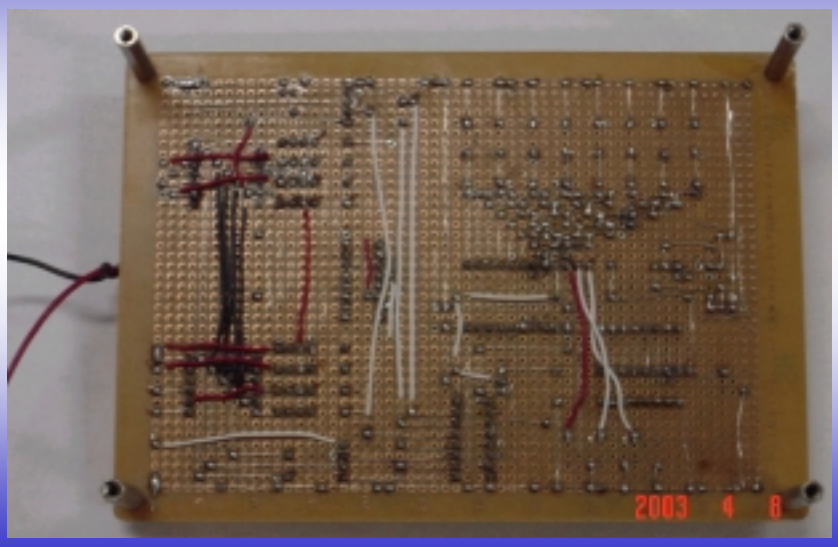
◆ 회로 수정
- P1 ➔ LED
- P0 ➔ 7-Seg.



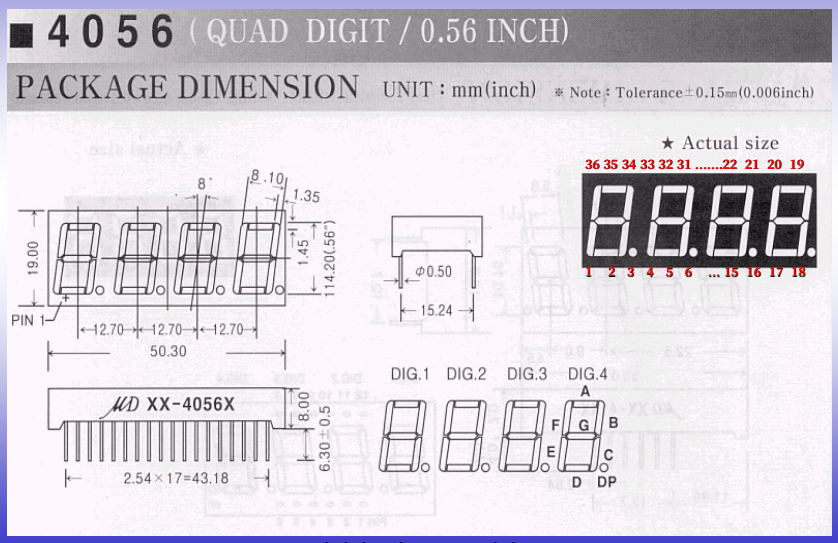
6.3 시계 제작
- 실험##. 6장 Soldering (1)



6.3 시계 제작
- 실험##. 6장 Soldering (2)

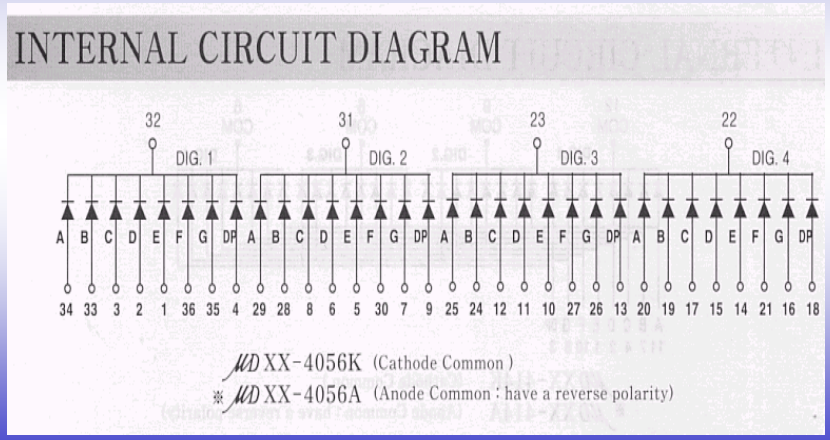


6.3 시계 제작
FND, 4056(QUAD DIGIT)



6.3 시계 제작
FND, 4056(QUAD DIGIT)

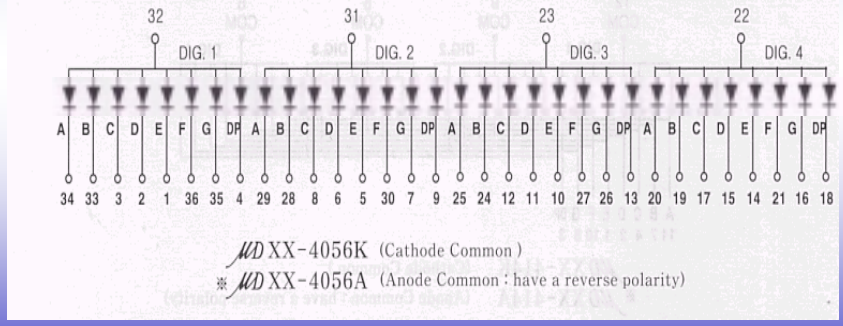
◆ **Common Cathode(참고용)**



6.3 시계 제작
FND, 4056(QUAD DIGIT)

◆ **Common Anode(제작회로)**

INTERNAL CIRCUIT DIAGRAM



MD XX-4056K (Cathode Common)

※ MD XX-4056A (Anode Common : have a reverse polarity)

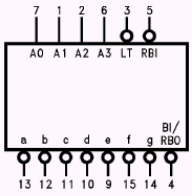
6.3 시계 제작
FND, 4056(QUAD DIGIT)

Device No.		R-4056X*	G-4056X*	SR-4056X*	Y-4056X*	UR-4056X*
CHIP	Materials	GaP/GaP	GaP/GaP	GaAsP/GaP	GaAsP/GaP	GaAlAs
	color	RED	GREEN	ORANGE	YELLOW	ULTRA RED
	λP	700	568	630	589	660
PD (mW)	TOTAL	1,280	1,280	1,280	1,280	1,280
	SEG.	40	40	40	40	40
VF (IF : 20mA)	Typ (V)	2.25	2.25	1.80	2.10	1.80
	Max (V)	2.60	2.60	2.10	2.60	2.00
IR	Max (μA)	10	10	10	10	10
	VR (V)	5	5	5	5	5
IV (IF : 10mA)	Min (μcd)	700	800	4,000	800	10,000
	Typ (μcd)	1,000	1,300	6,000	1,300	12,000

※ REMARK : X* ⇒ A = ANODE COMMON, K = CATHODE COMMON

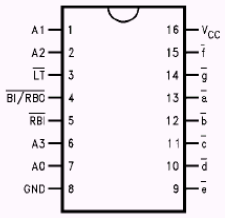
6.3 시계 제작
74LS47(BCD – 7.Segment Dis. Decoder)

Logic Symbol



V_{CC} = Pin 16
GND = Pin 8

Connection Diagram



Pin Descriptions

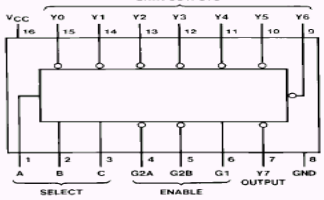
Pin Names	Description
A0–A3	BCD Inputs
$\overline{RBI}$	Ripple Blanking Input (Active LOW)
$\overline{LT}$	Lamp Test Input (Active LOW)
$\overline{BI/RBO}$	Blanking Input (Active LOW) or Ripple Blanking Output (Active LOW)
$\overline{a}$ –$\overline{g}$	Segment Outputs (Active LOW) (Note 1)

Note 1: OC—Open Collector

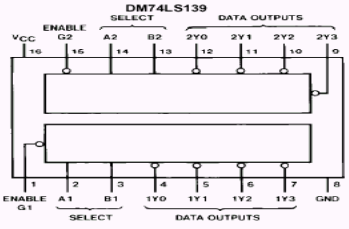
6.3 시계 제작
74LS138 Decoder)

Connection Diagrams

DM74LS138



DM74LS139



Function Tables

DM74LS138

Inputs					Outputs							
Enable		Select										
G1	G2 (Note 1)	C	B	A	YO	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	H	H	H	H	H	L	H	H	H	H
H	L	H	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	H	L	H	H
H	L	H	H	L	H	H	H	H	H	H	L	H
H	L	H	H	H	H	H	H	H	H	H	H	L

DM74LS139

Inputs			Outputs			
Enable	Select					
G	B	A	Y0	Y1	Y2	Y3
H	X	X	H	H	H	H
L	L	L	L	H	H	H
L	L	H	H	L	H	H
L	H	L	H	H	L	H
L	H	H	H	H	H	L

H = HIGH Level
L = LOW Level
X = Don't Care

Note 1: G2 = G2A + G2B

6.3 시계 제작
- ASM Coding(1)

```
DisplayBuffer   equ  30h ; 7-SEGMENT의
                           버퍼 구역 설정
ScanCounter     equ  38h ; SCAN COUNTER
                           를 38H로 설정
OneSecondCounter equ  39h
Hourse          equ  3ah
Minuite         equ  3bh
Second          equ  3ch
;
HourseKey         equ  p3.2
MinuiteKey        equ  p3.1
SecondKey         equ  p3.0


      org  0h ; RESET 후 점프
      jmp  Reset


      org  0bh ; TIMER 0의 인터럽트시
                 TIMER Int로 점프
      jmp  TimerInt
```

```
Reset:
      mov  sp,#70h ; SP를 70H로 지정
      mov  ScanCounter,#0
      mov  Hourse,#12h       ;초기 시간(12:00:00)
      mov  Minuite,#00h      ;초기 분
      mov  Second,#00h       ;초기 초
      mov  tmod,#00000001b; Timer0 & Mode 1
                            설정
      mov  th0,#(65536-4000)/256
      mov  tl0,#(65536-4000) mod 256
      mov  ie,#10000010b    ; Enable Timer 0
                              Intr(T0)
      setb tr0              ; Timer 0 run(Starting)


$1:   call DisplayClockInt0DisplayBuffer
      jmp  $1
```


6.3 시계 제작
- ASM Coding(2)

```
MainLoop:
        jb    SecondKey,CheckMinuiteKey

        mov  Second,#0
        mov  r5,#2
        call  Delay       ; 40 ms
        jnb   SecondKey,$  ; wait key release
;
CheckMinuiteKey:
        jb    MinuteKey,CheckHourseKey
        mov   a,Minuite
        add   a,#1
        da    a
        mov   Minuite,a
        cjne  a,#60h,Notover1
        mov   Minuite,#0
;
Notover1:
        mov   r5,#2
        call  Delay         ; debounce
        jnb   MinuiteKey,$
```

```
CheckHourseKey:
        jb    HourseKey,Mainloop
        mov  a,Hourse
        add   a,#1
        da    a
        mov   Hourse,a
        cjne  a,#24h,Notover2
        mov   Hourse,#0
;
Notover2:
        mov   r5,#2
        call  Delay         ; debounce
        jnb   HourseKey,$
;
;
        jmp   MainLoop
```

6.3 시계 제작
- ASM Coding(3)

```
TimerInt: ; 타이머의 인터럽트 서브 프로그램
          mov  th0,#(65536-4000)/256
          mov  tl0,#(65536-4000) mod 256

          push a
          push psw

          setb rs0
          clr  rs1
          djnz OneSecondCounter,NotOneSecond
          mov OneSecondCounter,#1000/4
          call  Clock
          call  DisplayClockIntoDisplayBuffer
;
NotOneSecond:
          call ScanDisplay

          pop  psw
          pop  a

          reti
```


6.3 시계 제작
- ASM Coding(4)

```
ScanDisplay: 이 프로그램의 주요 부분으로 SCAN 서브 프로그램
             mov  r0,#ScanCounter ; SCANCOUNTER 내용에 +1
             inc  @r0
             cjne @r0,#8,NotOver
             mov  @r0,#0

NotOver:
             mov  a,@r0
             add  a,#DisplayBuffer
             mov  r1,a
             mov  a,@r0
             swap a
             orl  a,@rl
             mov  p0,a ; 데이터를 P0로 출력
             ret ; 복귀
```

6.3 시계 제작
- ASM Coding(5)

```
Clock:  ; 시계의 내용에 1초를 더하는 서브 프로그램
        mov  a,Second
        add  a,#1
        da   a
        mov  Second,a
        cjne a,#60h,NotOverFlow
        mov  Second,#0
;
        mov  a,Minuite
        add  a,#1
        da   a
        mov  Minuite,a
        cjne a,#60h,NotOverFlow
        mov  Minuite,#0
;
        mov  a,Hourse
        add  a,#1
        da   a
        mov  Hourse,a
        cjne a,#24h,NotOverFlow
        mov  Hourse,#0
NotOverFlow:
        ret
```


6.3 시계 제작
- ASM Coding(6)

```
DisplayClockInt0DisplayBuffer: 시계의 내용을
;디스플레이 버퍼 구역내로 전환시켜 디스플레이 한
다.
        mov  r3,$3
        mov  r1,#DisplayBuffer
Loop:   mov  a,Second
        mov  b,#10h
        div  ab
        mov  @r1,b
        inc  r1
        mov  @r1,a
;
        inc  r1
        mov  a,Minuite
        mov  b,#10h
        div  ab
        mov  @r1,a
        inc  r1
        mov  @r1,a
;
        inc  r1
        mov  a,Hourse
        mov  b,#10h
        div  ab
        mov  @r1,b
        inc  r1
        mov  @r1,a
        ret
```

```
Delay:
        mov   r6,#40
Del:    mov   r7,#249
        djnz  r7,$
        djnz  r6,Del
        djnz  r5,Delay
        ret
;
;
;
        end
```