



논리 회로 실험 부호기와 복호기

2005. 5. 3.

부호기와 복호기란?
이론
실험 내용





Encoder & Decoder



❖ 개요

- ❖ 서로 다른 부호 간의 변환에 사용되는 것으로 디지털 신호를 압축하거나 전송 시 깨지지 않도록 바꾸는 등 여러 가지 목적에 의해 부호화 라는 장치와 부호화되어 전송되어온 신호를 다시 원래의 디지털 신호로 복호하는 장치들을 말한다.
- ❖ CODEC(enCOder DECoder)이라고도 한다.
 - ❖ 기타
 - ❖ 10진 to 2진(BCD) 부호기
 - ❖ BCD-7 Segment 복호기
 - ❖ 2 to 8 복호기,
 - ❖ 동영상 & 이미지 & 음악 관련
 - ❖ Mpeg2, WMA, MP3, OGG, Divx, MPEG4 Encoder & Decoder
 - ❖ 통신 관련
 - ❖ Manchester 인코딩 & 디코딩
 - ❖ Morsecode 인코딩 & 디코딩





Manchester Code



❖ Manchester Code

- ❖ 통신 선로를 통하여 신호를 전송 시에, 111 이나 0000과 같이 같은 신호를 연이어서 주게 되면, 신호 전송 시의 지연이나 모호함으로 인하여, 신호의 동기가 깨어질 수 있다. 이를 방지하고자, 1은 10으로 0은 01로 바꾸어서 보낸다. 이를 통하여 비트 경계의 모호함이 보완될 수 있는데, 맨체스터 코드를 Bitphase Code라고도 한다.
- ❖ IC 형태로 구성된 인코더 & 디코더 칩도 있으며, 때로는 어셈블리나 C언어 프로그램으로 작성하여 마이크로프로세서에 내장하기도 한다.

❖ 11001111 → 1010010110101010 ↔ 타 단말로 감.

❖ 4800 bps 9600 bps 주로 무선 통신



Manchester Encoder by C



```
❖ struct BitField{
❖     int bit0:1;
❖     int bit1:1;
❖     int bit2:1;
❖     int bit3:1;
❖     int bit4:1;
❖     int bit5:1;
❖     int bit6:1;
❖     int bit7:1;
❖ };

❖ void SendManchesterCharacter(char c)
❖ {
❖     union BitSend send[2];
❖
❖     if(c&0x80){
❖         send[0].SendBit.bit7 = 0;
❖         send[0].SendBit.bit6 = 1;
❖     }else{
❖         send[0].SendBit.bit7 = 1;
❖         send[0].SendBit.bit6 = 0;
❖     }
❖
❖     if(c&0x40){
❖         send[0].SendBit.bit5 = 0;
❖         send[0].SendBit.bit4 = 1;
❖     }else{
❖         send[0].SendBit.bit5 = 1;
❖         send[0].SendBit.bit4 = 0;
❖     }
❖
❖     if(c&0x20){
❖         send[0].SendBit.bit3 = 0;
❖         send[0].SendBit.bit2 = 1;
❖     }
❖
❖     else{
❖         send[0].SendBit.bit3 = 1;
❖         send[0].SendBit.bit2 = 0;
❖     }
❖
❖     if(c&0x10){
❖         send[0].SendBit.bit1 = 0;
❖         send[0].SendBit.bit0 = 1;
❖     }else{
❖         send[0].SendBit.bit1 = 1;
❖         send[0].SendBit.bit0 = 0;
❖     }
❖
❖     if(c&0x08){
❖         send[1].SendBit.bit7 = 0;
❖         send[1].SendBit.bit6 = 1;
❖     }else{
❖         send[1].SendBit.bit7 = 1;
❖         send[1].SendBit.bit6 = 0;
❖     }
❖
❖     if(c&0x04){
❖         send[1].SendBit.bit5 = 0;
❖         send[1].SendBit.bit4 = 1;
❖     }else{
❖         send[1].SendBit.bit5 = 1;
❖         send[1].SendBit.bit4 = 0;
❖     }
❖
❖     if(c&0x02){
❖         send[1].SendBit.bit3 = 0;
❖         send[1].SendBit.bit2 = 1;
❖     }
❖ }
```



Manchester Encoder by C



```
❖         else{  
❖             send[1].SendBit.bit3 = 1;  
❖             send[1].SendBit.bit2 = 0;  
❖         }  
❖  
❖         if(c&0x01){  
❖             send[1].SendBit.bit1 = 0;  
❖             send[1].SendBit.bit0 = 1;  
❖         }else{  
❖             send[1].SendBit.bit1 = 1;  
❖             send[1].SendBit.bit0 = 0;  
❖         }  
❖  
❖  
❖         Uart_Printf("%c",send[0].SendChar);  
❖         Uart_Printf("%c",send[1].SendChar);  
❖     }
```





Manchester Decoder by C



```
❖ void ReceiveManchesterCharacter(void)
❖ {
❖     unsigned char send[3], temp;
❖     int i, error=0;
❖
❖     send[0] = buffer[curptr++];
❖     send[1] = buffer[curptr++];
❖
❖     temp=send[1];
❖     for(i=0;i<4;i++){
❖         send[2]=send[2]<<1;
❖         if((temp&0x40)
❖         && !(temp&0x80)){
❖
❖             send[2]=send[2]|1;
❖             }else if(!(temp&0x40)
❖             && (temp&0x80)){
❖                 }else{
❖                     error=1;
❖                 }
❖                 temp=temp<<2;
❖             }
❖         }
❖     }
```

```
❖     temp=send[0];
❖     for(i=0;i<4;i++){
❖         send[2]=send[2]<<1;
❖         if((temp&0x40)
❖         && !(temp&0x80)){
❖
❖             send[2]=send[2]|1;
❖             }else if(!(temp&0x40) &&
❖             (temp&0x80)){
❖                 }else{
❖                     error=1;
❖                 }
❖                 temp=temp<<2;
❖             }
❖         }
❖     SendCharacter(send[2]);
❖ }
```



Manchester Encoder by ASM



```

;-----
;   MANCHESTER CODE 1 BYTE TRANSMIT
;-----
MTXBYTE:
    MOV     R6,#08      ;BIT COUNT #08
    MOV     A,TXBF      ;TX BUFF LOAD
    RRC     A
    MOV     TXBF,A

MTXB1:      MOV         TXDM,C      ; Manchester coding...
    MOV     D_reg,#43
    CALL    _delay

    CPL     TXDM              ; Manchester coding...
    DJNZ    R6,MTXB2
    JMP     MTXB3

MTXB2:      MOV     D_reg,#40
    CALL    _delay
    MOV     A,TXBF      ;TX BUFF LOAD
    RRC     A
    MOV     TXBF,A
    JMP     MTXB1

MTXB3:      MOV         D_reg,#35
    CALL    _delay
    RET
```



Manchester Decoder by ASM



```
-----  
; MANCHESTER CODE 1 BYTE RECEIVER  
-----  
MRXBYTE:  
CLR RXER ;RX ERROR FLAG  
  
MOV R6,#00 ;256MC START BIT DELAY  
MOV RXBF,#00  
  
MRXB1: JNB RXDM,MRXB2 ;START BIT CHECK  
DJNZ R6,MRXB1 ;2MC  
SETB RXER  
RET  
  
MRXB2:  
MOV D_reg,#22 ;delay until front of edge  
CALL _delay  
SETB P3.3  
CALL EDGR ;EDGE DETECTOR  
CLR P3.3  
JNB RXER,MRXB3  
JNB RXDM,$ ;for seek NEXT start bit  
RET  
  
MRXB3: JNC MRXB4 ;START CODE  
SETB RXER  
RET  
  
MRXB4: MOV R6,#8 ;8  
  
MOV D_reg,#MCLK+10 ;delay until front of edge  
CALL _delay
```

```
MRXB5: SETB P3.4  
CALL EDGR  
CLR P3.4  
JNB RXER,MRXB6  
RET  
  
MRXB6: MOV A,RXBF ;RX BUFF  
RRC A  
MOV RXBF,A  
DJNZ R6,MRXB7  
  
MOV D_reg,#MCLK+9 ;delay until front of edge  
CALL _delay  
NOP  
CALL EDGR  
NOP  
JNC MRXE  
SETB RXER  
MRXE: RET  
  
MRXB7:  
MOV D_reg,#MCLK+9 ;delay until front of edge  
CALL _delay  
JMP MRXB5  
  
-----  
EDGR: NOP  
; SETB P1.4 ;test  
CLR RXER ;46MC EDGE CHECK  
  
MOV R7,#13 ;9  
  
JNB RXDM,EDGRH ;EDGE LOW -->> HIGH 검출 시작
```



Manchester Decoder by ASM



```
❖ ;-----;
❖ EDGRL: NOP
❖ EDGRL1: JNB  RXDM,EDGRL3  ;EDGE HIGH -->> LOW 검출 시작
❖     DJNZ  R7,EDGRL1
❖     SETB  RXER          ;EDGE DETECT 실패
❖     NOP
❖     NOP
❖     RET

❖ EDGRL2: NOP
❖     NOP
❖ EDGRL3: DJNZ  R7,EDGRL2
❖     SETB  C            ;EDGE 검출 성공
❖     NOP
❖ ;     SETB  P1.4                ;test
❖     NOP
❖     RET

❖ ;-----;
❖ EDGRH: NOP
❖ EDGRH1: JB   RXDM,EDGRH3  ;EDGE LOW -->> HIGH 검출 시작
❖     DJNZ  R7,EDGRH1
❖     SETB  RXER          ;EDGE DETECT 실패
❖     NOP
❖     NOP
❖     RET

❖ EDGRH2: NOP
❖     NOP
❖ EDGRH3: DJNZ  R7,EDGRH2
❖     CLR   C            ;EDGE 검출 성공
❖     NOP
❖ ;     CLR   P1.4
❖     NOP
❖     RET
```

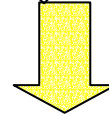


부호기 설계 방법

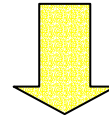


설계 절차

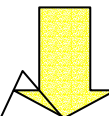
원하는 부호기의 진리표 작성



출력 논리 식을 구함



카르노 맵 또는 계산을 통한 수식 간략화



회로도 작성

류 대 우

davidryu@newtc.co.kr

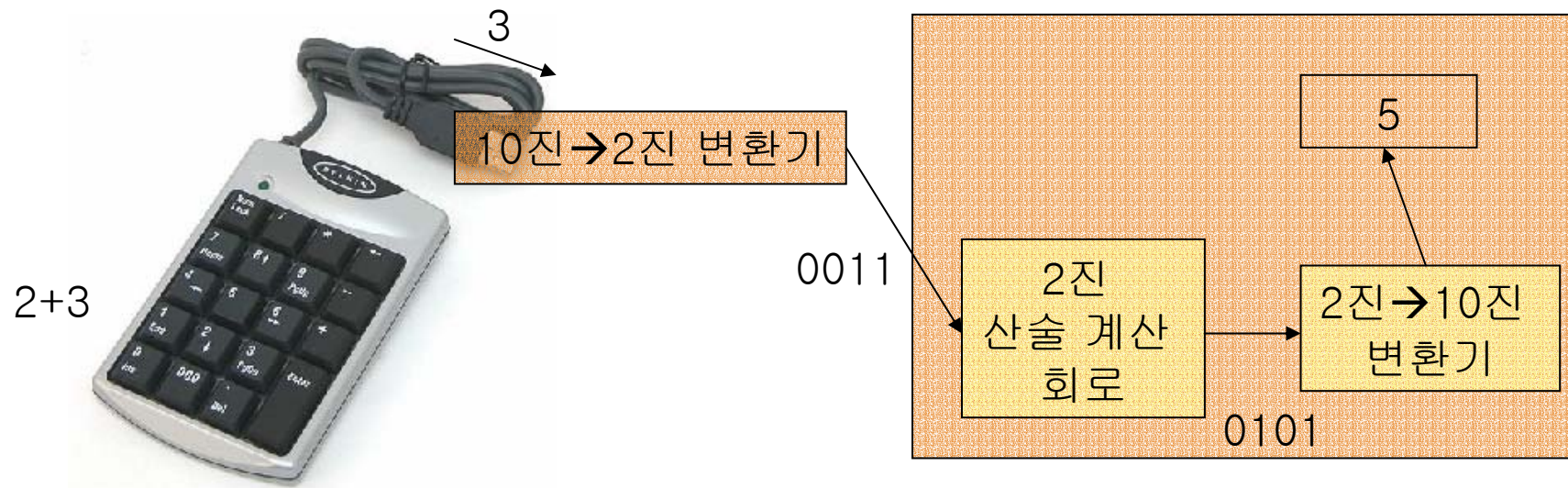


부호기 설계 방법



❖ 2진 $\leftrightarrow$ 10진 변환

- ❖ 디지털 회로들에서 주로 사용되는 것은 2진수이고,
- ❖ 사람들은 10진수를 사용하므로, 2진수 $\leftrightarrow$ 10진수로 바꾸는 것이 중요함.
- ❖ 실제로 하드웨어 회로들 또는 소프트웨어 프로그램으로 대부분의 모든 디지털 시스템에서 이러한 변환을 사용한다.





Encoder



❖ 10진 → 2진 변환기 (Encoder)

$$N_{10} = d_{p-1}d_{p-2}d \dots d_m d_{m-1} \dots d_1d_0$$

$$\text{Ex) } N_{10} = 249_{10}$$

i=0	$N_{10} = 249 / 2 = 124$	$b_0 = 1$
i=1	$N_{10} = 124 / 2 = 62$	$b_0 = 0$
i=2	$N_{10} = 62 / 2 = 31$	$b_0 = 0$
i=3	$N_{10} = 31 / 2 = 15$	$b_0 = 1$
i=4	$N_{10} = 15 / 2 = 7$	$b_0 = 1$
i=5	$N_{10} = 7 / 2 = 3$	$b_0 = 1$
i=6	$N_{10} = 3 / 2 = 1$	$b_0 = 1$
i=7	$N_{10} = 1 / 2 = 0$	$b_0 = 1$

$$N_2 = b_7b_6b_5b_4b_3b_2b_1b_0 = 11111001_2$$

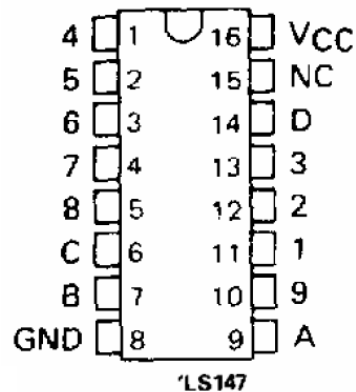


10개 선 → 4개 선 Encoder

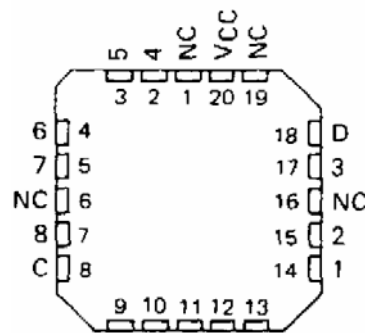


❖ 10개 선 → 4개 선 우선순위 변환기 진리표

❖ SN74147



LS147

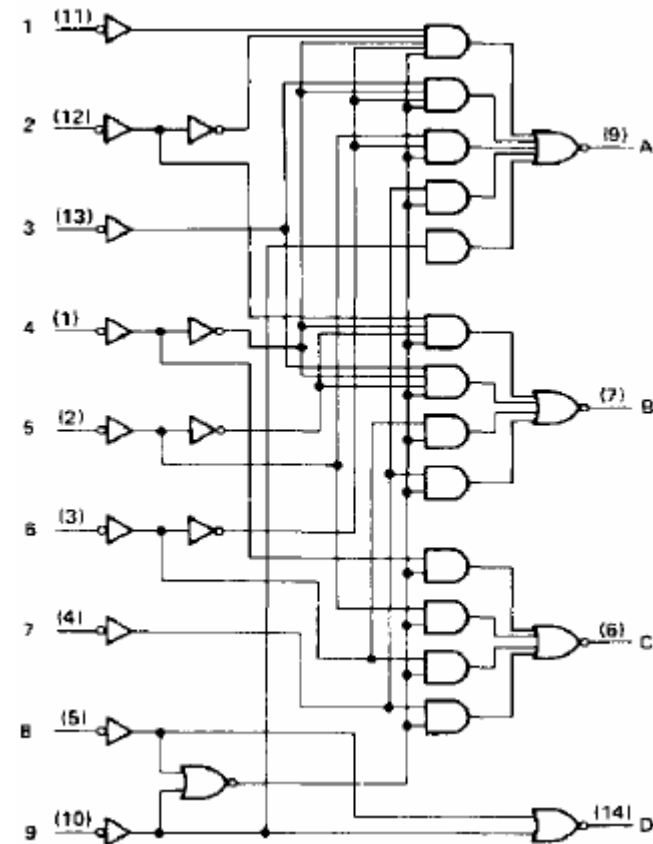
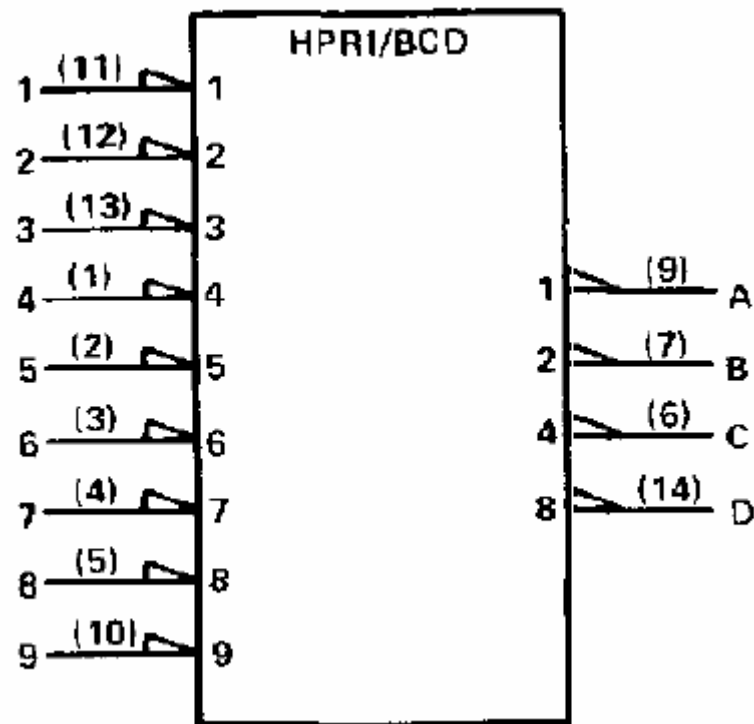


SN74147

입 력									출 력			
1	2	3	4	5	6	7	8	9	D	C	B	A
1	1	1	1	1	1	1	1	1	1	1	1	1
X	X	X	X	X	X	X	X	0	0	1	1	0
X	X	X	X	X	X	X	0	1	0	1	1	1
X	X	X	X	X	X	0	1	1	1	0	0	0
X	X	X	X	X	0	1	1	1	1	0	0	1
X	X	X	X	0	1	1	1	1	1	0	1	0
X	X	X	0	1	1	1	1	1	1	0	1	1
X	X	0	1	1	1	1	1	1	1	1	0	0
X	0	1	1	1	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1	1	1	0



Encoder





2진 10진 Decoder



❖ 2진(BCD) → 10진 변환기 (Decoder)

$$N = \sum_{i=0}^{n-1} b_i \times 2^i$$

$$\begin{aligned} \text{Ex) } 11101_2 &= 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 \\ &= 16 + 8 + 4 + 0 + 1 \\ &= 29 \end{aligned}$$



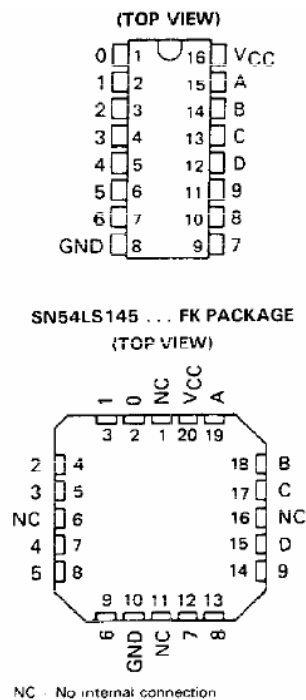


Decoder



❖ 2진 → 10진 디코더 진리표

❖ SN74145

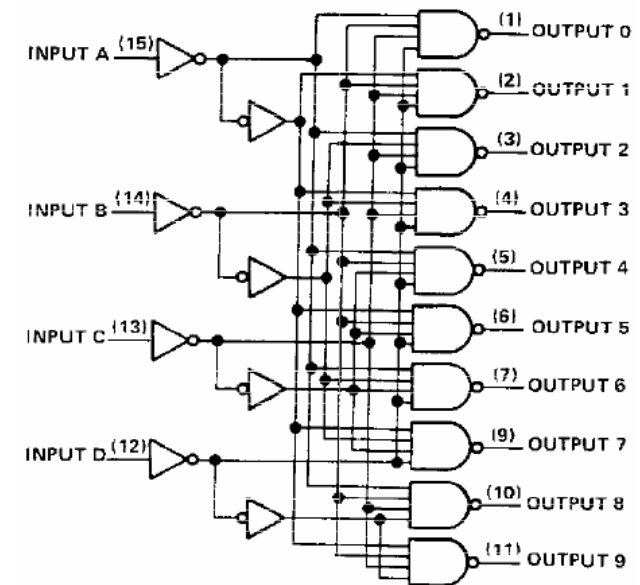


SN74145

FUNCTION TABLE												
NO.	입력들				출력들							
	D	C	B	A	0	1	2	3	4	5	6	7
0	L	L	L	L	L	H	H	H	H	H	H	H
1	L	L	L	H	H	L	H	H	H	H	H	H
2	L	L	H	L	H	H	L	H	H	H	H	H
3	L	L	H	H	H	H	H	L	H	H	H	H
4	L	H	L	L	H	H	H	H	L	H	H	H
5	L	H	L	H	H	H	H	H	H	L	H	H
6	L	H	H	L	H	H	H	H	H	H	L	H
7	L	H	H	H	H	H	H	H	H	H	H	L
8	H	L	L	L	H	H	H	H	H	H	H	L
9	H	L	L	H	H	H	H	H	H	H	H	L
비 정 확	H	L	H	L	H	H	H	H	H	H	H	H
	H	L	H	H	H	H	H	H	H	H	H	H
	H	H	L	L	H	H	H	H	H	H	H	H
	H	H	L	H	H	H	H	H	H	H	H	H

H = high level (off), L = low level (on)

logic diagram





Decoder



recommended operating conditions

	SN54145			SN74145			UNIT
	MIN	NOM	MAX	MIN	NOM	MAX	
Supply voltage, V_{CC}	4.5	5	5.5	4.75	5	5.25	V
Off-state output voltage, $V_{O(off)}$			15			15	V
Operating free-air temperature, T_A	-55		125	0		70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

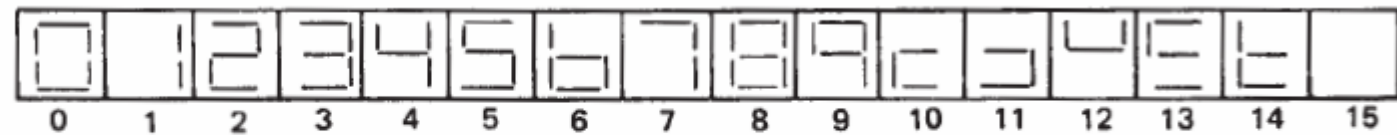
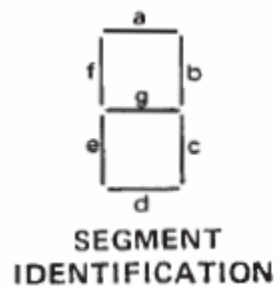
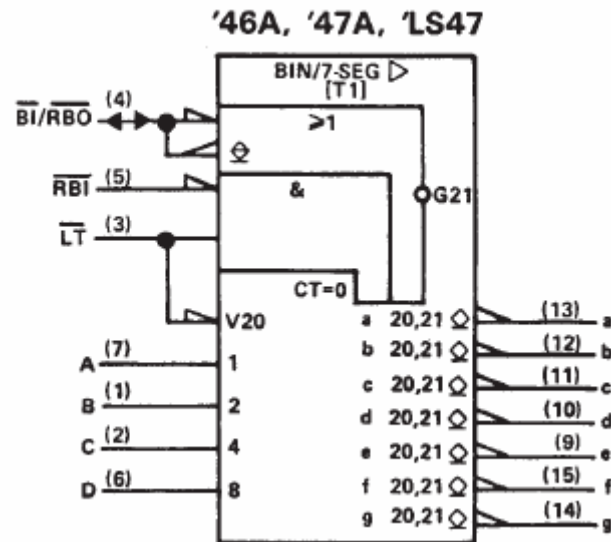
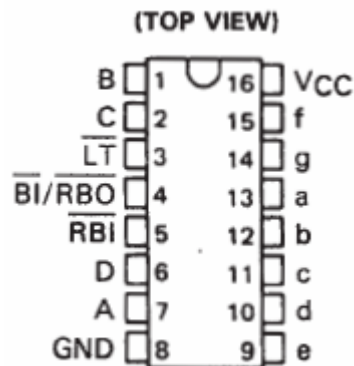
PARAMETER		TEST CONDITIONS ¹	MIN	TYP [‡]	MAX	UNIT
V_{IH}	High-level input voltage		2			V
V_{IL}	Low-level input voltage				0.8	V
V_{IK}	Input clamp voltage	$V_{CC} = \text{MIN.}$, $I_I = -12 \text{ mA}$			-1.5	V
$I_{O(off)}$	Off-state output current	$V_{CC} = \text{MIN.}$, $V_{IH} = 2 \text{ V}$, $V_{IL} = 0.8 \text{ V}$, $V_{O(off)} = 15 \text{ V}$			250	μA
$V_{O(on)}$	On-state output voltage	$V_{CC} = \text{MIN.}$, $V_{IH} = 2 \text{ V}$, $V_{IL} = 0.8 \text{ V}$	$I_{O(on)} = 80 \text{ mA}$		0.5	V
			$I_{O(on)} = 20 \text{ mA}$		0.4	
I_I	Input current at maximum input voltage	$V_{CC} = \text{MAX.}$, $V_I = 5.5 \text{ V}$			1	mA
I_{IH}	High-level input current	$V_{CC} = \text{MAX.}$, $V_I = 2.4 \text{ V}$			40	μA
I_{IL}	Low-level input current	$V_{CC} = \text{MAX.}$, $V_I = 0.4 \text{ V}$			-1.6	mA
I_{CC}	Supply current	$V_{CC} = \text{MAX.}$, See Note 2	SN54145	43	62	mA
			SN74145	43	70	



2진 → 7 세그먼트 Decoder



❖ 2진(BCD) → 7 세그먼트 복호기



NUMERICAL DESIGNATIONS AND RESULTANT DISPLAYS



Decoder



❖ 7447 진리표

'46A, '47A, 'LS47 FUNCTION TABLE (T1)

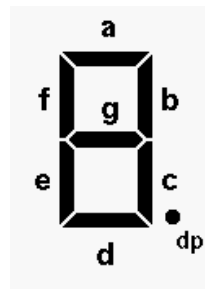
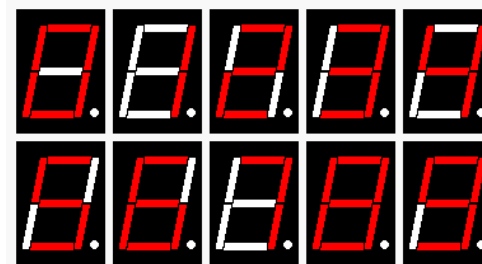
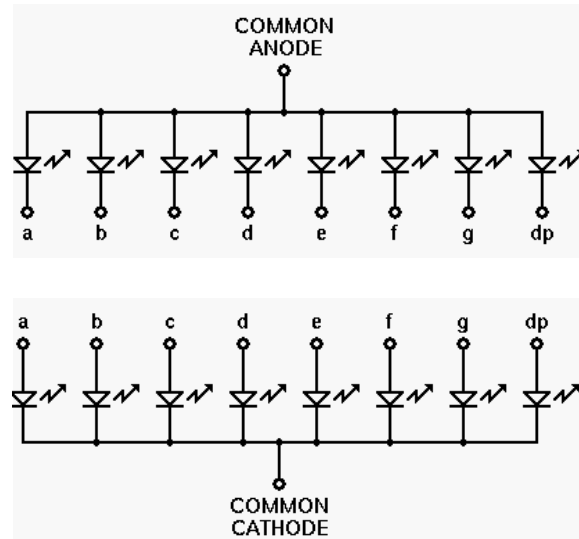
DECIMAL OR FUNCTION	INPUTS						$\overline{\text{BI}}/\overline{\text{RBO}}^\dagger$	OUTPUTS								NOTE
	$\overline{\text{LT}}$	$\overline{\text{RBI}}$	D	C	B	A		a	b	c	d	e	f	g		
0	H	H	L	L	L	L	H	ON	ON	ON	ON	ON	ON	OFF	1	
1	H	X	L	L	L	H	H	OFF	ON	ON	OFF	OFF	OFF	OFF		
2	H	X	L	L	H	L	H	ON	ON	OFF	ON	ON	OFF	ON		
3	H	X	L	L	H	H	H	ON	ON	ON	ON	OFF	OFF	ON		
4	H	X	L	H	L	L	H	OFF	ON	ON	OFF	OFF	ON	ON		
5	H	X	L	H	L	H	H	ON	OFF	ON	ON	OFF	ON	ON		
6	H	X	L	H	H	L	H	OFF	OFF	ON	ON	ON	ON	ON		
7	H	X	L	H	H	H	H	ON	ON	ON	OFF	OFF	OFF	OFF		
8	H	X	H	L	L	L	H	ON	ON	ON	ON	ON	ON	ON		
9	H	X	H	L	L	H	H	ON	ON	ON	OFF	OFF	ON	ON		
10	H	X	H	L	H	L	H	OFF	OFF	OFF	ON	ON	OFF	ON		
11	H	X	H	L	H	H	H	OFF	OFF	ON	ON	OFF	OFF	ON		
12	H	X	H	H	L	L	H	OFF	ON	OFF	OFF	OFF	ON	ON		
13	H	X	H	H	L	H	H	ON	OFF	OFF	ON	OFF	ON	ON		
14	H	X	H	H	H	L	H	OFF	OFF	OFF	ON	ON	ON	ON		
15	H	X	H	H	H	H	H	OFF	OFF	OFF	OFF	OFF	OFF	OFF		
BI	X	X	X	X	X	X	L	OFF	OFF	OFF	OFF	OFF	OFF	OFF	2	
RBI	H	L	L	L	L	L	L	OFF	OFF	OFF	OFF	OFF	OFF	OFF	3	
LT	L	X	X	X	X	X	H	ON	ON	ON	ON	ON	ON	ON	4	



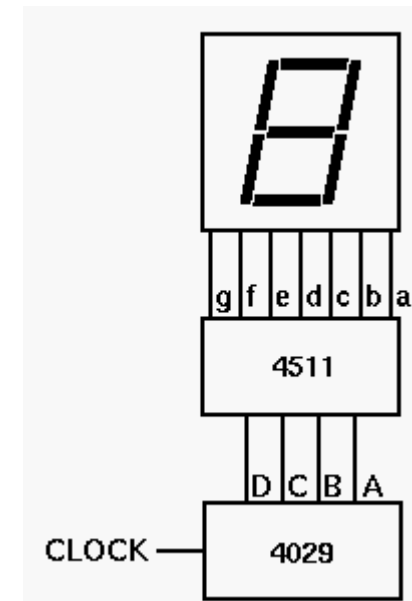
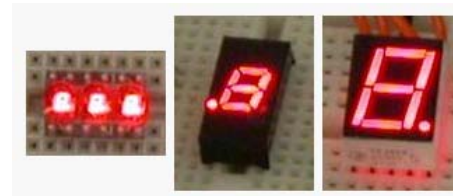
7 Segment



❖ Seven Segment



K : Common Cathode

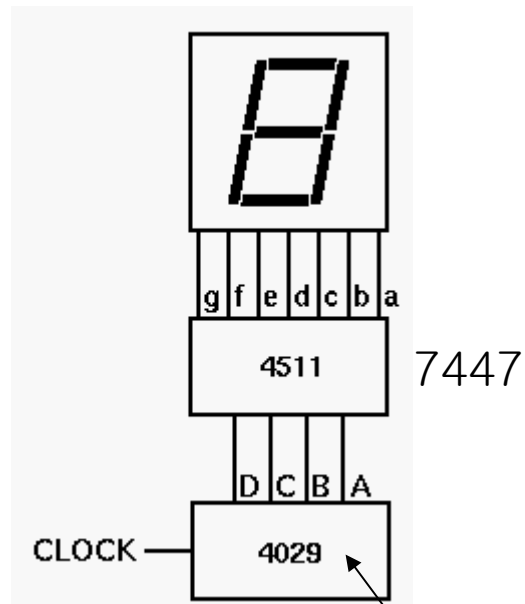




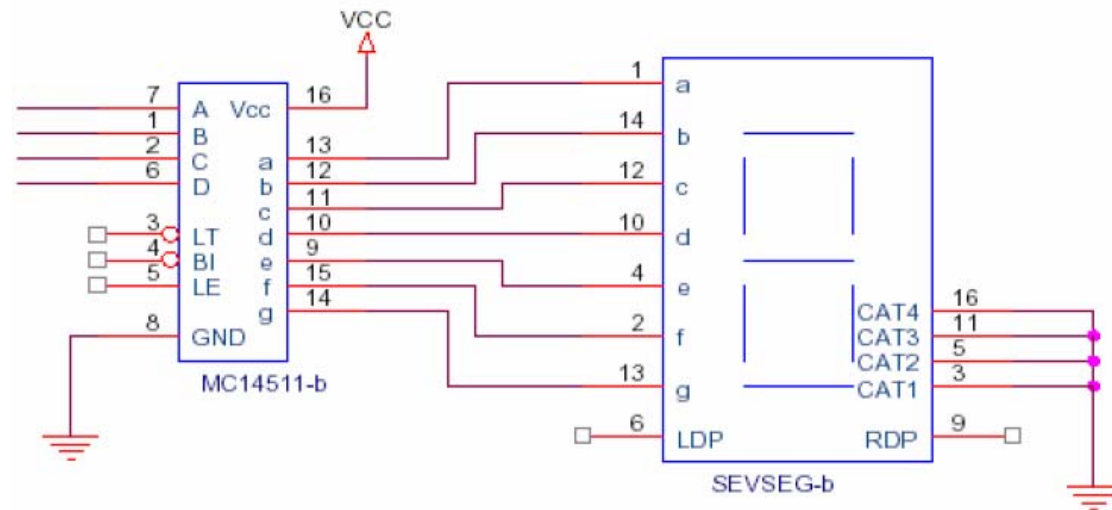
7 Segment



❖ 세븐 세그먼트의 사용 실험



(a) Up/Down Converter



(b)

실험 결과 보고 사항 및 차기 과제

•실험 결과 보고 사항

- 74147에 전원을 인가하고, D1~D9까지 차례로 하나만 0을 입력하고, 나머지는 1을 입력하여 DCBA 출력 값을 적으시오.
- 74147에 전원을 인가하고, 2진수를 입력하여 출력 값을 적으시오.
- 7447에 세븐 세그먼트를 연결하고, 입력을 2진수로 변화시킨 후, 값을 표 9-3에 완성하시오.

•리포트

- 비동기 직렬통신(Serial Communication)에 관하여 아는 데로 조사하시오.
(10점 숙제)
- AVR 및 8051 MCU에 관하여, 아는 데로 다 조사하시오.(30점 숙제)
→워드로 정리 하시오.
 - 내부구조,핀기능,메모리, 컴파일러, 타이머, 인터럽트, 직렬통신
- A/D & D/A 컨버터에 관하여 아는 데로 조사하시오.(워드로 정리)