

# 디지털논리회로

이론, 실습, 시뮬레이션

(Problem Solutions of Chapter 3)

### 1. 10진수를 BCD로 변환

- ① 104: 000100000100
- ② 275: 001001110101
- ③ 369: 001101101001
- ④ 547: 010101000111
- ⑤ 1052: 0001000001010010
- ⑥ 2639: 0010011000111001

### 2. BCD 코드를 10진수로 변환

- ① 10000000: 80
- ② 1000110111: 237
- ③ 1101000110: 346
- ④ 11101010100: 754
- ⑤ 1011010000011: 1683
- ⑥ 0110011001100111: 6667

### 3. BCD 덧셈

- ①  $4+3=7$
- ②  $6+1=7$
- ③  $28+23=4B$ ,  $4B+06=51$
- ④  $65+58=BD$ ,  $BD+66=3112$
- ⑤  $143+276=3B9$ ,  $3B9+060=419$
- ⑥  $295+157=3EC$ ,  $3EC+066=452$

### 4. 2421코드 변환

3	8	6	4
0011 or 1001	1110	1100 or 0110	0100 or 1010

### 5. 자기 보수적인 성질을 가진 코드

3초과 코드, 2421 코드, 84-2-1 코드, 51111 코드

### 6. 4311 코드변환 방법

10진수	4311 코드		
0	0000		
1	0001	0010	
2	0011		
3	0100		
4	0101	0110	1000
5	1010	1001	0111
6	1011		
7	1100		
8	1110	1101	
9	1111		

∴ 자기보수 성질을 갖는다.

### 7. 2진 코드와 그레이 코드 변환

- ①  $1011_{(2)} \rightarrow 1110_{(g)}$
- ②  $0111_{(2)} \rightarrow 0100_{(g)}$
- ③  $1001_{(2)} \rightarrow 1101_{(g)}$
- ④  $1000_{(2)} \rightarrow 1100_{(g)}$
- ⑤  $10010101010_{(2)} \rightarrow 11011111111_{(g)}$
- ⑥  $00100001010101_{(2)} \rightarrow 00110001111111_{(g)}$
- ⑦  $1010101010101010_{(2)} \rightarrow 1111111111111111_{(g)}$

- ⑧  $10110101_{(2)} \rightarrow 11101111_{(g)}$
- ⑨  $00101100_{(2)} \rightarrow 00111010_{(g)}$
- ⑩  $1011001101011010_{(2)} \rightarrow 1110101011110111_{(g)}$
- ⑪  $0011111000010010_{(2)} \rightarrow 0010000100011011_{(g)}$
- ⑫  $1110_{(g)} \rightarrow 1011_{(2)}$
- ⑬  $1101_{(g)} \rightarrow 1001_{(2)}$
- ⑭  $1001_{(g)} \rightarrow 1110_{(2)}$
- ⑮  $0011_{(g)} \rightarrow 0010_{(2)}$
- ⑯  $01011101011_{(g)} \rightarrow 01101001101_{(2)}$
- ⑰  $11100101011100_{(g)} \rightarrow 10111001101000_{(2)}$
- ⑱  $1010101010101010_{(g)} \rightarrow 110011001100110011_{(2)}$

### 8. excess-3 코드를 gray 코드로 변환

십진수	3-초과 코드	그레이 코드
0	0011	0010
1	0100	0110
2	0101	0111
3	0110	0101
4	0111	0100
5	1000	1100
6	1001	1101
7	1010	1111
8	1011	1110
9	1100	1010

### 9. 병렬 패리티를 이용한 에러 검사

3행과 4열이 만나는 비트에서 에러가 발생하였으며, 0을 1로 수정한다.

							패리티 비트 ↓
	1	0	1	1	0	0	0
	1	1	0	0	1	1	1
	0	1	1	0	1	0	0
	0	1	0	0	1	0	1
	1	0	1	0	1	0	0
	0	1	1	1	1	1	0
	1	1	1	1	0	0	0
패리티 워드 →	1	0	0	1	0	1	1

### 10. 해밍코드를 이용한 오류검출 방법

① 1 0 1 1 0 1 1 1 1 1 1 0

비트위치	1	2	3	4	5	6	7	8	9	10	11	12
기호	P <sub>1</sub>	P <sub>2</sub>	D <sub>3</sub>	P <sub>4</sub>	D <sub>5</sub>	D <sub>6</sub>	D <sub>7</sub>	P <sub>8</sub>	D <sub>9</sub>	D <sub>10</sub>	D <sub>11</sub>	D <sub>12</sub>
헤밍 코드	1	0	1	1	0	1	1	1	1	1	1	0
P <sub>1</sub> 계산	1	1		1		0		1		1		1
P <sub>2</sub> 계산	1		0	1			1	1			1	1
P <sub>4</sub> 계산	1				1	0	1	1				0
P <sub>8</sub> 계산	0							1	1	1	1	0
P <sub>8</sub> P <sub>4</sub> P <sub>2</sub> P <sub>1</sub> = 0111 = 7 : 7번 비트에 에러가 발생. 1 → 0으로 교정												

② 1 1 1 1 0 1 0 0 1 0 1 0

비트위치	1	2	3	4	5	6	7	8	9	10	11	12
기호	P <sub>1</sub>	P <sub>2</sub>	D <sub>3</sub>	P <sub>4</sub>	D <sub>5</sub>	D <sub>6</sub>	D <sub>7</sub>	P <sub>8</sub>	D <sub>9</sub>	D <sub>10</sub>	D <sub>11</sub>	D <sub>12</sub>
헤밍 코드	1	1	1	1	0	1	0	0	1	0	1	0
P <sub>1</sub> 계산	0	1		1		0		0		1		1
P <sub>2</sub> 계산	0		1	1			1	0			0	1
P <sub>4</sub> 계산	0				1	0	1	0				0
P <sub>8</sub> 계산	0							0	1	0	1	0
P <sub>8</sub> P <sub>4</sub> P <sub>2</sub> P <sub>1</sub> = 0000 = 0 : 에러 없음												

③ 0 1 1 0 1 1 0 0 1 1 1 0

비트위치	1	2	3	4	5	6	7	8	9	10	11	12
기호	P <sub>1</sub>	P <sub>2</sub>	D <sub>3</sub>	P <sub>4</sub>	D <sub>5</sub>	D <sub>6</sub>	D <sub>7</sub>	P <sub>8</sub>	D <sub>9</sub>	D <sub>10</sub>	D <sub>11</sub>	D <sub>12</sub>
헤밍 코드	0	1	1	0	1	1	0	0	1	1	1	0
P <sub>1</sub> 계산	0	0		1		1		0		1		1
P <sub>2</sub> 계산	1		1	1			1	0			1	1
P <sub>4</sub> 계산	0				0	1	1	0				0
P <sub>8</sub> 계산	1							0	1	1	1	0
P <sub>8</sub> P <sub>4</sub> P <sub>2</sub> P <sub>1</sub> = 1010 = 10 : 10번 비트에 에러가 발생. 1 → 0으로 교정												

④ 0 1 1 0 0 0 0 0 1 0 1 0 1 1 0

비트위치	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
기호	P <sub>1</sub>	P <sub>2</sub>	D <sub>3</sub>	P <sub>4</sub>	D <sub>5</sub>	D <sub>6</sub>	D <sub>7</sub>	P <sub>8</sub>	D <sub>9</sub>	D <sub>10</sub>	D <sub>11</sub>	D <sub>12</sub>	D <sub>13</sub>	D <sub>14</sub>	D <sub>15</sub>
헤밍 코드	0	1	1	0	0	0	0	0	1	0	1	0	1	1	0
P <sub>1</sub> 계산	0	0		1		0		0		1		1		1	0
P <sub>2</sub> 계산	0		1	1			0	0			0	1			1
P <sub>4</sub> 계산	0				0	0	0	0				0	1	1	0
P <sub>8</sub> 계산	0							0	1	0	1	0	1	1	0
P <sub>8</sub> P <sub>4</sub> P <sub>2</sub> P <sub>1</sub> = 0000 = 0 : 에러 없음															

⑤ 1 0 1 0 1 1 0 0 1 1 1 0 1 1 0

비트위치	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
기호	P <sub>1</sub>	P <sub>2</sub>	D <sub>3</sub>	P <sub>4</sub>	D <sub>5</sub>	D <sub>6</sub>	D <sub>7</sub>	P <sub>8</sub>	D <sub>9</sub>	D <sub>10</sub>	D <sub>11</sub>	D <sub>12</sub>	D <sub>13</sub>	D <sub>14</sub>	D <sub>15</sub>
헤밍 코드	1	0	1	0	1	1	0	0	1	1	1	0	1	1	0
P <sub>1</sub> 계산	0	1		1		1		0		1		1		1	0
P <sub>2</sub> 계산	1		0	1			1	0			1	1			1
P <sub>4</sub> 계산	0				0	1	1	0					0	1	1
P <sub>8</sub> 계산	1							0	1	1	1	0	1	1	0
P <sub>8</sub> P <sub>4</sub> P <sub>2</sub> P <sub>1</sub> = 1010 = 10 : 10번 비트에 에러가 발생. 1 → 0으로 교정															

### 11. ASCII코드 표현

- ① 1001010 : J
  - ② 1101111 : o
  - ③ 1101000 : h
  - ④ 1101110 0100000 : n (space)
  - ⑤ 1000100 1101111 1100101 : Doe
- ⇒ 1001010 1101111 1101000 1101110 0100000 1000100 1101111 1100101
- J            o            h            n                                  D            o            e

### 12. ASCII 코드 표현

- ① 47 → G
- ② 4E → N
- ③ 36 → 6
- ④ 52 → R

### 13. 각종 코드 표현방법

- ① 2진수: 100100111
- ② BCD 코드: 0010 1001 0101
- ③ ASCII 코드: 0110010 0111001 0110101

### 14. 표준 BCD 코드 표현

- ① E → 1110101
- ② S → 1010010
- ③ M → 1100100
- ④ 7 → 0000111

### 15. EBCDIC 코드 표현

- ① C7 → G
- ② E5 → V
- ③ D6 → O
- ④ F9 → 9