0908 논리회로

clock: 0과 1사이의 신호가 들어왔을 때 이게 저장되는 값의 타이밍을 저장

1) \*combinational 회로에 입력이 들어가면 출력이 나오는 것 (입력이 출력을 결정함) 사칙연산 같은 느낌

2) \*sequential: 얘도 입력이 들어가면 출력인데 메모리에 저장을 하고 메모리에서 저장된 값을 출력함

(\*memory + combinational logic) 앞의 clock에 대해 저장된 값을 사전에 알아야 함

(메모리에 기억장치가 있어서 이를 내보내고 다시 저장)

ex) 011100111 에서 11이 나온 위치를 찾아라

그러면 sequence는 앞에서도 순차적으로 실행하면서 해당 11이 나온 위치를 메모리에 저장

chapter1

digital 시스템을 설계할 때 이진수를 사용하는 이유는 on/off를 표현하기 위해서

중간고사 n진수를 m진수로 바꿔라

ex) 8진수를 24진수로 바꾸어라

1. 먼저 8진수를 10진수로 바꾼다.

10진수를 24진수로 바꾼다.

|  |
| --- |
|  |

(23F)24 를 7진수로 바꾸어라

(23F)24->(567)10

(567)10 -> (1440)7

보수개념 중요

컴퓨터는 양수는 0으로 시작하고 음수는 1로 시작한다.

|  |
| --- |
|  |

signed(부호) magintude(크기)[-7~+7]

signed(부호) 1's complement(1의 보수)[-7~+7]

ex) 1의 보수+5는 0101 -1의 보수-5는 1010

1의 보수는 1111 에서 뺴준것(1의 보수는 +0과 -0 이같기 때문에 사용하지 못함)

2의 보수는 1의보수에 마지막 자리에 1을 더한 것(음수만)

signed 2's complement [-8~+7]

컴퓨터는 양수만 계산할 수 있음 그렇기 때문에 보수를 구해줘서 +형태로 나타내어 줘야 함

9의 보수 (99..... 에서 빼주면 됨)

ex) 13-11= 13+99 -11 -99 = 101 -99 = 101 + 1 - (99+1) = 2

-> 99-11 = 88(11의 보수) / 88 +13 +1(보수 계산할 때 항상 더해준다고 봐도됨) = 102 -> 2

자릿수가 늘어남 3자리로 앞자리가 1이다 -> 양수이다(캐리)

ex) 11-13 = 11 +99 -13 -99 = 97 - 99 = -(99 -97) = -2

-> 99 - 13 = 86(13의 보수) 11+86+1 = 98

자릿수가 2자리로 유지 앞자리가 1일 아님 그러므로 얘는 -2로 봐야함

10의 보수 ( 9의 보수 +1 해주면 됨)

ex) 13-11 = 13 +89 -100 = 2

13 -11 = 13 +89 = 102 (앞의 1은 캐리 그러므로 양수) = 2

ex) 11 -13 = 98-100 =-2

11 - 13 = 11 + 87 = 98 (캐리가 없으므로 얘는 음수) -2

근데 이진수랑 10진수는 캐리 무시한다!(적용이 안되는 경우들이 있음)

ex) 이진수 -4 같은건 1100이다. 만약 캐리를 고려한다면 얘는 12로 보는 것이 맞지만 얘는 -4이기 때문에 X 그러므로 이러한 오류를 방지하기 위해서 자릿수를 넉넉히 잡자!

ex) 2-4 = 000010+001100= 001110 or 4+7 = 000111+000100=001011

1.7장 BCD

BCD: binary code dicimal(중간고사)

2진수의 합이 10보다 크거나 같으면 6을 더해준다 그렇게 하면 BCD가 나온다.

ex) 184 +576

184 = (0001) + (1000) + (0100)

576 = (0101) + (0111) + (0110)

---------------------------------------

0110 (1111 1010)2,3번째는 합이 10보다 크니 여기에 +6 즉 0110을 더해라

1111+1 (1)0000

[10000 캐리 발생]

+0110

(1)0110

캐리 발생

0111 0110 0000

1.weighted code 각자리마다 값이 존재하는 코드

ppt 15장 BCD, 2421, excess-3, 8 4 -2 -1

2. self complementary code

스스로 자기의 보수를 만드는 코드

2421, excess-3, 84-2-1

(참고로 1의 9의 보수는 8) 0과 9는 서로 0과 1을 바꾸어주면 보수이다.

1과 8은 서로 0과 1을 바꾸어주면 보수이다.

3. gray code

only one bit is changed in the sequence

to prevent an error or ambiguity during the program

(ex. 8421코드에서 7은 0111 8은 1000 7에서 8이 될떄 총 4비트가 달라진다)

gray code 만드는 법 채워넣을 것

|  |
| --- |
|  |

AscII코드

American Standard Code for Information Interchange 약자 알아올것 (중간고사)

5라는 걸 컴퓨터로 쓸떄 문자 5와 숫자 5를 쓰는 방법이 다름

문자 5 ()0110101{앞에 한비트는 비어놓는다}

숫자 5 00000101

문자 5 ()0110101{앞에 한비트는 비어놓는다}

한비트 비워 놓은 이유=> 전송오류를 막고자 한것 even parity

(1의 개수를 세서 1의 개수를 짝수로 유지하는데 사용)

근데 만약에 1개 이상이 바뀌면 오류를 잡지 못한다는 단점이 있다.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  | With even parity | With odd parity | | ASCII A = 1000001 | 01000001 | 11000001 | | ASCII T = 1010100 | 11010100 | 01010100 | |

register: n개의 0과 1을 저장하는 것

컴퓨터는 memory와 cpu가 있는데 memory에서 자료를 넘겨주면 cpu에서 계산해서 memory로 다시 넘겨줌

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
|  | |