LAB7 Pre-report

20160410 윤석훈

1. What is the BCD adder?

Binary-Coded Decimal로 표현된 숫자들을 더하는 가산기이다. Carry-in과 두 Binary Operands를 입력받아서 BCD의 각 4비트 자릿수끼리의 비트수준의 합인 Sum과 Carry-out을 출력한다.

2. What’s the 10’s complement?

10의 보수는 어떤 10진수의 각 자릿수와 9와의 차이를 구한 뒤 LSB에 1을 더함으로써 얻어진다.

3. How to subtract by using 10’s complement?

뺄셈에서 결과값이 음수인 경우와 음수가 아닌 경우를 나눈다.

A-B를 수행했을 때 결과값이 음수가 아닌 경우에는(A>=B) A에다 B의 10의 보수를 더한 값을 구한다. 그리고 이때 A와 B의 자릿수 이상의 Carry가 발생하면 무시한다.

A-B를 수행했을 때 결과값이 음수인 경우에는(A<B) A에다 B의 10의 보수를 더한 결과값이 10의 보수로 표현된 것이므로 100(2BCD이므로 10^2)에서 결과값을 뺀다. 이것은 A-B의 절댓값이므로 ‘–‘기호를 첨가하여 음수임을 나타낸다. A-B<0 이므로 A+(100-B)=100+(A-B)<100 이므로 Carry는 발생하지 않는다.

4. Pseudo Code of 10’s complement

Architecture of TensComplement (Input A, Output B)

One<=A(3 downto 0)

Ten <=A(7 downto 4)

U0: component 4BitBinaryAddition(“1001”, not One , Ans)

U1: component 4BitBinaryAddition(Ans, “0001”, AnsOne)

U2: component 4BitBinaryAddition(“1001”, not Ten , Ans)

U3: component 4BitBinaryAddition(Ans, “0001”, AnsTen)

Signal(3 downto 0)<=AnsOne

Signal(7 downto 4)<=AnsTen

U4: component BCDAddition(Signal, “00000001”,B)

// 8비트 BCD 입력 A에서 각 자릿수와 9와의 차이를 반환한 것을 8비트의 Signal에 담는다. 이 Signal과 1의 합을 BCDAdder로 구하면 그 값이 A의 10의 보수이다.

5. BCD Adder

Library ieee;

Use ieee.std\_logic\_1164.all;

Use ieee.std\_logic\_unsigned.all;

Entity BCDADDER is

PORT (X,Y: in std\_logic\_vector(7 downto 0);

S: OUT std\_logic\_vector(11 downto 0));

END BCDADDER;

Architecture behavior of BCDADDER is

Component 5BitBinaryAddition(X, Y: in std\_logic\_vector (4 downto 0); Ans: out std\_logic\_vector (5 downto 0));

Signal A,B,C,D,E,F,G,H,I ,J,K: std\_logic\_vector(4 downto 0);

Signal carry1,carry2 : std\_logic;

Begin

A<=’0’ & X(3 downto 0);

B<=’0’ & Y(3 downto 0);

U0: component 5BitBinaryAddition(A,B,C);

Carry1<=’1’ when C>9 ELSE ‘0’;

U1: component 5BitBinaryAddition(C,’00110’,D);

S(3 downto 0) <= C(3 downto 0) WHEN (Carry1=’0’) ELSE D(3 downto 0);

H<=’0000’ & D(4);

E<=’0’ & X(7 downto 4);

F<=’0’ & Y(7 downto 4);

U2: component 5BitBinaryAddition(E,F,G);

U3: component 5BitBinaryAddition(G,H,I);

Carry2<=’1’ when I>9 ELSE ‘0’;

U4: component 5BitBinaryAddition(I,’00110’,J);

S(7 downto 4) <= I(3 downto 0) WHEN (Carry2=’0’) ELSE J(3 downto 0);

K<=’0000’ & J(4);

S(11 downto 8) <= K;

End Architecture;

뺄셈의 경우 10의 보수를 위의 BCDADDER의 입력값으로 하여 구한다.