컴퓨터 구조

3장 컴퓨터 산술과 논리 연산: 문제풀이

안형태

anten@kumoh.ac.kr

디지털관 139호

1. 2의 보수 표현

- **[1-1]** 다음 수들에 대한 8 비트 길이의 부호화 크기, 1의 보수 및 2의 보수 표현을 각각 구하라.
  - (1) 19 (2) -19
  - (3) 124 (4) -124

- **[풀이]**
  - (1) 부호화-크기: 00010011, 1의 보수: 00010011, 2의 보수: 00010011
  - (2) 부호화-크기: 10010011, 1의 보수: 11101100, 2의 보수: 11101101
  - (3) 부호화-크기: 01111100, 1의 보수: 01111100, 2의 보수: 01111100
  - (4) 부호화-크기: 11111100, 1의 보수: 10000011, 2의 보수: 10000100

1. 2의 보수 표현

- **[1-2]** 다음 수들에 대한 16 비트 길이의 부호화 크기, 1의 보수 및 2의 보수 표현을 각각 구하라.
  - (1) 19　　　　(2) -19
  - (3) 124　　　　(4) -124

- **[풀이]**
  - (1) 부호화-크기: 0000000000010011, 1의 보수: 0000000000010011, 2의 보수: 0000000000010011
  - (2) 부호화-크기: 1000000000010011, 1의 보수: 1111111111101100, 2의 보수: 1111111111101101
  - (3) 부호화-크기: 0000000001111100, 1의 보수: 0000000001111100, 2의 보수: 0000000001111100
  - (4) 부호화-크기: 1000000001111100, 1의 보수: 1111111110000011, 2의 보수: 1111111110000100

2. 응용된 논리연산

- **[2-1]** A 레지스터에 '10001010'이 저장되어 있는 상태에서 B 레지스터와의 어떤 연산을 이용하여 상위 다섯 비트를 보수 값으로 바꾸고자 한다. 어떤 연산과 B 레지스터에 어떤 값이 필요한가?

- **[풀이]**
  - 선택적-보수 연산 수행
  - 연산: XOR, B = 11111000

A = 1 0 0 0 1 0 1 0
B = 1 1 1 1 1 0 0 0
----
0 1 1 1 0 0 1 0 (XOR 연산 결과)

2. 응용된 논리연산

- **[2-2]** A 레지스터에 저장되어 있는 데이터 '11010010'의 우측 다섯 비트에 '01110'을 삽입하려면 어떻게 해야 되는가?

- **[풀이]**
  - 삽입 연산 수행

A = 1 1 0 1 0 0 1 0
B = 1 1 1 0 0 0 0 0 **마스크 (AND 연산)**

A = 1 1 0 0 0 0 0 0 **첫 단계 결과**
B = 0 0 0 0 1 1 1 0 **삽입 (OR 연산)**

A = 1 1 0 0 1 1 1 0 **최종(삽입) 결과**

3. C 플래그와 시프트

- 초기 상태에서 어떤 레지스터에 ‘10110011’이 저장되어 있고, C 플래그의 값은 ‘1’이라고 하자.

- **[3-1]** RLC(Rotate Left with Carry) 연산을 수행한 결과는?

- **[풀이]**
  - C=1, R=01100111

- **[3-2]** 위의 결과에 대하여 RRC(Rotate Right with Carry) 연산을 두 번 연속 수행한 결과는?

- **[풀이]**
  - C=1, R=11011001

4. 오버플로우 탐지

❑ **[4-1]** 2의 보수로 표현된 아래의 수들에 대한 덧셈을 수행하고, 오버플로우가 발생하였는지 확인하라.

- (1) 01010001+01011100 (2) 11001100+10111010

❑ **[풀이]**

- (1) 81 + 92 = 173

Carry: 0 1 0 1 0 0 0 0

0 1 0 1 0 0 0 1

+ 0 1 0 1 1 1 0 0

----

1 0 1 0 1 1 0 1

$C_7 \oplus C_6$ **= 0 XOR 1 = 1** 이므로, 오버플로우가 발생함

4. 오버플로우 탐지

- (2) -48 + -70 = -118

Carry: 1 1 1 1 1 0 0 0

1 1 0 0 1 1 0 0

+ 1 0 1 1 1 0 1 0

버림→1 1 0 0 0 0 1 1 0

$C_7 \oplus C_6$ **= 1 XOR 1 = 0** 이므로, 오버플로우는 발생하지 않음

5. 부호 없는 2진수들 간의 곱셈

- **[5-1]** 부호 없는 2진수들 간의 곱셈 (1001×0111)이 수행되는 과정에서 레지스터들의 변화를 작성하라.

- **[풀이]**
  - M(피승수) = 1001(9), Q(승수) = 0111 (7), A = 0000, C = 0

	C	A	Q($Q_3Q_2Q_1Q_0$)	설명
초기 상태	0	0000	0111	초기상태
싸이클1	0 0	1001 0100	0111 1011	Q_0=1 이므로, A ← A+M 우측 시프트 (C-A-Q)
싸이클2	0 0	1101 0110	1011 1101	Q_0=1 이므로, A ← A+M 우측 시프트 (C-A-Q)
싸이클3	0 0	1111 0111	1101 1110	Q_0=1 이므로, A ← A+M 우측 시프트 (C-A-Q)
싸이클4	0	0011	1111	Q_0=0 이므로, 우측 시프트 (C-A-Q) 곱셈의 결과는 63

6. Booth Algorithm

❑ **[6-1]** Booth Algorithm을 활용하여 5×(-4)의 곱셈에서 레지스터들의 변화 과정을 작성하라.

❑ **[풀이]**

- M(피승수) = 0101(5), Q(승수) = 1100(-4), A = 0000, Q_{-1} = 0

A	**Q($Q_3Q_2Q_1Q_0$)**	Q_{-1}	**n**	**설명**
0000	1100	0	4	초기 상태
0000	0110	0	3	(Q_0Q_{-1})=00 이므로, 연산없이 AQQ_{-1}을 산술적 우측-시프트
0000	0011	0	2	(Q_0Q_{-1})=00 이므로, 연산없이 AQQ_{-1}을 산술적 우측-시프트
1011 1101	0011 1001	0 1	1	(Q_0Q_{-1})=10 이므로, A=A-M=0000+1011 산술적 우측-시프트
1110	1100	1	0	(Q_0Q_{-1})=11 이므로, 연산없이 AQQ_{-1}을 산술적 우측-시프트 계수에서 1을 빼면, 0이므로 계산 종료 곱셈의 결과는 -20

7. IEEE 754 표준

- **[7-1]** IEEE 754 표준을 사용하여 32-비트 부동소수점 형식으로 나타내라.
  - (1) 253.25 (2) -1.625

- **[풀이]**
  - (1) 253.25를 IEEE 754 표준 부동소수점 표현
    - $253.25_{10} = 11111101.01_2 = 1.111110101 \times 2^7$
    - 부호(S) 비트 = 0 (+)
    - 지수(E) 비트 = 00000111 + 01111111 = 10000110 (바이어스 127을 더함)
    - 가수(M) 비트=11111010100000000000000 (소수점 좌측의 첫번째 '1' 제외)

S	E	M
0	10000110	11111010100000000000000

7. IEEE 754 표준

- (2) -1.625를 IEEE 754 표준 부동소수점 표현
  - $1.625_{10} = 1.101_2 = 1.101 \times 2^0$
  - 부호(S) 비트 = 1 (-)
  - 지수(E) 비트 = 00000000 + 01111111 = 01111111 (바이어스 127을 더함)
  - 가수(M) 비트=10100000000000000000000 (소수점 좌측의 첫번째 ‘1’ 제외)

S	E	M
1	01111111	10100000000000000000000

Q & A

End!