1장 기본 개념

1.1 서론

- 예비지식
  - 컴퓨터 프로그래밍 (C, C++, Java, Basic 언어 등)
- 필요한 소프트웨어
  - 어셈블러(assembler) (예) MASM, TASM, NASM, AS (GNU)
  - 링커(linker) (예) LINK, LD (GNU)
  - 디버거(debugger) (예) DEBUG, GDB (GNU)

  (통합개발환경에서는 이러한 기능을 모두 제공함)
- 어셈블리 언어 프로그래밍을 위해서 필요한 사항
  - 컴퓨터 하드웨어/구조에 대한 기본 지식
  - 명령어 집합(instruction set)에 대한 이해
  - 기본적인 프로그래밍 능력

- 어셈블리 언어와 기계어의 연관 관계
  - 어셈블리언어 명령어(ADD, MOV 등)는 기계어와 **1대1 대응** 관계

  (참고) 고급언어 문장은 기계어와 **일대다** 대응관계
- 어셈블리 언어의 이식성
  - 기계어에 종속적이므로 다른 기계어를 사용하는 프로세서에 대해서는 이식성(portable)이 없음
- 어셈블리 언어를 배우는 목적
  - 메모리, 실행 속도 최적화가 필요한 프로그램 작성
  - 임베디드 시스템 프로그램 작성
  - 컴퓨터 하드웨어, 컴퓨터 구조, 운영체제 등에 대한 이해 증진
  - 특정 하드웨어를 위한 장치 드라이버(device driver) 프로그램 작성
  - 고급언어로 수행할 수 없는 특별한 명령어를 사용하는 프로그램 작성

어셈블리 언어와 고급 언어의 비교

- 한 platform을 위한 중간 크기 이상의 프로그램
  - 고급언어가 코드 작성과 관리가 용이함
- 다양한 platform을 지원하는 프로그램
  - 고급언어가 이식성이 좋음 – 최소한의 수정으로 컴파일 가능
- 하드웨어 device driver
  - 고급언어는 하드웨어에 대한 직접적인 접근을 지원하지 않을 수 있음
  - 어셈블리언어는 하드웨어 접근 가능
- 메모리 제약이 있는 시스템에서의 프로그램 (임베디드 시스템)
  - 어셈블리언어를 사용한 코드가 실행코드가 작고 빠르게 실행됨

(표 1-1 참고)

(cf) 최신 운영체제는 시스템 보호를 위해 응용 프로그램의 하드웨어 접근을 제한하며 어셈블리 언어 프로그램에도 적용됨

1.2 가상기계 개념

가상기계 VM1	L1 언어 실행 —— 고급언어: Java 언어
가상기계 VM0	L0 언어 실행 —— 기계어: Java 바이트코드

- L1 언어로 작성된 프로그램의 수행 방법
  - 해독 (interpretation)
  - 변환 (translation)

- 해독(Interpretation)
  - L0 프로그램이 L1 프로그램의 명령어를 하나씩 해석하여 수행
    (해독하는 L0 프로그램: 해독기, 인터프리터)
- 변환(Translation)
  - L1 프로그램을 완전히 L0 프로그램으로 변환한 후 L0 프로그램을 수행
    (변환하는 L0 프로그램: 컴파일러)

언어의 번역 과정

English: Display the sum of A times B plus C.

↓

C++: cout << (A * B + C);

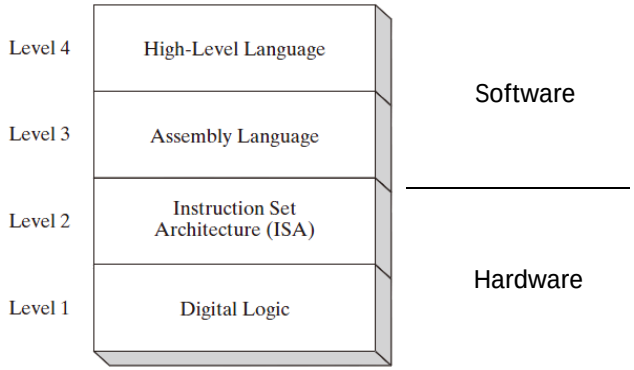
↓

Assembly Language:
mov eax, A
mul B
add eax, C
call WriteInt

→

Intel Machine Language:
A1 00000000
F7 25 00000004
03 05 00000008
E8 00500000

가상 기계 레벨

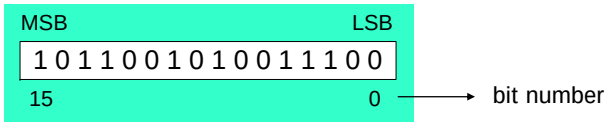


- Level 1: 디지털 논리 회로(Digital Logic) - 하드웨어
  - CPU: 하드웨어는 논리 게이트로 구성됨
  - 게이트들은 트랜지스터를 사용하여 구현됨
- Level 2: 명령어 집합 구조(Instruction Set Architecture)
  - 기계어(machine language)라고 함
  - 컴퓨터 하드웨어(Level1) 또는 마이크로프로그램에 의해서 수행됨
- Level 3: 어셈블리언어(Assembly Language)
  - 명령어 니모닉(mnemonic)은 기계어와 1대1 대응
  - 프로그램은 기계어(명령어집합구조)(Level 2)로 쉽게 변환됨
- Level 4: 고급 언어(High-Level Language)
  - 응용 프로그램 작성용 언어
    - C++, Java, Pascal, Visual Basic . . .
  - 프로그램은 어셈블리 언어(Level 3)로 변환된 후 기계어로 변환됨

1.3 데이터의 표현

- 비트(bit)
  - **bi**nary digi**t** 의 약자 (2진수 1자리)
  - 0 또는 1의 값을 가짐

- 2진수(binary number) 표현



  - MSB – **최상위 비트**(most significant bit)
  - LSB – **최하위 비트**(least significant bit)

부호없는 정수

- 2진수의 각 bit는 2의 거듭제곱을 표현

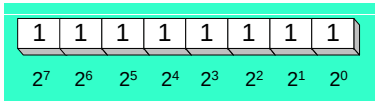
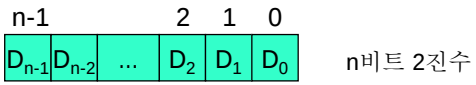


Table 1-3 Binary Bit Position Values.

2^n	Decimal Value	2^n	Decimal Value
2^0	1	2^8	256
2^1	2	2^9	512
2^2	4	2^{10}	1024
2^3	8	2^{11}	2048
2^4	16	2^{12}	4096
2^5	32	2^{13}	8192
2^6	64	2^{14}	16384
2^7	128	2^{15}	32768

- 2진수의 10진수 변환

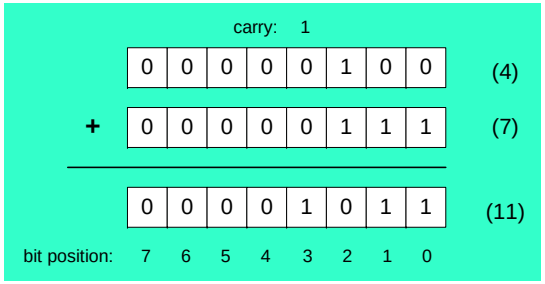


  - 10진수 값 = $(D_{n-1} \times 2^{n-1}) + (D_{n-2} \times 2^{n-2}) + \ldots + (D_1 \times 2^1) + (D_0 \times 2^0)$

(예) $00001001_2 = (1 \times 2^3) + (1 \times 2^0) = 8+1 = 9_{10}$

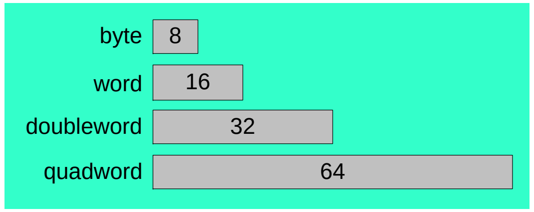
2진수 덧셈

- 1비트 덧셈
  - 0 + 0 = 0 0 + 1 = 1
  - 1 + 0 = 1 1 + 1 = 10 1 + 1 + 1 = 11
- n비트 덧셈
  - LSB부터 덧셈
  - carry는 다음 자리로 올려서 덧셈



데이터의 크기

■ 데이터의 크기
- 바이트(byte) – 8 비트
- 워드(word) – 16 비트, 2 바이트
- 더블워드(doubleword) – 32 비트, 4 바이트, 2 워드
- 쿼드워드(quadword) – 64 비트, 8 바이트, 4 워드



(cf) Intel 프로세서에서 워드의 크기는 16비트이지만,
다른 프로세서에서는 32비트와 같이 다른 크기로 사용될 수 있다.

부호없는 정수의 표현 범위와 크기 단위

■ 부호없는 정수의 표현 범위

Table 1-4 Ranges of Unsigned Integers.

Storage Type	Range (low–high)	Powers of 2
Unsigned byte	0 to 255	0 to $(2^8 - 1)$
Unsigned word	0 to 65,535	0 to $(2^{16} - 1)$
Unsigned doubleword	0 to 4,294,967,295	0 to $(2^{32} - 1)$
Unsigned quadword	0 to 18,446,744,073,709,551,615	0 to $(2^{64} - 1)$

■ 크기 단위

K = 2^{10} 또는 10^3 (kilo)
M = 2^{20} 또는 10^6 (mega)
G = 2^{30} 또는 10^9 (giga)
T = 2^{40} 또는 10^{12} (tera)
P = 2^{50} 또는 10^{15} (peta)
E = 2^{60} 또는 10^{18} (exa)
Z = 2^{70} 또는 10^{21} (zetta)
Y = 2^{80} 또는 10^{24} (yotta)

16진수 표현

■ 2진수, 10진수, 16진수 표현

Binary	Decimal	Hexadecimal	Binary	Decimal	Hexadecimal
0000	0	0	1000	8	8
0001	1	1	1001	9	9
0010	2	2	1010	10	A
0011	3	3	1011	11	B
0100	4	4	1100	12	C
0101	5	5	1101	13	D
0110	6	6	1110	14	E
0111	7	7	1111	15	F

■ 2진수의 16진수 표현: 2진수 4자리를 16진수 1자리로 표현
- 2진수 0100 1100 = 16진수 4C = 10진수 76 (=16x4+12)
  01001100**B** 4C**H** 76

16진수 덧셈과 뺄셈

■ 16진수 덧셈

		1	1
36	28	28	6A
42	45	58	4B
78	6D	80	B5

21 / 16 = 1, rem 5

■ 16진수 뺄셈

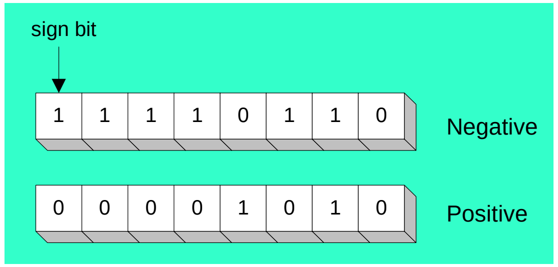
	–1
C6	75
A2	47
24	2E

10h + 5 = 15h

■ 어셈블리 언어에서 주소 계산을 하는 데 16진수 덧셈/뺄셈이 많이 사용됨

부호있는 정수(signed integer)

- MSB를 부호 비트로 사용
  - 1 = negative, 0 = positive



2의 보수 표현법

- 2의 보수 표현법 – 부호있는 정수를 표현하는 방법 중 하나
  - 양수: 크기의 2진수 표현 (M)
  - 음수: 크기의 2의 보수 표현 (2^n-M)

(예)

양수: 65 → 01000001 (최상위비트=0, 65)

음수: -65 → 10111111 (최상위비트=1, 2^8-65)

(예)

65+(-65) →	01000001	65	
	+)10111111	2^8-65	
	~~1~~ 00000000	~~2^8~~	→ carry 무시

2의 보수를 이용한 2진수 뺄셈

- B의 2의 보수 = 2^n – B (n비트 2진수)
- A – B 계산 → A + (B의 2의 보수) 계산

= A + (2^n – B)

= { A ≥ B: ~~2^n~~ + (A – B) carry 무시하면 정답
  { A < B: 2^n – (B – A) B–A의 2의 보수 → –(B–A)를 표현

```
  0 0 0 0 1 1 0 0              0 0 0 0 1 1 0 0
– 0 0 0 0 0 0 1 1   ──────►    1 1 1 1 1 1 0 1
                             1 0 0 0 0 1 0 0 1
```

(마지막 줄 앞의 1은 취소선)

부호있는 정수의 표현 범위

- 부호있는 정수의 표현 범위
  - MSB가 부호 표현에 사용됨
  - 나머지 n-1비트가 크기 표현에 사용됨

  → – 2^{n-1} to (2^{n-1} –1)

Storage Type	Range (low–high)	Powers of 2
Signed byte	–128 to +127	-2^7 to $(2^7 - 1)$
Signed word	–32,768 to +32,767	-2^{15} to $(2^{15} - 1)$
Signed doubleword	–2,147,483,648 to 2,147,483,647	-2^{31} to $(2^{31} - 1)$
Signed quadword	–9,223,372,036,854,775,808 to +9,223,372,036,854,775,807	-2^{63} to $(2^{63} - 1)$

문자의 표현

- 문자의 표현
  - 문자를 2진수에 mapping시켜서 표현
- 문자 표현 방식
  - 표준 ASCII: 7비트 (0 – 127) → 영문자, 숫자, 일부특수문자 표현
  - ANSI 문자집합 (확장 ASCII): 8비트 (0 – 255) → 서유럽문자 포함
  - 유니코드(Unicode):
    - **UTF16**: 16비트 → 다국어 문자 포함(한글/한자/일어 등)
    - UTF8: 8비트 → 유니코드를 위한 가변길이 부호화
    - UTF32: 32비트

    (**U**nicode **T**ransformation **F**ormat)

(예)

ASCII코드	문자
0110000 ~ 0111001	0 ~ 9
1000001 ~ 1011010	A ~ Z
1100001 ~ 1111010	a ~ z

ASCII 제어 문자

- ASCII 제어문자
  - 화면 및 동작 제어에 사용되는 문자
  - 0부터 31까지의 문자코드가 사용됨 (Ctrl+문자 로 입력 가능)

ASCII 코드(10진수)	설명
8	백스페이스(backspace) (한 칸 왼쪽으로 이동한다.)
9	수평 탭 (*n*칸 앞으로 건너뛴다.)
10	라인피드(line feed) (다음 줄로 이동한다.)
12	폼피드(form feed) (다음 프린터 페이지로 이동한다.)
13	캐리지 리턴(carriage return) (가장 왼쪽 칸으로 이동한다.)
27	탈출 문자(escape character)

숫자 데이터 표현

- 2진수 정수(Binary Integer)
  - 8비트 배수 크기(8, 16, 32, 48, 64)의 2진수로 메모리에 저장됨
- ASCII digit string
  - 수를 눈에 보이게 단순히 표현한 것

형식	값
ASCII 2진수	"01000001"
ASCII 10진수	"65"
ASCII 16진수	"41"
ASCII 8진수	"101"