알기쉬운

# 電子情報處理組織(EDPS) ◀ II ▶

圖協出版部

CHAPTER 2.

## 데이터의 표현 방법

### 2.1 개 요

기호들도 정보를 전달한다. 기호 자체는 정보가 아니지만, 단지 정보를 표시한다. 이 페이지에 인쇄된 문자들은 어떤 사람에게는 한가지 뜻을 전하고, 다른 사람에게는 또 다른 뜻을 전하며, 이것의 의미를 모르는 사람에게 있어서는 아무런 뜻도 없는 기호(symbol)들이다(Fig. 2-1).

컴퓨터 시스템에서 표시되는 자료는 편지로서 사람들 간에 통신하는 방법과 여러 면에서 비슷하다. 전달될 정보는 일련의 기호로 바꾸어져야 한다. 영어에 있어서 이러한 것들은 알파벳이나 숫자, 구두점과 매우 비슷하다. 이러한 기호들은 규정된 순서대로 종이 위에 기록되며, 이것을 읽고 해석할 수 있는 사람에게 전해진다.

보편적으로 컴퓨터 시스템과 통신은 자료처리 기계가 읽고 해석할 수 있는 부호의 셋트로 자료를 변화시

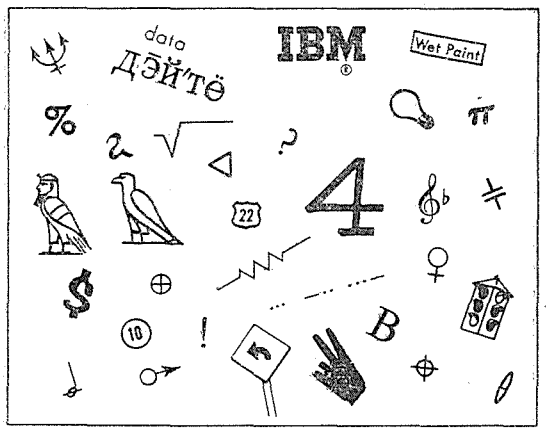
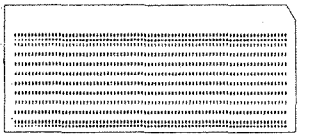
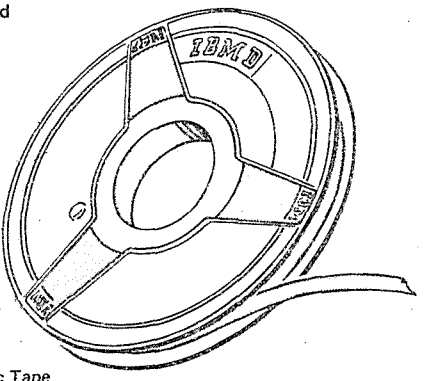


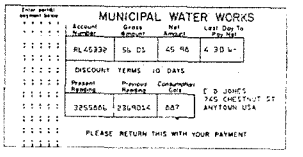

Fig. 2-1. Symbols for communication



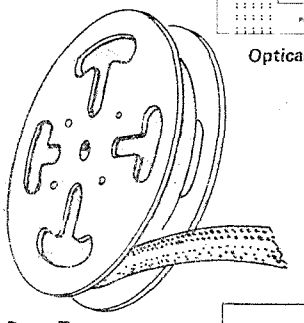
IBM Card



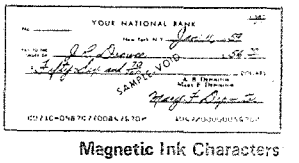

Magnetic Tape




Optically Readable Characters



Paper Tape




Magnetic Ink Characters

Direct Access Storage Device

Fig. 2-2. Data recording media

켜 줄 것을 요구하고 있다. 이러한 부호(기호)들은 사람들이 사용하는 것과는 다르다. 왜냐하면, 표시된 정보는 기계의 설계와 작업에 맞아야 하기 때문이다. 이러한 부호(표시된 부호들의 뜻)들의 선택은 디자이너 분야에 있어서는 약속에 관한 문제이다. 중요한 사실은 정보가 부호로서 표시되어야 한다는 것이며, 이 부호들은 사람과 기계 간의 통신을 위한 언어가 된다.

컴퓨터 시스템에 사용되는 정보는 Punched card, Paper tape, Magnetic tape, Direct Access Storage Device(DASD), Magnetic ink character, Optically recognizable character, Micro-film과 Display screen image, Communication network signal 등의 형태이다. 이러한 종류는 매년 크게 발달하고 있다. 몇 개는 [Fig. 2-2]에서 묘사하였다.

데이터는 카아드의 특정한 지역에 작은 직사각형 구멍의 표시로서 펀치된 카아드 위에 나타내지게 된다. 비슷한 방법으로서 종이 테이프의 길이에 따라 나타나는 작은 원형의 구멍이 데이터를 표시한다. 마그네틱 테이프나 DASD에서 기호들은 Spot나 혹은 Bit라 불리우는 특정한 형태를 배열한 작은 자화된 지역이다. Magnetic ink character는 종이 위에 인쇄된다. 그 문자의 형상과 그 잉크의 마그네틱 특성은 전부 인쇄된 데이터를 사람과 기계가 읽을 수 있도록 하는 것이다. Optical character의 형태는 배후의 종이와 대조하여 동시에 기계나 사람이 Optical character를 읽을 수 있도록 해 준다.

각 매체는 데이터를 표시하기 위해서 코우드나 또는 기호의 특별한 배열을 필요로 한다. 이 코우드에 관해서는 [2.3]에서 설명한다.

컴퓨터의 입력장치는 기록 매체들 중의 하나로부터 정보를 감지하거나 읽을 수 있도록 설계된 기계이다. 읽는 과정에서 기록된 자료는 전자적인 형태로 바뀌어지거나 기호로 나타내지게 되며, 이 자료는 후에 자료처리를 위하여 기계가 사용하게 된다.

출력 장치는 컴퓨터 시스템으로부터 정보를 받아서 지시된 출력 매체를 통해 그것을 기록하는 기계이다.

모든 입출력 장치가 모든 컴퓨터 시스템에 직접 사용될 수는 없다. 그렇지만 하나의 매체 위에 기록된 자료는 다른 시스템에서 사용하기 위해 다른 매체에 바꿔 쓰여질 수 있다. 예를 들어 카아드나 테이프에 기록된 자료는 마그네틱 테이프에 옮겨 쓰여질 수 있다. 반대로 마그네틱 테이프의 자료도 카아드나 종이 테이프, 인쇄된 보고서 또는 Plotted graph로 바꿔질 수 있다.

사람과 기계 사이에 통신이 이루어지는 것과 같이, 하나의 기계에서 다른 기계로 통신하는 것도 역시 가능하다(Fig. 2-3). 이러한 상호 통신은 전선이나 케이블을 통한 자료(전자적인 형태), 혹은 Radio wave, 또

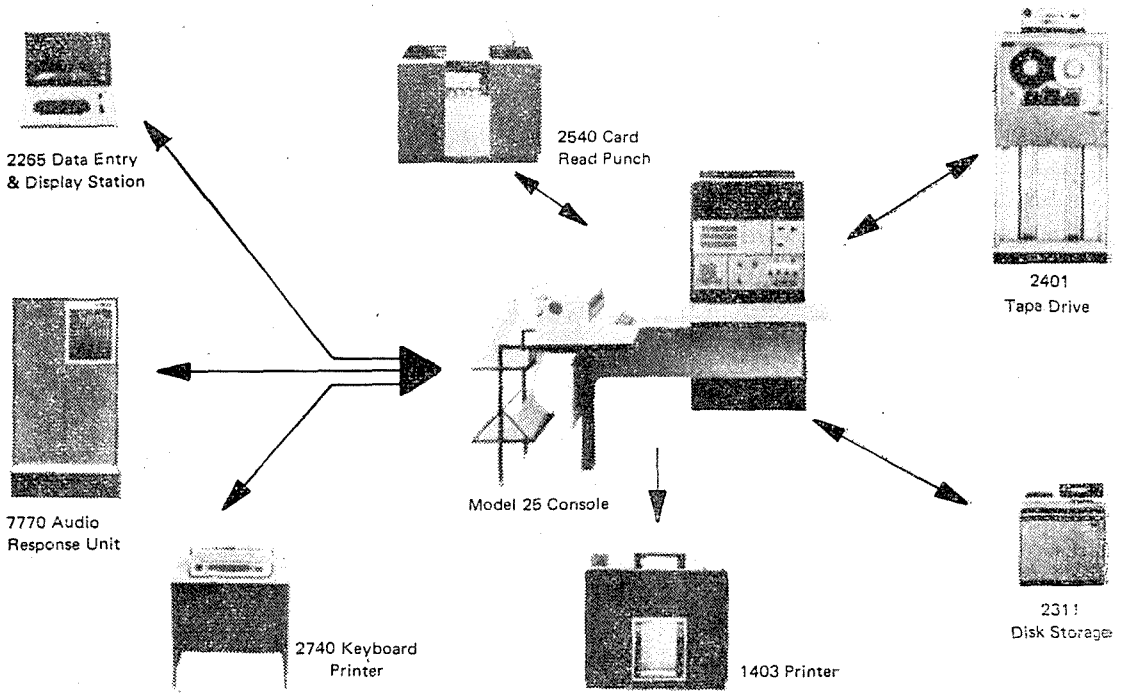



Fig. 2-3. Machine-to-machine communication

는 다른 기계나 시스템의 입력으로 사용될 수 있는 한 기계나 시스템에 기록된 출력의 직접적인 교환일 것이다.

## 2.2 컴퓨터의 자료(Data Representation)

자료표시 방법에는 카아드(card), 종이 테이프(paper tape), 마그네틱 테이프(magnetic tape) 또는 마그네틱 잉크 문자(magnetic ink character)로서 하는 것 뿐만 아니라, 기계 내부에서 자료를 표시하는 방법도 있다.

컴퓨터에서 자료는 트랜지스터(transister), 자기 코어(magnetic core), 전선 등의 많은 전자적 요소들로서 표시된다. 이러한 장치를 통하는 자료의 저장과 흐름은 전자 신호나 표시로 나타난다. 카아드에서 구멍의 유무가 자료를 표시하는 것과 마찬가지로 특정 회로에서 이러한 신호의 유무가 자료를 표시하는 방법인 것이다.

### 2진수 상태

컴퓨터는 2진수 상태로서 그 기능을 수행한다. 이것은 컴퓨터(computer) 구성 요소들이 단지 표시가 가능한 두 상태나 조건만을 표시할 수 있다는 것을 의미한다. 예를 들면, 일상적으로 사용되는 전구는 2진수 형태로서 동작한다. 즉, 켜지거나 꺼지는 것이다. 마찬가지로 컴퓨터 내에서도 트랜지스터는 전도되거나 비전도되는 상태를 유지하고, 자화물질은 한 방향으로 자화되거나 반대 방향으로 자화되며, 특정한 전압·전위는 나타나거나 없어지거나 한다(Fig. 2-4). 구성 요소들이 2진수 형태로 작동하는 것은 컴퓨터에 대한 신호이며, 전기적인 전구로부터 발생되는 빛의 유무는 사람에 대한 신호이다.

컴퓨터 내에서의 자료표시는 특정한 값을 2진수로 표시하거나 일단의 2진수 표시들로 지정 제휴함으로써 이루어진다. 예를 들면 10진수 값을 표시하는 장치는 4개의 Electronic light bulb와 각 전구를 켜거나 끌 수

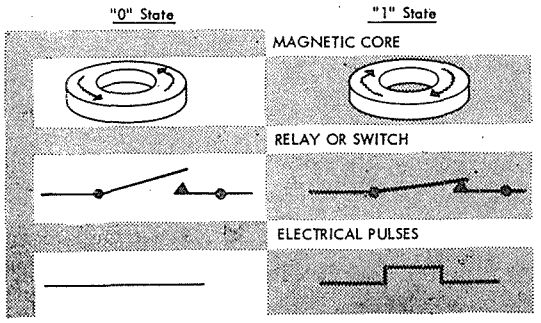



Fig. 2-4. Binary components

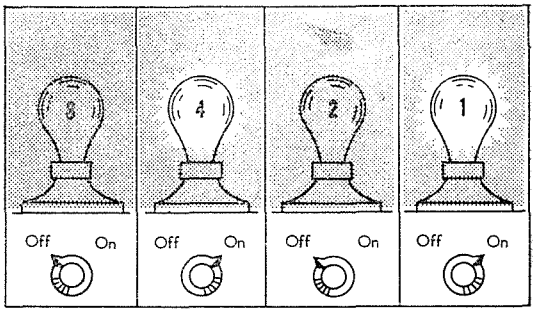



Fig. 2-5. Representing decimal data with binary components

있는 스윗치들로 설계되어 있다(Fig. 2-5).

각 전구는 10진수 값(decimal value) 1, 2, 4, 8을 나타낸다. 불이 켜졌을 때 이것은 10진수 값을 표시한다. 불이 꺼지면 10진수 값은 지워진다. 이러한 배열로서 4개의 전구로 표시되는 10진수 값은 불켜진 전구가 지시하는 수의 합계일 것이다.

0에서부터 15까지의 10진수 값을 나타낼 수가 있다. 0의 값은 모든 전구를 끔으로서 나타낼 수 있으며, 15는 모든 전구를 켬으로서, 9는 8과 1의 전구를 켜고 4와 2의 전구를 끔으로서, 5는 1과 4를 켜고 8과 2를 끄는 등 이와 같은 방법으로 나낼타 수 있다.

앞의 예에서와 같이 각 전구나 Indicator에 지정된 값은 사용된 값 외에 다른 값을 가질 수 있다. 이러한 경우는 새로운 값을 지정하거나 새로운 작업의 설계를 결정할 경우에 해당된다. 컴퓨터에 있어서 2진수 표시의 특정한 숫자로 지정된 값들은 자료표시를 위한 코우드(code)나 언어(language)가 된다.

왜냐하면 2진수 표시는 컴퓨터 내에서 자료를 표시하며, 2진수 표시 방법은 이러한 표시를 설명하는 데 사용된다. 2진수 표기 시스템은 특정한 값을 표시하는데, 0과 1의 단지 두 가지 부호만을 사용하게 된다. 이와 같이 2진수 표기에 있어서, 0은 관련 또는 지정된 값의 부재를 나타내며, 1은 관련 또는 지정된 값의 실재를 나타낸다. 예를 들면, [Fig. 2-5]에 나타난 불켜진 전구의 표시를 설명하면, 다음의 2진수 표기는 0101로 사용되었다.

0과 1의 2진수 표기를 일반적으로 Bit(binary digit)라 부른다. 정확하게 그것은 0 bit와 1 bit라 부른다. 그렇지만 때때로 막연하게 No bit(0)와 Bit(1)라고 부르기도 한다. 예를들면, [Fig. 2-5]의 2진수 표기 0101은 1 bit를 Bit position 1과 4에 갖고 있고, 0 bit를 Bit position 2와 8에 갖고 있다고 설명된다.

### 2진수 체계

어떤 컴퓨터에 있어서 2진수 표기와 관련된 값들은

직접 2진수 시스템과 관계되어 있다. 이러한 시스템은 모든 컴퓨터에서 적용되지는 않지만 이 Numbering 시스템을 사용하여 값을 표시하는 방법은 자료표시의 일반적인 개념을 공부하는데 유용하다.

일반적인 10진법 수 체계는 수값을 표시하기 위하여 열개의 부호나 숫자를 사용하며, 숫자의 위치 값은 일, 십, 백, 천 등을 의미한다. 2진수나 Base-two수 체계는 단지 0과 1의 두가지 부호나 숫자를 사용한다. 비트(bit) 부호들(0이나 1)의 위치 값은 동력의 2가지 진행에 기초를 두고 있다. 2진수의 단위 위치는 1의 값을 가지며, 다음 위치는 2, 다음은 4, 다음은 8, 다음은 16 등을 나타낸다(Fig. 2-6).


Fig. 2-6. Place value of binary numbers

순수한 2진수 표기에 있어서, 2진수의 숫자나 비트들은 일치하는 두가지 Power가 그 수의 각 Position에 있느냐 없느냐를 나타낸다. 1 bit는 그 값의 실재를 의미하며, 0 bit는 그 값의 부재를 나타낸다.

숫자들의 위치 값은(place value) 10진법에서 처럼 일, 십, 백, 천 등을 나타내지 않고, 대신에 일, 이, 사, 팔, 십육 등을 나타낸다. 예를 들어 이러한 시스템을 사용했을 때, 12는 $(1\times2^3)+(1\times2^2)+(0\times2^1)+(0\times2^0)$ 또는 $(1\times8)+(1\times4)+(0\times2)+(0\times1)$을 뜻하는 부호 1100으로 표시된다.

[Fig. 2-7]은 10진수 0에서 16까지를 2진수 표시로 나타낸 것이다.

10진수 0에서 9는 4개의 2진수 숫자들로 표시됨을 주의하라. 10진수 숫자들을 동일한 2진수 숫자로 Coding하거나 표시하는 시스템은 Binary coded decimal (BCD)로 알려져 있다. 예를 들어서 10진수 값 265, 498은 [Fig. 2-8]에서 보이는 Binary coded decimal 로 나타낼 수 있다.

| Decimal Value | Place Value 16 | 8 | 4 | 2 | 1 |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 |
| 2 | 0 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 0 | 1 | 1 |
| 4 | 0 | 0 | 1 | 0 | 0 |
| 5 | 0 | 0 | 1 | 0 | 1 |
| 6 | 0 | 0 | 1 | 1 | 0 |
| 7 | 0 | 0 | 1 | 1 | 1 |
| 8 | 0 | 1 | 0 | 0 | 0 |
| 9 | 0 | 1 | 0 | 0 | 1 |
| 10 | 0 | 1 | 0 | 1 | 0 |
| 11 | 0 | 1 | 0 | 1 | 1 |
| 12 | 0 | 1 | 1 | 0 | 0 |
| 13 | 0 | 1 | 1 | 0 | 1 |
| 14 | 0 | 1 | 1 | 1 | 0 |
| 15 | 0 | 1 | 1 | 1 | 1 |
| 16 | 1 | 0 | 0 | 0 | 0 |

Fig. 2-7. Binary representation of decimal values 0~16

| Decimal Digits | 2 | 6 | 5 | 4 | 9 | 8 |
|---|---|---|---|---|---|---|
| Binary Value | 0 0 1 0 | 0 1 1 0 | 0 1 0 1 | 0 1 0 0 | 1 0 0 1 | 1 0 0 0 |
| Place Value | 8 4 2 1 | 8 4 2 1 | 8 4 2 1 | 8 4 2 1 | 8 4 2 1 | 8 4 2 1 |

Fig. 2-8. Binary coded decimal representation of decimal number 265, 498

## 2·3 컴퓨터 코우드(Computer Code)

자료를 표시(기호화)하는데 사용되는 방법은 Code나 시스템으로 알려져 있다. 컴퓨터에서, 코우드는 2진수 표시(기호)의 Fixed number의 자료와 관계지워져 있다. 예를 들면, 숫자나 알파벳 문자를 표시하는데, 사용되는 코우드는 2진수 표시의 8자리를 사용한다. 2진수 표시(0 bit, 1 bit)의 독특한 배열로서, 모든 문자들을 Bit들의 다른 조합으로 나타낼 수 있게 되어 있다.

사용되고 있는 어떤 컴퓨터 코우드는 Six-bit alphameric code, Eight-bit alphameric code, Two-out-of-count code, Six-bit(packed) numeric code 등이다.

### 코우드 점검(Code Checking)

대부분의 컴퓨터 코우드는 Self-checking 기능을 가진다. 즉, 컴퓨터는 코우드화된 정보의 타당성을 점검하는 Built-in 방법을 갖추고 있다. Code checking은 컴퓨터 내에서 자료처리 작업이 수행되는 동안 자동적으로 이루어진다. 타당성 점검의 방법은 코우드 설계의 일부분이다.

어떤 코우드에 있어서, 자료의 각 단위나 문자는 1 bit의 짝수를 항상 갖고 있어야 하는 특정한 수의 Bit position으로 표시된다. 다른 문자는 1 bit들인 다른 조합으로 만들어지지만, 어떠한 확실한 문자의 1 bit의 수는 항상 짝수이다. 이러한 코우드 시스템에서는 1 bit를 홀수 개 가진 문자는 발견되며, 착오(error)로 지적된다. 마찬가지로, 홀수개의 1 bit를 가져야 하는 모든 문자를 사용하는 코우드에서는 짝수개의 1 bit를 가진 문자는 발견되고 착오가 지적된다.

이러한 형태의 점검은 Parity check로 알려져 있다. 1 bit를 짝수개 사용하는 코우드를 짝수 Parity(even parity), 홀수개 사용하는 코우드를 홀수 Parity(odd parity)로 부른다.

다른 코우드(code)에 있어서, 1 bit의 수는 각 자료의 단위가 고정되어 있음을 나타내는 것이다. 예를 들면, 한개의 코우드가 모든 문자를 코우드화하는데 8개의 Bit position을 사용할 수 있지만, 4개가 1 bit로 되어 문자를 나타내게 될 것이다. 4개가 1 bit된 것보다 많거나 적은 문자는 Error indication을 발생시킨다.

이러한 점검 조직을 Fixed-count check라고 하며, 다중처리 망(teleprocessing network)에서 자료를 전송하기 위하여 가끔 사용된다.

**Six-bit Alphameric Code**(Binary coded decimal system)

이 코우드에서는 숫자, 알파벳, 특수 문자(special character)와 같은 모든 문자들은 2진수 표기(parity bit position을 더하여)의 6개의 Position을 사용하여 표시(코우드)된다. 이러한 Position은 3개의 그룹으로 나누어진다. 하나는 Check position이고, 두번째는 Zone position, 다른 4개 bit는 Numeric position(Fig. 2-9)이다.


Fig. 2-9. Bit positions, six-bit alphameric code

이 4개의 Numeric position은 10진수 값 8, 4, 2, 1로 정해지며, Binary Coded Decimal 형태로 숫자 0~9 (Fig. 2-10)까지 나타낸다. 0은 1010으로서 실제 10진수로 10을 표시하는 것에 주의하여야 한다. B와 A Zone bit는 숫자가 0에서 9까지 표시될 때는 나타나지 않는다(00).

| Decimal Digit | Place Value 8 | 4 | 2 | 1 |
|---|---|---|---|---|
| 0 | 1 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 2 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 1 | 1 |
| 4 | 0 | 1 | 0 | 0 |
| 5 | 0 | 1 | 0 | 1 |
| 6 | 0 | 1 | 1 | 0 |
| 7 | 0 | 1 | 1 | 1 |
| 8 | 1 | 0 | 0 | 0 |
| 9 | 1 | 0 | 0 | 1 |

Fig. 2-10. Numeric bit configurations, decimal digits 0~9, Six-bit alphameric code

Zone과 Numeric bit의 조합은 알파벳과 특수 문자를 나타낸다. B와A bit는 10, 01, 11, 00의 4가지 가능한 조합을 할 수 있다.

Check bit로 알려진 C position은 단지 code checking에 사용된다. 왜냐하면, Six-bit alphameric 코우드는 항상 짝수 Parity code이므로 한 문자를 나타내는데 사용되는 Bit의 총수는 1 bit의 짝수개를 가져야 하며, 그렇지 않으면 그 문자는 무효로 취급된다. 문자를 표시하기 위하여 사용되는 Zone과 Numeric bit의 합계가 홀수일 때 한개의 1 bit가 문자에 더해질 경우에 문자 내에 있는 Bit의 수가 C bit 없이 짝수이면 C bit는 0이다.

**Standard BCD Interchange Code**

여러 종류의 컴퓨터 시스템들 간에 자료를 교환함에 있어서 융통성을 제공하기 위해 Standard BCD(binary coded decimal) interchange code가 개발되었다. 이 코우드의 구조는 기본적인 64개의 다른 문자들로 구성되어 있다.

[Fig. 2-11]은 IBM 1401, 1410, 7010, 7040, 7044 자료 처리 조직(data processing system)에서 사용되는 BCD standard interchange code를 나타낸다.

도표는 순서의 배열, 그래프, 카아드 코우드(card

| CHARACTER Report | Program | CARD CODE | BCD CODE (Core Storage) B | A | 8 | 4 | 2 | 1 |
|---|---|---|---|---|---|---|---|---|
| b | | No Punches | C | | | | | |
| . | | 12-3-8 | | B | A | 8 | | 2 | 1 |
| 口 | ) | 12-4-8 | C | B | A | 8 | 4 | | |
| [ | | 12-5-8 | | B | A | 8 | 4 | | 1 |
| ≤ | | 12-6-8 | | B | A | 8 | 4 | 2 | |
| ≠ | | 12-7-8 | C | B | A | 8 | 4 | 2 | 1 |
| & | + | 12 | C | B | A | | | | |
| $ | | 11-3-8 | C | B | | 8 | | 2 | 1 |
| * | | 11-4-8 | | B | | 8 | 4 | | |
| ] | | 11-5-8 | C | B | | 8 | 4 | | 1 |
| ; | | 11-6-8 | C | B | | 8 | 4 | 2 | |
| △ | | 11-7-8 | | B | | 8 | 4 | 2 | 1 |
| — | | 11 | | B | | | | | |
| / | | 0-1 | C | | A | | | | 1 |
| , | | 0-3-8 | C | | A | 8 | | 2 | 1 |
| % | ( | 0-4-8 | | | A | 8 | 4 | | |
| ⌒ | | 0-5-8 | C | | A | 8 | 4 | | 1 |
| \ | | 0-6-8 | C | | A | 8 | 4 | 2 | |
| # | | 0-7-8 | | | A | 8 | 4 | 2 | 1 |
| ↑ | | 2-8 | | | A | | | | |
| # | = | 3-8 | | | | 8 | | 2 | 1 |
| @ | ' | 4-8 | C | | | 8 | 4 | | |
| : | | 5-8 | | | | 8 | 4 | | 1 |
| > | | 6-8 | | | | 8 | 4 | 2 | |
| V | | 7-8 | C | | | 8 | 4 | 2 | 1 |
| ? | | 12-0 | C | B | A | 8 | | 2 | |
| A | | 12-1 | | B | A | | | | 1 |
| B | | 12-2 | | B | A | | | 2 | |
| C | | 12-3 | C | B | A | | | 2 | 1 |
| D | | 12-4 | | B | A | | 4 | | |
| E | | 12-5 | C | B | A | | 4 | | 1 |
| F | | 12-6 | C | B | A | | 4 | 2 | |
| G | | 12-7 | | B | A | | 4 | 2 | 1 |
| H | | 12-8 | | B | A | 8 | | | |
| I | | 12-9 | C | B | A | 8 | | | 1 |
| ! | | 11-0 | | B | | 8 | | 2 | |
| J | | 11-1 | C | B | | | | | 1 |
| K | | 11-2 | C | B | | | | 2 | |
| L | | 11-3 | | B | | | | 2 | 1 |
| M | | 11-4 | C | B | | | 4 | | |
| N | | 11-5 | | B | | | 4 | | 1 |
| O | | 11-6 | | B | | | 4 | 2 | |
| P | | 11-7 | C | B | | | 4 | 2 | 1 |
| Q | | 11-8 | C | B | | 8 | | | |
| R | | 11-9 | | B | | 8 | | | 1 |
| ± | | 0-2-8 | | | A | 8 | | 2 | |
| S | | 0-2 | C | | A | | | 2 | |
| T | | 0-3 | | | A | | | 2 | 1 |
| U | | 0-4 | C | | A | | 4 | | |
| V | | 0-5 | | | A | | 4 | | 1 |
| W | | 0-6 | | | A | | 4 | 2 | |
| X | | 0-7 | C | | A | | 4 | 2 | 1 |
| Y | | 0-8 | C | | A | 8 | | | |
| Z | | 0-9 | | | A | 8 | | | 1 |
| Ø | | 0 | C | | | 8 | | 2 | |
| 1 | | 1 | | | | | | | 1 |
| 2 | | 2 | | | | | | 2 | |
| 3 | | 3 | C | | | | | 2 | 1 |
| 4 | | 4 | | | | | 4 | | |
| 5 | | 5 | C | | | | 4 | | 1 |
| 6 | | 6 | C | | | | 4 | 2 | |
| 7 | | 7 | | | | | 4 | 2 | 1 |
| 8 | | 8 | | | | 8 | | | |
| 9 | | 9 | C | | | 8 | | | 1 |

Low → COLLATING SEQUENCE → High

NOTE: Tape may use even parity.

Fig. 2-11. Standard BCD interchange code

| BCD Code | Graphic Subset 1 Print Arrangement A | Graphic Subset 2 Print Arrangement H |
|---|---|---|
| 8-2-1 | # | = |
| 8-4 | @ | ' |
| A-8-4 | % | ( |
| 8-A | & | + |
| 8-A-8-4 | ¤ | ) |

Fig. 2-12. Graphic subsets 1 and 2

code), 64개의 각각 다른 비트 조합을 위한 BCD 코우드이다. Parity checking 처리를 위해 사용되는 C bit는 Standard BCD interchange code를 사용하는 특별한 컴퓨터에서만 존재한다. 만약 그 시스템이 Odd parity를 사용하면, 하나의 C bit는 Bit의 짝수 개를 갖고 있는 문자의 각 C position에 자동적으로 놓여지게 된다. 반대로 시스템이 Even parity를 사용하면 홀수 개의 Bit를 갖고 있는 문자의 C position에 C bit를 놓게 된다.

다섯 가지의 Standard BCD bit 조합은 프린터(print out)에서 사용되는 Type set에 의존하며, 두가지 다른 문자들(graphic이라 칭함)로 인쇄된다. 두가지 변화를 Graphic subset 1과 Graphic subset 2(Fig. 2-12)라고 부른다.

| SYMBOL | NAME |
|---|---|
| ‡ | Group Mark |
| ǂ | Record Mark |
| ǂǂ | Segment Mark |
| m | Word Separator |
| @ | At Sign |
| # | Number Sign |
| & | Ampersand |
| + | Plus |
| * | Asterisk |
| % | Percent |
| / | Slash |
| \ | Backslash |
| □ | Lozenge |
| b | Blank |
| ъ | Substitute Blank |
| ( | Left Parenthesis |
| ) | Right Parenthesis |
| [ | Left Bracket |
| ] | Right Bracket |
| √ | Tape Mark |
| < | Less than |
| > | Greater than |
| = | Equal to |
| ; | Semicolon |
| : | Colon |
| . | Period or Point |
| ' | Prime or Apostrophe |
| - | Minus or Hyphen (Dash) |
| Δ | Delta |

Fig. 2-13. Spacial character names

Graphic subset 1은 주로 컴퓨터의 보고서 작성과 상업적인 면에 사용되고, Graphic subset 2는 FORTRAN, COBOL과 같은 진보된 프로그램 언어(programming language)와 일반적인 수학적 부호 사용의 요구를 충족시키는데 사용된다.

〔Fig. 2-13〕은 표준 BCD interchange code에 포함된 특별한 문자들의 선택된 표준 술어를 나타낸다.

**Eight-bit Alphameric code**(Extended Binary Coded Decimal Interchange Code-EBCDIC)

이 코우드(Fig. 2-14)는 Parity checking을 위한 하나의 Position을 더해서 각 문자 구성(character format)을 위하여 여덟 개의 binary position을 사용한다.

이 8개의 Bit position을 사용함으로써, 256 가지의 각각 다른 문자를 코우드화할 수 있다. 이 코우드는 이를테면 알파벳의 대문자와 소문자의 코우딩과, 훨씬 큰 범위의 특수 문자와 어떤 입출력 장치에서 의미가 있는 많은 제어 문자(control character)를 코우딩할 수 있게 해 준다. 현재는 많은 비트 형태가 정해진 기능(제어나 graphic)을 갖고 있지는 않다. 이것은 다음에 지정을 하기 위하여 남겨두었다. EBCDIC는 System/360에 있어서 두가지 기본적인 코우딩 조직 중의 하나이다.

**Eight-bit Alphameric Code-8** (USASCII-8)

정보 교환을 위한 미국 표준 코우드(USASCII)는 기계 대 기계와 시스템 대 시스템의 통신을 단순화하고 표준화하기 위한 시도로서 통신 장비의 사용자들과 자료 처리 사업체들의 협력을 통하여 개발된 Seven-bit code이다.

왜냐하면 System/360은 Eight-bit 문자용량을 갖고 있으므로 USASCII를 Eight-bit 표시로 확장시키는 것은 필요하다. 이 확장된 표시는 IBM이 USASCII-8 (USA Standard Code for Information Interchange)로서 나타내었다. 이 코우드는 내부 처리와 USASCII가 표준화한 매체로서의 S/360으로서 입출력 처리에 사용된다.

## 2.4 수의 진법과 변환

**Binary System**

자료 표시에서 2진수 시스템을 사용하는 컴퓨터들은 IBM S/360으로 대표된다. 이러한 시스템에 있어서, 정보의 기본 단위는 Byte이다. 4 byte는 다른 시스템에서 문자나 숫자와 같이, 하나의 단위로 설명되는 32개의 연속적인 Bit position의 정보로서 구성되는 Word

| EBCDIC | Bit Configuration |
|---|---|
| NUL | 0000 0000 |
| SOH | 0000 0001 |
| STX | 0000 0010 |
| ETX | 0000 0011 |
| PF | 0000 0100 |
| HT | 0000 0101 |
| LC | 0000 0110 |
| DEL | 0000 0111 |
| | 0000 1000 |
| RLF | 0000 1001 |
| SMM | 0000 1010 |
| VT | 0000 1011 |
| FF | 0000 1100 |
| CR | 0000 1101 |
| SO | 0000 1110 |
| SI | 0000 1111 |
| DLE | 0001 0000 |
| DC1 | 0001 0001 |
| DC2 | 0001 0010 |
| TM | 0001 0011 |
| RES | 0001 0100 |
| NL | 0001 0101 |
| BS | 0001 0110 |
| IL | 0001 0111 |
| CAN | 0001 1000 |
| EM | 0001 1001 |
| CC | 0001 1010 |
| CU1 | 0001 1011 |
| IFS | 0001 1100 |
| IGS | 0001 1101 |
| IRS | 0001 1110 |
| IUS | 0001 1111 |
| DS | 0010 0000 |
| SOS | 0010 0001 |
| FS | 0010 0010 |
| | 0010 0011 |
| BYP | 0010 0100 |
| LF | 0010 0101 |
| ETB | 0010 0110 |
| ESC | 0010 0111 |
| | 0010 1000 |
| | 0010 1001 |
| SM | 0010 1010 |
| CU2 | 0010 1011 |
| | 0010 1100 |
| ENQ | 0010 1101 |
| ACK | 0010 1110 |
| BEL | 0010 1111 |
| | 0011 0000 |
| | 0011 0001 |
| SYN | 0011 0010 |
| | 0011 0011 |
| PN | 0011 0100 |
| RS | 0011 0101 |
| UC | 0011 0110 |
| EOT | 0011 0111 |
| | 0011 1000 |
| | 0011 1001 |
| | 0011 1010 |
| CU3 | 0011 1011 |
| DC4 | 0011 1100 |
| NAK | 0011 1101 |
| | 0011 1110 |
| SUB | 0011 1111 |
| SP | 0100 0000 |
| | 0100 0001 |
| | 0100 0010 |
| | 0100 0011 |
| | 0100 0100 |

| EBCDIC | Bit Configuration |
|---|---|
| | 0100 0101 |
| | 0100 0110 |
| | 0100 0111 |
| | 0100 1000 |
| . | 0100 1001 |
| ¢ [ | 0100 1010 |
| . | 0100 1011 |
| < | 0100 1100 |
| ( | 0100 1101 |
| + | 0100 1110 |
| \| | 0100 1111 |
| & | 0101 0000 |
| | 0101 0001 |
| | 0101 0010 |
| | 0101 0011 |
| | 0101 0100 |
| | 0101 0101 |
| | 0101 0110 |
| | 0101 0111 |
| | 0101 1000 |
| | 0101 1001 |
| ! ] | 0101 1010 |
| $ | 0101 1011 |
| * | 0101 1100 |
| ) | 0101 1101 |
| ; | 0101 1110 |
| ¬ | 0101 1111 |
| — | 0110 0000 |
| / | 0110 0001 |
| | 0110 0010 |
| | 0110 0011 |
| | 0110 0100 |
| | 0110 0101 |
| | 0110 0110 |
| | 0110 0111 |
| | 0110 1000 |
| | 0110 1001 |
| 7/12 | 0110 1010 |
| , | 0110 1011 |
| % | 0110 1100 |
| _ | 0110 1101 |
| > | 0110 1110 |
| ? | 0110 1111 |
| | 0111 0000 |
| | 0111 0001 |
| | 0111 0010 |
| | 0111 0011 |
| | 0111 0100 |
| | 0111 0101 |
| | 0111 0110 |
| | 0111 0111 |
| | 0111 1000 |
| 6/0 | 0111 1001 |
| : | 0111 1010 |
| # | 0111 1011 |
| @ | 0111 1100 |
| ' | 0111 1101 |
| = | 0111 1110 |
| " | 0111 1111 |
| | 1000 0000 |
| a | 1000 0001 |
| b | 1000 0010 |
| c | 1000 0011 |
| d | 1000 0100 |
| e | 1000 0101 |
| f | 1000 0110 |
| g | 1000 0111 |
| h | 1000 1000 |
| i | 1000 1001 |

| EBCDIC | Bit Configuration |
|---|---|
| | 1000 1010 |
| | 1000 1011 |
| | 1000 1100 |
| | 1000 1101 |
| | 1000 1110 |
| | 1000 1111 |
| | 1001 0000 |
| j | 1001 0001 |
| k | 1001 0010 |
| l | 1001 0011 |
| m | 1001 0100 |
| n | 1001 0101 |
| o | 1001 0110 |
| p | 1001 0111 |
| q | 1001 1000 |
| r | 1001 1001 |
| | 1001 1010 |
| | 1001 1011 |
| | 1001 1100 |
| | 1001 1101 |
| | 1001 1110 |
| | 1001 1111 |
| | 1010 0000 |
| ~ | 1010 0001 |
| s | 1010 0010 |
| t | 1010 0011 |
| u | 1010 0100 |
| v | 1010 0101 |
| w | 1010 0110 |
| x | 1010 0111 |
| y | 1010 1000 |
| z | 1010 1001 |
| | 1010 1010 |
| | 1010 1011 |
| | 1010 1100 |
| | 1010 1101 |
| | 1010 1110 |
| | 1010 1111 |
| | 1011 0000 |
| | 1011 0001 |
| | 1011 0010 |
| | 1011 0011 |
| | 1011 0100 |
| | 1011 0101 |
| | 1011 0110 |
| | 1011 0111 |
| | 1011 1000 |
| | 1011 1001 |
| | 1011 1010 |
| | 1011 1011 |
| | 1011 1100 |
| | 1011 1101 |
| | 1011 1110 |
| | 1011 1111 |
| PZ 7/11 | 1100 0000 |
| A | 1100 0001 |
| B | 1100 0010 |
| C | 1100 0011 |
| D | 1100 0100 |
| E | 1100 0101 |
| F | 1100 0110 |
| G | 1100 0111 |
| H | 1100 1000 |
| I | 1100 1001 |
| | 1100 1010 |
| | 1100 1011 |
| ⌐ | 1100 1100 |
| | 1100 1101 |
| ⌐ | 1100 1110 |

| EBCDIC | Bit Configuration |
|---|---|
| | 1100 1111 |
| MZ 7/13 | 1101 0000 |
| J | 1101 0001 |
| K | 1101 0010 |
| L | 1101 0011 |
| M | 1101 0100 |
| N | 1101 0101 |
| O | 1101 0110 |
| P | 1101 0111 |
| Q | 1101 1000 |
| R | 1101 1001 |
| | 1101 1010 |
| | 1101 1011 |
| | 1101 1100 |
| | 1101 1101 |
| | 1101 1110 |
| | 1101 1111 |
| RM 5/12 | 1110 0000 |
| | 1110 0001 |
| S | 1110 0010 |
| T | 1110 0011 |
| U | 1110 0100 |
| V | 1110 0101 |
| W | 1110 0110 |
| X | 1110 0111 |
| Y | 1110 1000 |
| Z | 1110 1001 |
| | 1110 1010 |
| | 1110 1011 |
| H | 1110 1100 |
| | 1110 1101 |
| | 1110 1110 |
| | 1110 1111 |
| 0 | 1111 0000 |
| 1 | 1111 0001 |
| 2 | 1111 0010 |
| 3 | 1111 0011 |
| 4 | 1111 0100 |
| 5 | 1111 0101 |
| 6 | 1111 0110 |
| 7 | 1111 0111 |
| 8 | 1111 1000 |
| 9 | 1111 1001 |
| ∓ | 1111 1010 |
| | 1111 1011 |
| | 1111 1100 |
| | 1111 1101 |
| | 1111 1110 |
| EO | 1111 1111 |

Fig. 2-14. Configurations, Extended Binary Coded Deicmal Interchange Code (EBCDIC)

를 구성한다.

Word 내에 있는 Bit section은 2진법 수체계와 관련된 Place significance를 가진다. 즉 Word 내에 있는 Bit의 Place position은 그 Bit의 값을 결정한다. 2진법 수 체계(binary number system)에 있어서 그 위치의 10진수 값은(오른쪽에서 왼쪽) [Fig. 2-6]에서 보이는 것처럼 **1, 2, 4, 8, 16, 32, 64** 등이다.

Word의 Bit place value는 2진법 수 체계의 값이므로 그들은 다른 2진수를 나타내는 것과 같은 방법으로 표시되거나 처리될 수 있다. 예를 들어 32 bit word(Fig. 2-15)는 하나의 32-place 2진법 숫자로서, 8자리의 16진법 숫자로서, 4개의 Alphameric 문자들로서(알파벳이나 숫자일 수 있다) 또는 프로그래머가 설정한 어떤 미리 결정된 표시로서 설명될 수 있는 것이다.

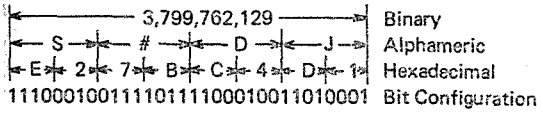



Fig. 2-15. The 32-bit word

### Octal System

등등한 숫자를 표시하는데 있어서 2진법 숫자는 10진법 숫자보다 몇 배 많은 Position을 필요로 함은 명백하다. 말하거나 쓸 때 이러한 2진법 숫자는 다루기 힘든다. 1과 0들의 긴 열은 사람들 사이에서 전달에 효과적이 될 수가 없다. 몇개로 간단히 할 수 있는 방법이 필요한데 8진법과 16진법이 이러한 요구를 충족시킨다. 2진수로서의 간단한 연결 때문에 숫자들은 점점을 해줌으로써 한 시스템에서 다른 시스템으로 바뀌어질 수 있다. Octal system의 근본이나 기초는 8이다. 이것은 **0, 1, 2, 3, 4, 5, 6, 7**의 8개의 부호가 있음을 뜻한다. 이 Number system에는 ,8, 9같은 것은 없다. 기억해야 할 중요한 관계는 3개의 Binary position은 한 개의 Octal position과 동등하다는 것이다. [Fig. 2-16]에서, 보이는 샘플(sample) 도표는 Binary, Octal, Decimal 시스템 간의 변환에 사용된다.

컴퓨터의 내부 회로는 단지 2진수 1과 0만을 알 수 있음을 기억하라. Octal system은 2진수의 읽고 쓰기에 있어서 간단한 방법을 제공하는데 사용된다. Octal system에 있어서, 기본 숫자는 8이다. 수의 각 숫자들은 8의 승멱(昇冪)의 계수를 나타내는 것이다.

Octal number를 살펴보기로 하자.

$$173=(1\times8^2)+(7\times8^1)+(3\times8^0)$$
$$= 64 + 56 + 3$$
$$=123 \text{ (decimal)}$$

비슷한 방법으로;

mal systems.

| BINARY | OCTAL | DECIMAL |
|---|---|---|
| 000 | 0 | 0 |
| 001 | 1 | 1 |
| 010 | 2 | 2 |
| 011 | 3 | 3 |
| 100 | 4 | 4 |
| 101 | 5 | 5 |
| 110 | 6 | 6 |
| 111 | 7 | 7 |

At this point, a carry to the next-higher position of the number is necessary, since all eight symbols have been used.

| BINARY | OCTAL | DECIMAL |
|---|---|---|
| 001 000 | 10 | 8 |
| 001 001 | 11 | 9 |
| 001 010 | 12 | 10 |
| 001 011 | 13 | 11 |
| 001 100 | 14 | 12 |
| . | . | . |
| . | . | . |
| . | . | . |

Fig. 2-16. Binary, octal, decimal conversion

octal 173

| | | |
|---|---|---|
| 3 | 단위 = | 3 |
| 7 | 8 = | 56 |
| 1 | 64 = | 64 |
| | | 123 |

Binary system이나 Octal system으로 표시된 수를 기억함으로써, 이 수는 위에서 보여진 방법으로 등등한 10진수로 바뀌어질 수 있다. 수가 점점 커짐에 따라 이 방법은 사용하기가 점점 어렵게 된다.

다음에서 계속하려 한 System에서 다른 System으로 변환시키는 상세한 설명을 한다.

### Integer Conversion

#### Decimal to Octal

규칙 : 10진법의 수를 8로 계속 나누고, 각 나눗셈의 단계에서 생기는 나머지들로부터 Octal number를 전개한다.

예 : Decimal number 149를 Octal 값으로 변환한다.

| | | | | |
|---|---|---|---|---|
| 8 \| 149 | 나머지 | 5 | | |
| 8 \| 18 | 나머지 | 2 | = | 225 |
| 8 \| 2 | 나머지 | 2 | | ↑ |
| 0 | | | | |

처음 바뀌어질 초기 수를 8로 나눈다. 이 나눗셈의 나머지는 변환 (5)의 Low-order 숫자로 된다. 다시 그 상(처음 나눗셈으로부터 받은)을 8로 나눈다.

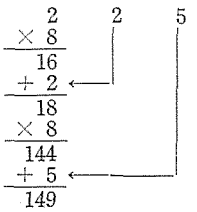
다시 그 나머지는 그 해답(next higher order 2)의 부분이 된다.

이 작업은 그 상이 나눔수 보다 작게 될 때까지 계속된다. 이 때 마지막 상은 그 변환 (2)의 High-order로 생각된다.

#### Octal to Decimal

규칙 : 계속 8을 곱하고 다음의 Octal digit를 더하라.

예 : Octal number 225를 Decimal 값으로 변환한다.




High order digit는 8로 곱해진다. 그리고 Next lower order digit는 그 결과에 더해진다. 결과로 생기는 답은 다시 8로 곱해지며 Next lower order digit는 그 결과에 더해진다. Lower order-digit가 그 답에 더하여질 때 그 작업은 끝난다. 다음의 예들에서 곱셈이나 나

놋셈이 사용되는데, 자세한 설명은 그 방식이 비슷하므로 생략하기로 한다.

**Octal to Binary와 Binary to Octal**

규칙 : 수를 3개의 binary group으로 나타낸다.

예 : Octal to Binary　　Binary to Octal

2 2 5　　010 010 101

010 010 101 =010 010 101　　2 2 5 = 225

**Decimal to Binary**

규칙 : Decimal number를 2로 나눈다. 그리고 그 나머지로부터 Binary number를 전개(develop)한다.

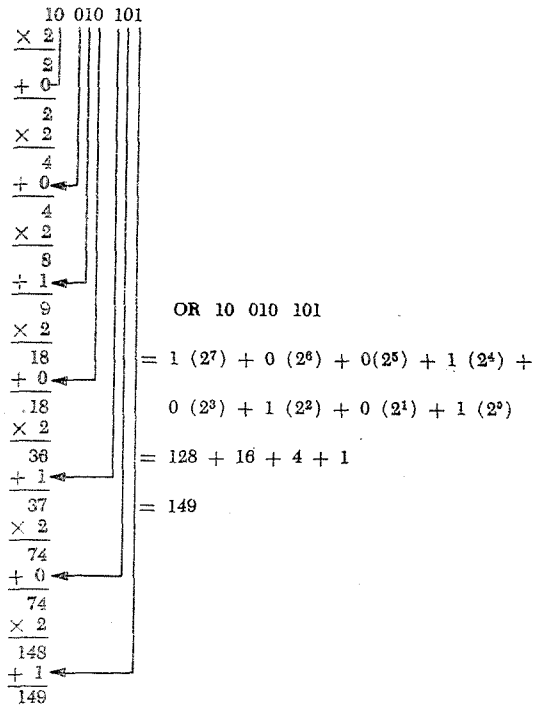
예 : 149를 2진수로 변환시켜라.

| | | | |
|---|---|---|---|
| 2 \| 149 | 나머지 | 1 | |
| 2 \| 74 | 나머지 | 0 | |
| 2 \| 37 | 나머지 | 1 | |
| 2 \| 18 | 나머지 | 0 | |
| 2 \| 9 | 나머지 | 1 | = 010 010 101 |
| 2 \| 4 | 나머지 | 0 | |
| 2 \| 2 | 나머지 | 0 | |
| 2 \| 1 | 나머지 | 1 | |
| 0 | | | |

**Binary to Decimal**

규칙 : 계속 2를 곱하고 다음 자리의 Binary digit를 더한다.

예 : 010 010 101을 10진수로 변환하여라.




**분수 변환**

**Decimal to Octal**

규칙 : 8을 곱하고 ;Carry로부터 Octal number를 전개한다.

예 :

.149
× 8
1 .192
× 8
1 .536
× 8
4 .288
× 8
2 .304
= .1142+

**Octal to Decimal**

규칙 : 8의 역지수로서 표시하고 더하라.

예 : $.1142 = 1(8^{-1}) + 1(8^{-2}) + 4(8^{-3}) + 2(8^{-4})$

$= 1/8 + 1/64 + 4/512 + 2/4096$

$= 610/4096$

$= .1489$

또는 .149

**Octal to Binary와 Binary to Octal**

규칙 : 모든 수에 관하여 같은 규칙이 분수에 적용된다.

예 :

.1 1 4 2　　.001 001 100 010

.001 001 100 010　　.1 1 4 2

**Binary to Decimal**

규칙 : 모든 수에 관하여 같은 규칙이 적용된다.

예 :

.001 001 100 010

$= 1(2^{-3}) + 1(2^{-6}) + 1(2^{-7}) + 1(2^{-11})$

$= 1/8 + 1/64 + 1/128 + 1/2048$

$= 305/2048$

$= .1489+$

또는 .149

**16진법 시스템(Hexadecimal System)**

모든 IBM 컴퓨터는 2의 곱으로서 Full word의 크기나 Field의 지정에 이르기까지 Binary system을 사용하기 때문에 S/360에서의 기본적인 사람과 기계와의 통신은 Hexadecimal numbering system으로서 한다. 예를 들어 어셈블러 프로그램(assembler program)은 보통 스토러지(storage)의 내용을 Hexadecimal 표기로서 작성하며, Literature describing operation code나 Storage format은 16진법 표기로서 정보를 준다. Hexadecimal 은 16진수를 뜻한다.

| DECIMAL SYSTEM | HEXADECIMAL SYSTEM | BINARY SYSTEM 8 4 2 1 Bit values |
|---|---|---|
| 0 | 0 | 0 0 0 0 |
| 1 | 1 | 0 0 0 1 |
| 2 | 2 | 0 0 1 0 |
| 3 | 3 | 0 0 1 1 |
| 4 | 4 | 0 1 0 0 |
| 5 | 5 | 0 1 0 1 |
| 6 | 6 | 0 1 1 0 |
| 7 | 7 | 0 1 1 1 |
| 8 | 8 | 1 0 0 0 |
| 9 | 9 | 1 0 0 1 |
| 10 | A | 1 0 1 0 |
| 11 | B | 1 0 1 1 |
| 12 | C | 1 1 0 0 |
| 13 | D | 1 1 0 1 |
| 14 | E | 1 1 1 0 |
| 15 | F | 1 1 1 1 |

Fig. 2-17. Relationship among decimal, hexadecimal, and binary systems

Hexadecimal 시스템은 16이 될 때까지 Binary bit를 계산하여 사용하며 Carry하고, 그 후에 계산을 다시 시작한다. 그렇지만 숫자로 16까지 세지는 않는다. 0에서 9까지는 숫자로 세며, 이 후에 10은 **A**, 11은 **B**, 12는 **C**, 13은 **D**, 14는 **E**, 15는 **F**의 16진법 표기로 한다.

컴퓨터 내에서 **F**(15)까지 세는 데는 4개의 Binary bit position이 필요하다. [Fig. 2-17]에서는 어떻게 쓰여지는지를 보여 준다.

System/360에서는 스트러지의 각 Byte에 8개의 비트를 가지므로, 각 Byte는 두 개의 Hexadecimal-system 숫자로서 생각된다.

예를들면 :

| | |
|---|---|
| 10진법 | 248 |
| 2진법 | 1111 1000 |
| 16진법 | F 8 |

8진법과 마찬가지로 16진법 시스템은 System/360과 같은 컴퓨터 내에서 2진법 Bit형태를 표시하는데 사용되는 간결한 Shorthand(속기) 표시이다. 이와 같이 이것은 앞서 정의한 다른 코우드 구조와 역시 관계가 있다.

Extended binary coded decimal interchange code (EBCDIC)에 있어서 이 여덟 비트 문자(eight bit character—16진법에서는 F8이라 부른다)는 숫자을 나타낸다. USASCII-8 코우드에서는 문자 'X'이다. 이것은 코우드에 따라서 다른 뜻을 갖게 되고, 코우드의 의미에 관계 없이 16진법에서는 F8로서 표시된다.

### Integer Translation, Hexadecimal to Decimal

우리가 A4B5와 같은 16진법의 수를 가지고 있다고 생각해보자. 어떻게 우리가 이 수를 10진법의 수로 바꿀 것인가?

처음에 Rightmost(low-order) position을 Position 1이라고 생각하고, 왼쪽의 다음 Position을 Position 2, 그 다음을 Position 3, Leftmost position을 Position 4라고 하자. [Fig.2-18]을 보고 다음을 주의해서 보라.

| | | |
|---|---|---|
| 5 in hex(hexadecimal) position | 1은 | 5 |
| B in hex position | 2는 | 176 |
| 4 in hex position | 3은 | 1,024 |
| A in hex position | 4는 | 40,960 |
| A 4B5의 10진수 값의 합은 | | (42,165) |

주의 : 참고 도표없이 Octal을 Decimal로, 즉 승수 8을 16으로 대체하는 것으로 앞서 설명한 것의 변환방법을 사용하라.

| HEX | DEC | HEX | DEC | HEX | DEC | HEX | DEC | HEX | DEC | HEX | DEC | HEX | DEC | HEX | DEC |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 268,435,456 | 1 | 16,777,216 | 1 | 1,048,576 | 1 | 65,536 | 1 | 4,096 | 1 | 256 | 1 | 16 | 1 | 1 |
| 2 | 536,870,912 | 2 | 33,554,432 | 2 | 2,097,152 | 2 | 131,072 | 2 | 8,192 | 2 | 512 | 2 | 32 | 2 | 2 |
| 3 | 805,306,368 | 3 | 50,331,648 | 3 | 3,145,728 | 3 | 196,608 | 3 | 12,288 | 3 | 768 | 3 | 48 | 3 | 3 |
| 4 | 1,073,741,824 | 4 | 67,108,864 | 4 | 4,194,304 | 4 | 262,144 | 4 | 16,384 | 4 | 1,024 | 4 | 64 | 4 | 4 |
| 5 | 1,342,177,280 | 5 | 83,886,080 | 5 | 5,242,880 | 5 | 327,680 | 5 | 20,480 | 5 | 1,280 | 5 | 80 | 5 | 5 |
| 6 | 1,610,612,736 | 6 | 100,663,296 | 6 | 6,291,456 | 6 | 393,216 | 6 | 24,576 | 6 | 1,536 | 6 | 96 | 6 | 6 |
| 7 | 1,879,048,192 | 7 | 117,440,512 | 7 | 7,340,032 | 7 | 458,752 | 7 | 28,672 | 7 | 1,792 | 7 | 112 | 7 | 7 |
| 8 | 2,147,483,648 | 8 | 134,217,728 | 8 | 8,388,608 | 8 | 524,288 | 8 | 32,768 | 8 | 2,048 | 8 | 128 | 8 | 8 |
| 9 | 2,415,919,104 | 9 | 150,994,944 | 9 | 9,437,184 | 9 | 589,824 | 9 | 36,864 | 9 | 2,304 | 9 | 144 | 9 | 9 |
| A | 2,684,354,560 | A | 167,772,160 | A | 10,485,760 | A | 655,360 | A | 40,960 | A | 2,560 | A | 160 | A | 10 |
| B | 2,952,790,016 | B | 184,549,376 | B | 11,534,336 | B | 720,896 | B | 45,056 | B | 2,816 | B | 176 | B | 11 |
| C | 3,221,225,472 | C | 201,326,592 | C | 12,582,912 | C | 786,432 | C | 49,152 | C | 3,072 | C | 192 | C | 12 |
| D | 3,489,660,928 | D | 218,103,808 | D | 13,631,488 | D | 851,968 | D | 53,248 | D | 3,328 | D | 208 | D | 13 |
| E | 3,758,096,384 | E | 234,881,024 | E | 14,680,064 | E | 917,504 | E | 57,344 | E | 3,584 | E | 224 | E | 14 |
| F | 4,026,531,840 | F | 251,658,240 | F | 15,728,640 | F | 983,040 | F | 61,440 | F | 3,840 | F | 240 | F | 15 |
| Hexadecimal Positions | 8 | | 7 | | 6 | | 5 | | 4 | | 3 | | 2 | | 1 |

Fig. 2-18. Hexadecimal—decimal integer conversion table

### Integer Translation, Decimal to Hexadecimal

반대의 처리로 10진수 16,428을 [Fig.2-18]에서 1,6 428보다 바로 적은 수를 보고 16진법의 수로 바꾸어라. 16진법으로서의 동가치와 Position number를 주의하라. 16,428로부터 16진법의 수의 10진수 값을 빼고 [Fig. 2-18]에서 그 나머지를 찾아보라. 그 작업 과정은 다음과 같다.

| | |
|---|---|
| 10진수의 16진법 수와의 동가치를 찾는다. | 16,428 |
| hex position 4에서의 4는 | 16,384 |
| 나머지 | 44 |
| hex position 3에서의 0은 | 0 |
| 나머지 | 44 |
| hex position 2에서의 2는 | 32 |
| 나머지 | 12 |
| hex position 1에서의 C는 | 12 |

그러므로 402C는 16,428의 hex 동가치이다.

주의 : 도표 없이 Decimal을 Octal로, 즉 8대신 16을 나눔수로 하는 같은 변환 방법을 사용하라.

### Fraction Translation, Hexadecimal to Decimal

Hex fraction을 Decimal fraction으로 바꾸기 위하여 2 fraction의 각 Position에서 Decimal equivalent의 합계를 찾아 보아라. [Fig.2-19]를 사용하라.

예 : Hex ABC를 Decimal로 바꾸라.

| | |
|---|---|
| hex position 1의 .A는 | .6250 |
| hex position 2의 .0B는 | .0429 6875 |
| hex position 3의 .00C는 | .0029 2968 7500 |
| hex ABC는 Decimal로 | .6708 9843 7500 |

주의 : 도표 없이, 각 숫자의 값을 계산하는데 있어서, Base로서 8대신 16을 대체하여, Octal decimal로 바꾸는 것과 같은 방법을 사용하라.

### Fraction Translation, Decimal to Hexadecimal

Decimal을 Hex로 바꾸기 위하여 [Fig. 2-19]에서

| H E X | 0123 DEC | H E X | 4567 DECIMAL | H E X | 0123 DECIMAL | H E X | 4567 DECIMAL EQUIVALENT |
|---|---|---|---|---|---|---|---|
| 0 | .0000 | 00 | .0000 0000 | .000 | .0000 0000 0000 | .0000 | .0000 0000 0000 0000 |
| 1 | .0625 | 01 | .0039 0625 | .001 | .0002 4414 0625 | .0001 | .0000 1525 8789 0625 |
| 2 | .1250 | 02 | .0078 1250 | .002 | .0004 8828 1250 | .0002 | .0000 3051 7578 1250 |
| 3 | .1875 | 03 | .0117 1875 | .003 | .0007 3242 1875 | .0003 | .0000 4577 6367 1875 |
| 4 | .2500 | 04 | .0156 2500 | .004 | .0009 7656 2500 | .0004 | .0000 6103 5156 2500 |
| 5 | .3125 | 05 | .0195 3125 | .005 | .0012 2070 3125 | .0005 | .0000 7629 3945 3125 |
| 6 | .3750 | 06 | .0234 3750 | .006 | .0014 6484 3750 | .0006 | .0000 9155 2734 3750 |
| 7 | .4375 | 07 | .0273 4375 | .007 | .0017 0898 4375 | .0007 | .0001 0681 1523 4375 |
| 8 | .5000 | 08 | .0312 5000 | .008 | .0019 5312 5000 | .0008 | .0001 2207 0312 5000 |
| 9 | .5625 | 09 | .0351 5625 | .009 | .0021 9726 5625 | .0009 | .0001 3732 9101 5625 |
| A | .6250 | 0A | .0390 6250 | .00A | .0024 4140 6250 | .000A | .0001 5258 7890 6250 |
| B | .6875 | 0B | .0429 6875 | .00B | .0026 8554 6875 | .000B | .0001 6784 6679 6875 |
| C | .7500 | 0C | .0468 7500 | .00C | .0029 2968 7500 | .000C | .0001 8310 5468 7500 |
| D | .8125 | 0D | .0507 8125 | .00D | .0031 7382 8125 | .000D | .0001 9836 4257 8125 |
| E | .8750 | 0E | .0546 8750 | .00E | .0034 1796 8750 | .000E | .0002 1362 3046 8750 |
| F | .9375 | 0F | .0585 9375 | .00F | .0036 6210 9375 | .000F | .0002 2888 1835 9375 |
| | 1 | | 2 | | 3 | | 4 |

Fig. 2-19. Hexadecimal—decimal fraction conversion table

Next-lower decimal 값과 그것의 Hex 동가치를 찾아라.

요구된 Decimal 값에서 찾은 Decimal 값을 빼라. 그리고 다음의 Hex 동가치를 측정하기 위해 이것을 사용하라. 요구하는 Position의 수를 찾기 위해 이 작업을 반복하라.

예 : 10진수 .13을 Hex로 바꾸라.

| | | |
|---|---|---|
| Decimal number to convert | .1300 | to Hex |
| Next-lower decimal number | .1250 | equals .2 |
| Remainder | .0050 0000 | |
| Next-lower decimal number | .0039 0625 | .01 |
| Remainder | .0010 9375 0000 | |
| Next-lower decimal number | .0009 7656 2500 | .004 |
| Remainder | .0001 1718 7500 | |
| Next-lower decimal number | .0001 0681 1523 4375 | .0007 |

.13 decimal approximately equal hex .2147

주의 : 이런 편리한 도표가 없을 때는 곱하임수로 8대신 16을 대체하여, Decimal을 octal로 바꾸는 것과 같은 방법을 사용하여라.

## 2.5 정보의 기록 매체(Recording Media)

### IBM 카아드

IBM펀치 카아드는 기계와의 통신을 위한 가장 성공적인 매개체 중의 하나이다. 정보는 표준 사이즈 카아드(Fig. 2-20)의 특정 지역에 펀치된 작은 직사각형의 구멍으로서 기록된다. 특정 지역에 구멍의 유무로서 표시(코우드)되는 정보는 카아드가 카아드 판독기를 통과할 때 읽혀지거나 감지된다.

기본적으로, 카아드를 읽고 감지하는 것은 구멍으로 기록된 자료를 전자적인 자극(electronic impulse)으로서 자동적으로 바꾸는 작업이며, 그것에 의하여 자료는 기계 내로 들어가게 된다. 이와 같이 카아드는 어떤 초기의 원천으로부터 자료를 기계로 옮기는 의미 뿐만 아니라 기계 사이에서 정보를 교환하는 일반적인 매개체 역할도 한다.

IBM 카아드는 80개의 수직 칼럼(vertical column)과 각 칼럼에 12개의 Punching position을 가지고 있다. 12개의 수평적인 줄로부터 12개의 Punching position은 카아드를 가로 지르고 있다. 하나의 칼럼에 한개나

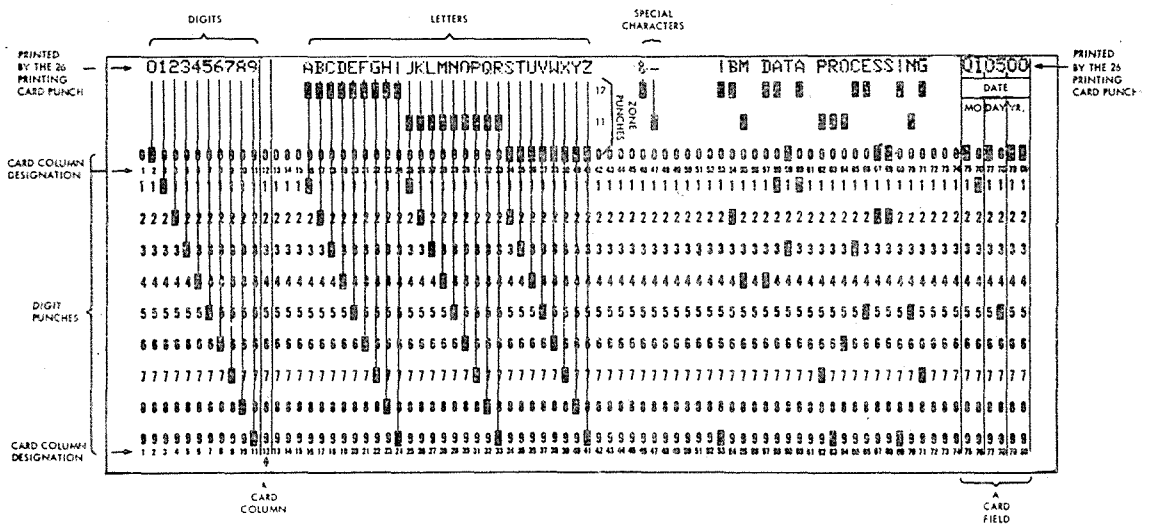



Fig. 2-20. IBM punched card, Standard hole pattern

그 이상의 펀치가 문자를 표시한다. 사용된 칼럼의 수는 표현된 자료의 양과 같다.

카아드는 종종 Unit record라 불려진다. 왜냐하면 자료는 80칼럼으로 제한 받고, 카아드는 한 단위의 정보로서 읽혀지거나 펀치되기 때문이다. 그렇지만 사실상 카아드 위에 기억된 자료는 한 레코오드(record)의 부분으로 하나의 레코오드 또는 하나 이상의 레코오드로 구성될 수 있다. 만약 한 레코오드의 자료를 싣는데 80칼럼 이상이 필요하면 두장이나 그 이상의 카아드가 사용될 것이다. 한 레코오드의 카아드들 사이에 연속성은 각 카아드의 지정된 칼럼에 동일한 정보라는 펀치를 함으로써 이루어질 수 있다.

카아드에 펀치된 정보는 Card reader라고 불리는 기계로 읽거나 해석할 수 있으며, Card punch라고 불리는 기계로서 정보를 카아드에 기록할 수 있다. 자료는 사람이 조작하는 Card punch machine으로서 초기 기록을 펀치된 카아드에 옮길 수 있다.

### IBM 카아드 코우드(구멍 형태)

표준 IBM 카아드 코우드는 숫자, 알파벳, 특수문자를 표시하기 위해 카아드 위에 수직 칼럼의 12개의 펀치 가능한 Position을 사용한다(Fig.2-20).

12개의 Hole position은 숫자와 문자의 두개 구역으로 나누어진다. 카아드의 아래 가장자리로부터 처음 9개의 Hole position은 숫자의 Hole position이며, 각각 9, 8, 7, 6, 5, 4, 3, 2, 1의 정해진 값을 갖는다. 0, 11, 12의 남은 3개의 Position은 Zone position이다(0 position은 숫자와 문자의 양 position이 되는 것으로 생각된다).

0에서 9까지의 숫자는 하나의 수직 칼럼에 하나의 구멍으로서 나타난다. 예를 들어, 0은 칼럼의 0 zone position의 하나의 구멍으로 표시된다.

알파벳 문자는 하나의 수직 칼럼에 두개의 구멍으로 표시된다. 하나는 Numeric hole이고, 하나는 zone hole이다. A에서 I까지의 알파벳 문자는 각각 12번째의 Zone hole과 1에서 9까지의 Numeric hole을 사용한다. J에서 R까지는 11번째의 Zone hole과 1에서 9까지의 Numeric hole을, S에서 Z까지는 0 zone hole과 2에서 9까지의 Numeric hole을 각각 사용한다.

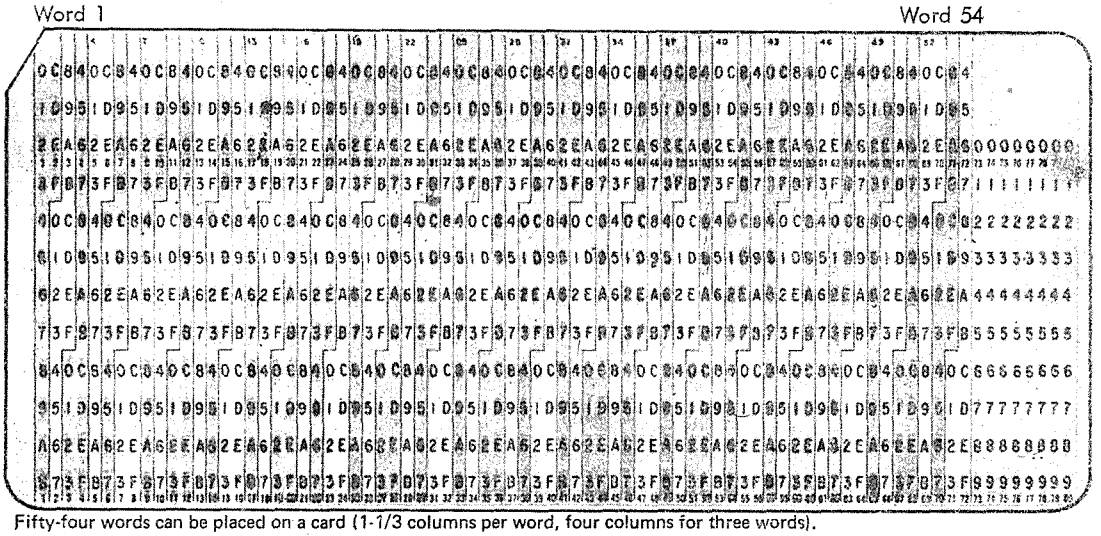
표준 특수문자 $, *, % 등은 카아드의 칼럼에서 하나나 둘 또는 세개의 구멍으로 표시되며, 숫자나 문자를 표시하기 위해 사용되지 않은 Hole의 조합으로 구성된다.

### Column Binary Data Representation

Column Binary는 카아드 위에 Binary 정보를 기록하는 한가지 방법을 가지고 있다. 이 시스템에서 정보는 Word로 배열되며, 12번째 줄에서 시작하여 칼럼의 아래로 내려 간다. 각 펀치는 그 Word position에 1개의 Binary를 표시한다. IBM 1130 전산 조직은 Column binary feature를 사용한다. 각 Word는 16 bit로 구성되며, 1 word 당 $1\frac{1}{3}$ 칼럼을 사용한다. [Fig. 2-21]은 IBM 1130 column binary card의 예이다.

### 종이 테이프(Paper Tape)

펀치된 종이 테이프(paper tape)는 펀치 카아드와 거




Fifty-four words can be placed on a card (1-1/3 columns per word, four columns for three words).
The word numbers appear in every third column across the top of the card.

Fig. 2-21. IBM 1130 binary card

의 같은 목적을 채워 준다. 전선을 통한 Telegraph message의 전달을 위해 개발된 종이 테이프는 역시 자료처리 전달을 위해 사용된다. 원거리 전송에서 기계는 카아드로부터 자료를 바꾸어 Keyboard로 Paper tape에 구멍을 뚫어서 전선의 다른 한쪽 끝에서 똑 같은 Paper tape를 얻기 위해서 전화나 Telegraph wire를 통하여 정보를 보내며, 다음의 처리를 위해 그 정보를 Punched card에 다시 바꾸게 된다.

데이터는 특정한 배열의 Punched hole로서 기록되며, 이 Punched hole은 Paper tape(Fig. 2-22, 2-23)의 길이를 따라 정밀하게 배열된다. Tape는 그 길이가 고정된 카아드에 비교하여 계속적인 기록을 할 수 있는 매체이다. 이와 같이 Paper tape는 어떠한 길이의 Record이던지 데이터를 기록하는데 사용되며, 단지 데이터가 들어가거나 또 자료가 받아들여지게 될 스토러지(storage)의 용량에 의해서만 제한 받게 된다.

Paper tape에 펀치된 데이터는 Paper tape reader로 읽거나 해석되어 지며, Paper tape punch로 기록한다.

**Eight-channel Code**(hole pattern)

데이터는 Paper tape의 길이를 따라서 여덟 개의 평행의 채널(channel)에 놓여진 구멍으로서 읽혀지거나 기록(펀치)된다.

테이프의 폭을 가로지르는 여덟 개의 가능한 펀치 Position(각 채널당 하나)의 한 칼럼은 숫자, 알파벳, 특수 문자, Function character를 코우드화하는데 사용된다. [Fig. 2-22]는 여덟 개의 채널과 수 개의 코우드화된 문자를 가진 Paper tape를 보여준다.

테이프 아랫 부분의 네개의 채널(공급 구멍을 포함하여)은 1, 2, 4, 8로 명칭을 붙이며, 숫자를 기록하는데 사용한다. 숫자 0에서 9까지는 이 네개의 Position에 하나의 펀치나 펀치들로 표현된다. 이 Position value의 합은 그 숫자의 수값을 정하게 된다. 예를 들어 1번 채널 하나의 구멍은 숫자 1을 나타내며, 1번과 2번 채널 구멍의 합은 숫자 3을 나타낸다.

X와 O채널은 IBM 펀치 카아드에서의 Zone punch와 비슷하다. 이 채널들은 알파벳과 특수 문자를 기록하기 위해서 Numeric channel과 합하여 사용된다. 알파벳과 특수 문자의 코우딩은 [Fig. 2-22]에서 보여준다.

각 문자가 정확하게 기록되었는지를 점검하기 위해서 테이프의 각 칼럼은 홀수 개의 구멍으로 펀치된다. Check hole은 그것의 기본적인 코우드(X, 0, 8, 4, 2, 1)가 짝수 개의 구멍으로 된 어떤 칼럼의 Check channel에서 반드시 나타난다.

EL(end-of-line) channel의 펀치는 테이프에서 End of record를 표시하는데 사용되는 Special function character이다. Tape feed code는 X, 0, 8, 4, 2, 1 채널에 펀치되며, Blank character position을 지정하는데 사용된다. Paper tape reader는 Tape feed code가 펀치된 테이프 에리어를 자동적으로 건너 뛴다.

**Five-channel Code**(hole pattern)

데이터는 Paper tape의 길이를 따라서 다섯 개의 평행 채널들에 구멍으로서 기록(펀치)되거나 읽혀 진다. 다섯 개의 가능한 Punching position(각 채널의 하나)

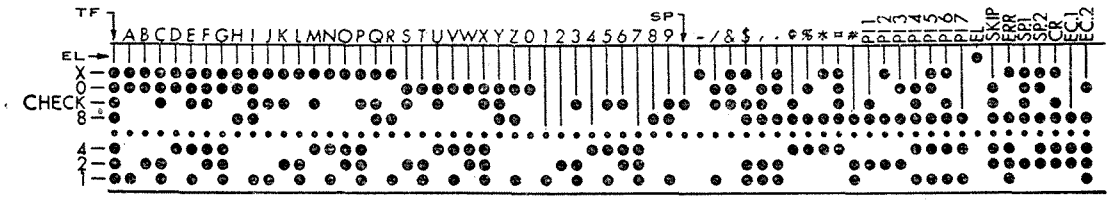



Fig. 2-22. Paper tape, eight-channel code

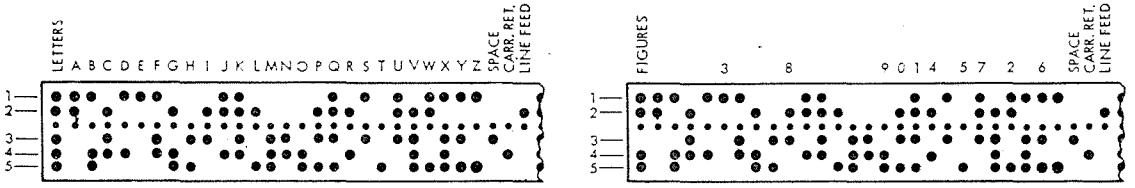



Fig. 2-23. Paper tape, five-channel code

의 한 칼럼은 테이프의 폭을 가로 지르며, 이것은 숫자, 알파벳, 특수 문자, Function character를 코우드하는데 사용된다. [Fig. 2-23]은 5개의 채널을 가진 Paper tape의 부분과 몇 개의 코우드된 문자들을 보여준다.

다섯 개의 Punching position을 사용할 때에는 단지 32개의 가능한 펀치의 조합 밖에 없기 때문에 Shift system이 이용할 수 있는 코우드의 수를 배로 하기 위하여 사용된다. Letters(LTRS) code punch가 테이프의 Section 앞에 나타났을 때, 뒤따라 나타나는 문자들은 알파벳 문자(Fig. 2-23)로 해석된다. Figures(FIGS) code punch가 테이프의 Section 앞에 나타났을 때에는 코우드된 펀치들은 숫자나 특수 문자로 해석된다.

32개 문자들 중의 10개는 각각 알파벳 P, Q, W, E, R, T, Y, U, I, O와 10진수 0에서 9까지의 양쪽 Coding으로 사용된다.

번역은 이러한 문자 앞에 나타나는 LTRS나 FIGS 같은 Shift code에 따른다. 마찬가지로 특수 문자를 위한 코우드도 다른 알파벳 코우드와 같다. 주어진 특수 문자 코우드와 같은 사실상의 알파벳 코우드는 사용자의 요구에 따라 변하게 된다.

Space, Carriage return(CR), Line feed(LF) 같은 Function character들은 LTRS나 FIGS Shift에서는 동일하다. Space code는 테이프에 데이터가 없음을 표시하는데 사용된다. CR과 LF 문자들의 사실상의 기능은 그들이 사용하는 기계에 달려 있다.

### 마그네틱 테이프(Magnetic Tape)

마그네틱 테이프는 컴퓨팅 시스템(computing system)의 주요한 입력과 출력기록 매체 중의 하나이다. 이것은 역시 계산의 중간 결과를 저장하는데, 또는 큰 파일의 데이터 스토러지를 간결히 하는데 광범위하게 사용된다.

마그네틱 테이프 장치는 시스템으로부터 처리된 데이터를 효과적으로 아주 빠른 기록 뿐만 아니라, 데이터를 컴퓨터 시스템 안에 고속도로 받아 들일 수도 있다. 1초당 640,000개의 숫자를 고도로 정확하게 입출력할 수 있는 것이 있다.

마그네틱 테이프 장치는 컴퓨터 시스템에 있어서 입력과 출력 장치의 양쪽 기능을 한다. 이 장치는 마그네틱 테이프를 Read/Write head 를 가로 지르도록 움직여서 테이프에 정보의 실질적인 읽기와 쓰기를 수행시킨다.

정보는 Bit라 불리는 자화된 점으로서 마그네틱 테이프 위에 기록된다. 기록은 무한히 보관될 수 있으며 또는 기록된 정보는 자동적으로 지워질 수도 있고, 테이프는 계속적으로 높은 신빙성을 갖고 여러번 사용될 수도 있다.

테이프를 쉽게 취급하고 처리하기 위해서 개별적인 Reel이나 먼지 방지 Cartridge에 감아둔다. Individual reel의 테이프는 폭이 1/2인치이며 한 Reel당 길이가 2400피트(feet)까지 된다. Cartridge-containe tape는 자동적으로 꿰어진다.

IBM 2420 마그네틱 테이프 장치인 Model 7은 단순화된 자동 Threading으로서 구성 시간을 줄여준다. 10초 동안 테이프의 Free-end는 공급 Reel에서 Load point에 위치한 Takeup reel까지 테이프의 통로를 따라 감겨지면, 공백 칼럼으로 옮겨지게 된다. 꽉찬 2400피트의 Reel을 다시 감는데도 테이프가 공백 칼럼으로부터 다시 옮겨 질 수 없기 때문에, 단지 1분 밖

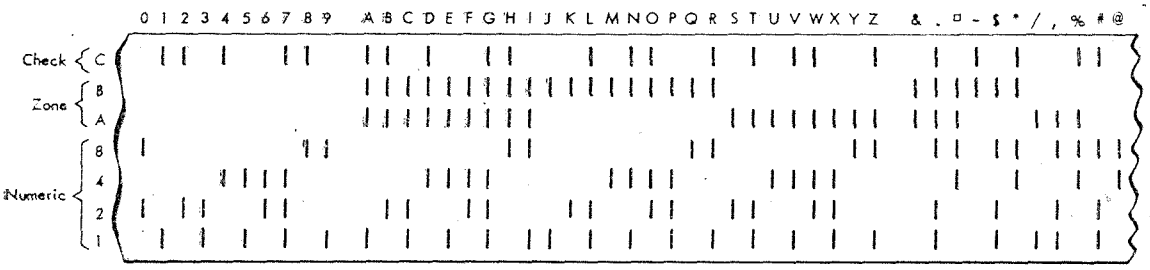



Fig. 2-24. Magnetic tape, seven-track, seven-bit Alphameric code

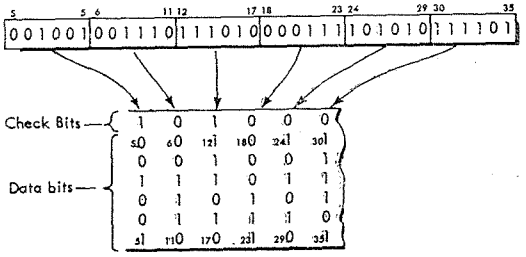



Fig. 2-25. Magnetic tape, seven-track, binary recording

에 걸리지 않는다.

데이터는 테이프의 길이에 따라서 Parallel channel이나 Track으로 기록된다. 테이프의 폭을 가로 지르는 Track들은 한 줄로 데이터를 공급한다. 수직열(vertical row) 사이의 공백은 Writing 작업동안 자동적으로 발생하며, 기록에 사용되는 문자의 밀도에 따라 변하게 된다. 인치당 1600자의 높은 문자 밀도로 이용할 수 있다.

긴 공백은 테이프 레코오드들의 Block 사이에서 자동적으로 발생한다. 이 공백을 Interblock gap(전에는 Inter-record gap이라 불렀다)이라 부른다.

IBM 컴퓨터의 데이터는 Binary coded decimal(BCD)나 Binary의 두 가지 형태의 7-track 마그네틱 테이프로 코우드된다. 코우드는 '컴퓨터가 창설한 데이터에 의하여 사용된다.

IBM 2400 series tape unit는 보통 9-track read/write head를 가지고 있다. 7-track read/write head는 필요에 따라 사용된다. 9-track head는 모델에 따라 한 개 테이프에 인치당 800~1600 byte의 밀도로 9-track 길이를 따라서 정보를 읽거나 쓴다. 각 Byte는 여덟 개의 Data bit와 하나의 Parity bit로 구성된다. 각 Bit는 두 개로 묶어지는 Decimal digit, Eight binary bit, 또는 하나의 특수 문자나 Alphameric character를 나타낸다.

묶는(packed) 방법은 기록된 10진수는 기록하는 데 제한 받지만, 이것은 어떤 전산 조직에서 숫자 데이터를 위한 읽고 쓰는 속도를 효과적으로 증복할 수 있게 하나의 Tape row에 두 개의 숫자를 기록할 수 있는 잇점이 있다.

### BCD Coding on Tape

알파벳 문자, 십진 숫자와 특수 문자들은 Binary coded decimal 코우드를 사용하여 마그네틱 테이프에 기록할 수 있다(Fig. 2-24).

### Binary Coding on Tape

몇 개의 컴퓨터 시스템은 2진수 표기로 마그네틱 테이프 위에 데이터를 기록한다(Fig. 2-25).

BCD기록에서 C track은 테이프의 읽기와 쓰기에서 정확성을 점검하는 데 사용된다. 그렇지만, Binary tape 같은 Bit의 각 줄은 1 bit의 수가 홀수 개를 가져야만 한다.

Binary 형태에 있어서의 세로 Parity check는 BCD 형태에서와 비슷하다. Record black의 각 수평 트랙에서 Bit의 전체수는 짝수이어야 한다.

### Nine-Track Coding on Tape

Series 2400 테이프 장치를 위한 Nine-track 마그네틱 테이프는 System/360 Eight-bit 코우드로서 [Fig. 2-26]에서 나타난 System/360 중앙 처리 장치(CPU)

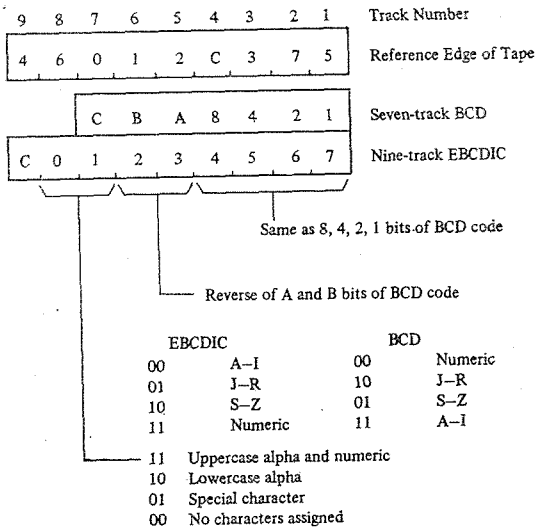



Fig. 2-27. Comparison of seven-track and nine-track alphabetic code

| Hexa-decimal | Mnemonic | Graphic & Control Symbols BCDIC | EBCDIC | Punched Card Code | System/360 8-Bit Code |
|---|---|---|---|---|---|
| 00 | | NUL | | 12-0-9-8-1 | 0000 0000 |
| 01 | | | | 12-9-1 | 0000 0001 |
| 02 | | | | 12-9-2 | 0000 0010 |
| 03 | | | | 12-9-3 | 0000 0011 |
| 04 | SPM | PF | | 12-9-4 | 0000 0100 |
| 05 | BALR | HT | | 12-9-5 | 0000 0101 |
| 06 | BCTR | LC | | 12-9-6 | 0000 0110 |
| 07 | BCR | DEL | | 12-9-7 | 0000 0111 |
| 08 | SSK | | | 12-9-8 | 0000 1000 |
| 09 | ISK | | | 12-9-8-1 | 0000 1001 |
| 0A | SVC | | | 12-9-8-2 | 0000 1010 |
| 0B | | | | 12-9-8-3 | 0000 1011 |
| 0C | (EBCDIC +) | | | 12-9-8-4 | 0000 1100 |
| 0D | (EBCDIC −) | | | 12-9-8-5 | 0000 1101 |
| 0E | | | | 12-9-8-6 | 0000 1110 |
| 0F | | | | 12-9-8-7 | 0000 1111 |
| 10 | LPR | | | 12-11-9-8-1 | 0001 0000 |
| 11 | LNR | | | 11-9-1 | 0001 0001 |
| 12 | LTR | | | 11-9-2 | 0001 0010 |
| 13 | LCR | | | 11-9-3 | 0001 0011 |
| 14 | NR | RES | | 11-9-4 | 0001 0100 |
| 15 | CLR | NL | | 11-9-5 | 0001 0101 |
| 16 | OR | BS | | 11-9-6 | 0001 0110 |
| 17 | XR | IL | | 11-9-7 | 0001 0111 |
| 18 | LR | | | 11-9-8 | 0001 1000 |
| 19 | CR | | | 11-9-8-1 | 0001 1001 |
| 1A | AR | | | 11-9-8-2 | 0001 1010 |
| 1B | SR | | | 11-9-8-3 | 0001 1011 |
| 1C | MR | | | 11-9-8-4 | 0001 1100 |
| 1D | DR | | | 11-9-8-5 | 0001 1101 |
| 1E | ALR | | | 11-9-8-6 | 0001 1110 |
| 1F | SLR | | | 11-9-8-7 | 0001 1111 |
| 20 | LPDR | | | 11-0-9-8-1 | 0010 0000 |
| 21 | LNDR | | | 0-9-1 | 0010 0001 |
| 22 | LTDR | | | 0-9-2 | 0010 0010 |
| 23 | LCDR | | | 0-9-3 | 0010 0011 |
| 24 | HDR | BYP | | 0-9-4 | 0010 0100 |
| 25 | | LF | | 0-9-5 | 0010 0101 |
| 26 | | EOB | | 0-9-6 | 0010 0110 |
| 27 | | PRE | | 0-9-7 | 0010 0111 |
| 28 | LDR | | | 0-9-8 | 0010 1000 |
| 29 | CDR | | | 0-9-8-1 | 0010 1001 |
| 2A | N ADR | SM | | 0-9-8-2 | 0010 1010 |
| 2B | N SDR | | | 0-9-8-3 | 0010 1011 |
| 2C | N MDR | | | 0-9-8-4 | 0010 1100 |
| 2D | N DDR | | | 0-9-8-5 | 0010 1101 |
| 2E | AWR | | | 0-9-8-6 | 0010 1110 |
| 2F | SWR | | | 0-9-8-7 | 0010 1111 |
| 30 | LPER | | | 12-11-0-9-8-1 | 0011 0000 |
| 31 | LNER | | | 9-1 | 0011 0001 |
| 32 | LTER | | | 9-2 | 0011 0010 |
| 33 | LCER | | | 9-3 | 0011 0011 |
| 34 | HER | PN | | 9-4 | 0011 0100 |
| 35 | | RS | | 9-5 | 0011 0101 |
| 36 | | | | 9-6 | 0011 0110 |
| 37 | | EOT | | 9-7 | 0011 0111 |
| 38 | LER | | | 9-8 | 0011 1000 |
| 39 | CER | | | 9-8-1 | 0011 1001 |
| 3A | N AER | | | 9-8-2 | 0011 1010 |
| 3B | N SER | | | 9-8-3 | 0011 1011 |
| 3C | N MER | | | 9-8-4 | 0011 1100 |
| 3D | N DER | | | 9-8-5 | 0011 1101 |
| 3E | AUR | | | 9-8-6 | 0011 1110 |
| 3F | SUR | | | 9-8-7 | 0011 1111 |
| 40 | STH | SP | | no punches | 0100 0000 |
| 41 | LA | | | 12-0-9-1 | 0100 0001 |
| 42 | STC | | | 12-0-9-2 | 0100 0010 |
| 43 | IC | | | 12-0-9-3 | 0100 0011 |
| 44 | EX | | | 12-0-9-4 | 0100 0100 |
| 45 | BAL | | | 12-0-9-5 | 0100 0101 |
| 46 | BCT | | | 12-0-9-6 | 0100 0110 |
| 47 | BC | | | 12-0-9-7 | 0100 0111 |
| 48 | LH | | | 12-0-9-8 | 0100 1000 |
| 49 | CH | | | 12-8-1 | 0100 1001 |
| 4A | AH | ¢ | | 12-8-2 | 0100 1010 |
| 4B | SH | . | | 12-8-3 | 0100 1011 |
| 4C | MH | 표 ) | < | 12-8-4 | 0100 1100 |
| 4D | | ( | ( | 12-8-5 | 0100 1101 |
| 4E | CVD | < | + | 12-8-6 | 0100 1110 |
| 4F | CVB | ≠ | \| | 12-8-7 | 0100 1111 |
| 50 | ST | & | & | 12 | 0101 0000 |
| 51 | | | | 12-11-9-1 | 0101 0001 |
| 52 | | | | 12-11-9-2 | 0101 0010 |
| 53 | | | | 12-11-9-3 | 0101 0011 |
| 54 | N | | | 12-11-9-4 | 0101 0100 |
| 55 | CL | | | 12-11-9-5 | 0101 0101 |

| Hexa-decimal | Mnemonic | Graphic & Control Symbols BCDIC | EBCDIC | Punched Card Code | System/360 8-Bit Code |
|---|---|---|---|---|---|
| 56 | O | | | 12-11-9-6 | 0101 0110 |
| 57 | X | | | 12-11-9-7 | 0101 0111 |
| 58 | L | | | 12-11-9-8 | 0101 1000 |
| 59 | C | | | 11-8-1 | 0101 1001 |
| 5A | A | | ! | 11-8-2 | 0101 1010 |
| 5B | S | $ | $ | 11-8-3 | 0101 1011 |
| 5C | M | * | * | 11-8-4 | 0101 1100 |
| 5D | D | ] | ) | 11-8-5 | 0101 1101 |
| 5E | AL | ; | ; | 11-8-6 | 0101 1110 |
| 5F | SL | △ | ¬ | 11-8-7 | 0101 1111 |
| 60 | STD | - | | 11 | 0110 0000 |
| 61 | | / | / | 0-1 | 0110 0001 |
| 62 | | | | 11-0-9-2 | 0110 0010 |
| 63 | | | | 11-0-9-3 | 0110 0011 |
| 64 | | | | 11-0-9-4 | 0110 0100 |
| 65 | | | | 11-0-9-5 | 0110 0101 |
| 66 | | | | 11-0-9-6 | 0110 0110 |
| 67 | | | | 11-0-9-7 | 0110 0111 |
| 68 | LD | | | 11-0-9-8 | 0110 1000 |
| 69 | CD | | | 0-8-1 | 0110 1001 |
| 6A | N AD | | | 12-11 | 0110 1010 |
| 6B | N SD | , | , | 0-8-3 | 0110 1011 |
| 6C | N MD | % ( | % | 0-8-4 | 0110 1100 |
| 6D | N DD | ∨ | − | 0-8-5 | 0110 1101 |
| 6E | AW | \ | > | 0-8-6 | 0110 1110 |
| 6F | SW | ## | ? | 0-8-7 | 0110 1111 |
| 70 | STE | | | 12-11-0 | 0111 0000 |
| 71 | | | | 12-11-0-9-1 | 0111 0001 |
| 72 | | | | 12-11-0-9-2 | 0111 0010 |
| 73 | | | | 12-11-0-9-3 | 0111 0011 |
| 74 | | | | 12-11-0-9-4 | 0111 0100 |
| 75 | | | | 12-11-0-9-5 | 0111 0101 |
| 76 | | | | 12-11-0-9-6 | 0111 0110 |
| 77 | | | | 12-11-0-9-7 | 0111 0111 |
| 78 | LE | | | 12-11-0-9-8 | 0111 1000 |
| 79 | CE | | | 8-1 | 0111 1001 |
| 7A | N AE | 쉼 | : | 8-2 | 0111 1010 |
| 7B | N SE | # | # | 8-3 | 0111 1011 |
| 7C | N ME | @ | @ | 8-4 | 0111 1100 |
| 7D | N DE | : | ' | 8-5 | 0111 1101 |
| 7E | AU | > | = | 8-6 | 0111 1110 |
| 7F | SU | √ | " | 8-7 | 0111 1111 |
| 80 | SSM | | | 12-0-8-1 | 1000 0000 |
| 81 | | | a | 12-0-1 | 1000 0001 |
| 82 | LPSW | | b | 12-0-2 | 1000 0010 |
| 83 | (Diagnose) | | c | 12-0-3 | 1000 0011 |
| 84 | WRD | | d | 12-0-4 | 1000 0100 |
| 85 | RDD | | e | 12-0-5 | 1000 0101 |
| 86 | BXH | | f | 12-0-6 | 1000 0110 |
| 87 | BXLE | | g | 12-0-7 | 1000 0111 |
| 88 | SRL | | h | 12-0-8 | 1000 1000 |
| 89 | SLL | | i | 12-0-9 | 1001 1001 |
| 8A | SRA | | | 12-0-8-2 | 1000 1010 |
| 8B | SLA | | | 12-0-8-3 | 1000 1011 |
| 8C | SRDL | | | 12-0-8-4 | 1000 1100 |
| 8D | SLDL | | | 12-0-8-5 | 1000 1101 |
| 8E | SRDA | | | 12-0-8-6 | 1000 1110 |
| 8F | SLDA | | | 12-0-8-7 | 1000 1111 |
| 90 | STM | | | 12-11-8-1 | 1001 0000 |
| 91 | TM | | j | 12-11-1 | 1001 0001 |
| 92 | MVI | | k | 12-11-2 | 1001 0010 |
| 93 | TS | | l | 12-11-3 | 1001 0011 |
| 94 | NI | | m | 12-11-4 | 1001 0100 |
| 95 | CLI | | n | 12-11-5 | 1001 0101 |
| 96 | OI | | o | 12-11-6 | 1001 0110 |
| 97 | XI | | p | 12-11-7 | 1001 0111 |
| 98 | LM | | q | 12-11-8 | 1001 1000 |
| 99 | | | r | 12-11-9 | 1001 1001 |
| 9A | | | | 12-11-8-2 | 1001 1010 |
| 9B | | | | 12-11-8-3 | 1001 1011 |
| 9C | SIO | | | 12-11-8-4 | 1001 1100 |
| 9D | TIO | | | 12-11-8-5 | 1001 1101 |
| 9E | HIO | | | 12-11-8-6 | 1001 1110 |
| 9F | TCH | | | 12-11-8-7 | 1001 1111 |
| A0 | | | | 11-0-8-1 | 1010 0000 |
| A1 | | | | 11-0-1 | 1010 0001 |
| A2 | | | s | 11-0-2 | 1010 0010 |
| A3 | | | t | 11-0-3 | 1010 0011 |
| A4 | | | u | 11-0-4 | 1010 0100 |
| A5 | | | v | 11-0-5 | 1010 0101 |
| A6 | | | w | 11-0-6 | 1010 0110 |
| A7 | | | x | 11-0-7 | 1010 0111 |
| A8 | | | y | 11-0-8 | 1010 1000 |
| A9 | | | z | 11-0-9 | 1010 1001 |
| AA | | | | 11-0-8-2 | 1010 1010 |

| Hexa-decimal | Mnemonic | Graphic & Control Symbols BCDIC | EBCDIC | Punched Card Code | System/360 8-Bit Code |
|---|---|---|---|---|---|
| AB | | | | 11-0-8-3 | 1010 1011 |
| AC | | | | 11-0-8-4 | 1010 1100 |
| AD | | | | 11-0-8-5 | 1010 1101 |
| AE | | | | 11-0-8-6 | 1010 1110 |
| AF | | | | 11-0-8-7 | 1010 1111 |
| B0 | | | | 12-11-0-8-1 | 1011 0000 |
| B1 | | | | 12-11-0-1 | 1011 0001 |
| B2 | | | | 12-11-0-2 | 1011 0010 |
| B3 | | | | 12-11-0-3 | 1011 0011 |
| B4 | | | | 12-11-0-4 | 1011 0100 |
| B5 | | | | 12-11-0-5 | 1011 0101 |
| B6 | | | | 12-11-0-6 | 1011 0110 |
| B7 | | | | 12-11-0-7 | 1011 0111 |
| B8 | | | | 12-11-0-8 | 1011 1000 |
| B9 | | | | 12-11-0-9 | 1011 1001 |
| BA | | | | 12-11-0-8-2 | 1011 1010 |
| BB | | | | 12-11-0-8-3 | 1011 1011 |
| BC | | | | 12-11-0-8-4 | 1011 1100 |
| BD | | | | 12-11-0-8-5 | 1011 1101 |
| BE | | | | 12-11-0-8-6 | 1011 1110 |
| BF | | | | 12-11-0-8-7 | 1011 1111 |
| C0 | | ? | | 12-0 | 1100 0000 |
| C1 | | A | A | 12-1 | 1100 0001 |
| C2 | | B | B | 12-2 | 1100 0010 |
| C3 | | C | C | 12-3 | 1100 0011 |
| C4 | | D | D | 12-4 | 1100 0100 |
| C5 | | E | E | 12-5 | 1100 0101 |
| C6 | | F | F | 12-6 | 1100 0110 |
| C7 | | G | G | 12-7 | 1100 0111 |
| C8 | | H | H | 12-8 | 1100 1000 |
| C9 | | I | I | 12-9 | 1100 1001 |
| CA | | | | 12-0-9-8-2 | 1100 1010 |
| CB | | | | 12-0-9-8-3 | 1100 1011 |
| CC | | | | 12-0-9-8-4 | 1100 1100 |
| CD | | | | 12-0-9-8-5 | 1100 1101 |
| CE | | | | 12-0-9-8-6 | 1100 1110 |
| CF | | | | 12-0-9-8-7 | 1100 1111 |
| D0 | | ! | | 11-0 | 1101 0000 |
| D1 | MVN | J | J | 11-1 | 1101 0001 |
| D2 | MVC | K | K | 11-2 | 1101 0010 |
| D3 | MVZ | L | L | 11-3 | 1101 0011 |
| D4 | NC | M | M | 11-4 | 1101 0100 |
| D5 | CLC | N | N | 11-5 | 1101 0101 |
| D6 | OC | O | O | 11-6 | 1101 0110 |
| D7 | XC | P | P | 11-7 | 1101 0111 |
| D8 | | Q | Q | 11-8 | 1101 1000 |
| D9 | | R | R | 11-9 | 1101 1001 |
| DA | | | | 12-11-9-8-2 | 1101 1010 |
| DB | | | | 12-11-9-8-3 | 1101 1011 |
| DC | TR | | | 12-11-9-8-4 | 1101 1100 |
| DD | TRT | | | 12-11-9-8-5 | 1101 1101 |
| DE | ED (4) | | | 12-11-9-8-6 | 1101 1110 |
| DF | EDMK (4) | | | 12-11-9-8-7 | 1101 1111 |
| E0 | | ‡ | | 0-8-2 | 1110 0000 |
| E1 | | | | 11-0-9-1 | 1110 0001 |
| E2 | | S | S | 0-2 | 1110 0010 |
| E3 | | T | T | 0-3 | 1110 0011 |
| E4 | | U | U | 0-4 | 1110 0100 |
| E5 | | V | V | 0-5 | 1110 0101 |
| E6 | | W | W | 0-6 | 1110 0110 |
| E7 | | X | X | 0-7 | 1110 0111 |
| E8 | | Y | Y | 0-8 | 1110 1000 |
| E9 | | Z | Z | 0-9 | 1110 1001 |
| EA | | | | 11-0-9-8-2 | 1110 1010 |
| EB | | | | 11-0-9-8-3 | 1110 1011 |
| EC | | | | 11-0-9-8-4 | 1110 1100 |
| ED | | | | 11-0-9-8-5 | 1110 1101 |
| EE | | | | 11-0-9-8-6 | 1110 1110 |
| EF | | | | 11-0-9-8-7 | 1110 1111 |
| F0 | | 0 | 0 | 0 | 1111 0000 |
| F1 | MVO | 1 | 1 | 1 | 1111 0001 |
| F2 | PACK | 2 | 2 | 2 | 1111 0010 |
| F3 | UNPK | 3 | 3 | 3 | 1111 0011 |
| F4 | | 4 | 4 | 4 | 1111 0100 |
| F5 | | 5 | 5 | 5 | 1111 0101 |
| F6 | | 6 | 6 | 6 | 1111 0110 |
| F7 | | 7 | 7 | 7 | 1111 0111 |
| F8 | ZAP (4) | 8 | 8 | 8 | 1111 1000 |
| F9 | CP (4) | 9 | 9 | 9 | 1111 1001 |
| FA | AP (4) | | | 12-11-0-9-8-2 | 1111 1010 |
| FB | SP (4) | | | 12-11-0-9-8-3 | 1111 1011 |
| FC | MP (4) | | | 12-11-0-9-8-4 | 1111 1100 |
| FD | DP (4) | | | 12-11-0-9-8-5 | 1111 1101 |
| FE | | | | 12-11-0-9-8-6 | 1111 1110 |
| FF | | | | 12-11-0-9-8-7 | 1111 1111 |

Fig. 2-26. IBM System/360 Eight-bit code

코우딩을 받아들인다.

그러나 이는 Odd-parity bit를 더하고, 문자(byte)줄에 있는 Bit의 순서를 [Fig. 2-27]과 같이 재정열한다.

Nine-track 테이프의 4~7 recording position은 Seven-track 테이프의 8, 4, 2, 1 bit position의 기능과 일치시킨다. 2와 3 recording position은 7-track의 A와 B bit position과 정반대이다.

Position 0과 1은 문자를 분류하여 네가지 분류 중의 하나에 넣는 두 개의 부가적인 기록 체널(recording channel)이다. 대문자는 알파벳과 숫자, 소문자는 알파벳, 특수 문자, 그리고 정해지지 않은 문자이다. 9-rack 테이프의 Actual channel은 0~7 순서로 동작되지 않음을 주의하라.

## Character Reading

### Magnetic Ink Character

기계 처리를 위하여 종이 매체 위에 자료를 표시하는 다른 방법은 사람과 기계가 같이 읽을 수 있는 방법, 즉 Magnetic ink character로 한다. Magnetic ink character는 [Fig. 2-28]과 같이 종이 위에 인쇄될 수 있다. 문자의 형태는 쉽게 눈으로 판단될 수 있다. 특수한 Magnetic ink는 읽음과 번역을 기계로서만 가능하게 한다.

Paper document 위에 Magnetic Ink character의 인쇄(새김)는 기계로서 한다. Document paper는 폭이 $2\frac{3}{4}$인치에서 $3\frac{2}{3}$인치, 길이가 6인치에서 $8\frac{3}{4}$인치, 두

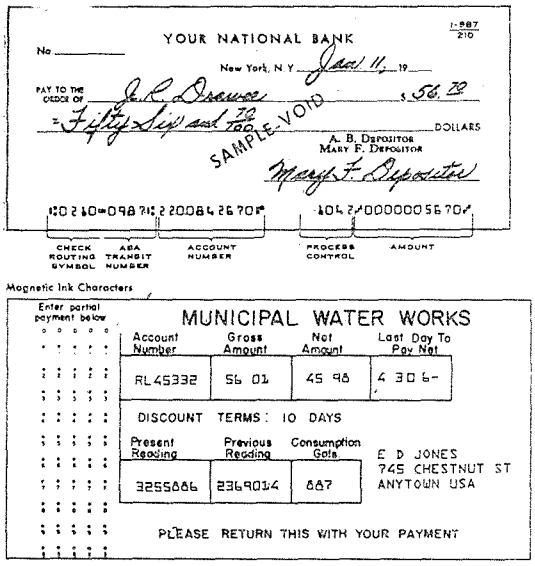



Fig. 2-28. Magnetically and optically readable characters

께가 0.003인치에서 0.007인치 범위 내에서 자유로운 크기의 종이나 카아드이다.

표준 검산까지 수행할 수 있는 IBM 1260 Electronic inscriber는 은행업무에 관련된 기능을 하며, 서류를 기재한다. 서류를 기재한 후에 IBM 1419 Magnetic character reader는 서류로부터의 기록한 정보를 읽고, 그것을 기계어로 바꾼다. 이 때, 정보는 직접 IBM 전자계산기로 들어가게 된다. IBM 1419는 서류도 역시 분류할 수 있다.

**Optically Read Character**

정보 처리 조직에서의 입력을 위한 서류(paper document) 상의 데이터 표시의 다른 방법은 광학적으로 읽을 수 있는 문자의 사용이다. [Fig. 2-28]은 1418/1428 형의 Optical reader에 받아 들여질 수 있는 몇개의 문자들을 보여 준다. 이것은 26자의 알파벳, 0에서 9까지의 숫자, 특수 문자를 포함한다.

1231 Optical mark page reader는 보통의 No. 2 연필 마크나 특정($8\frac{1}{2}\times11''$)한 Sheet에 1403이나 1443 Printer로부터의 인쇄된 Mark를 읽을 수 있다.

IBM 1285는 Cash register adding machine으로부터 산출된 것과 같은 인쇄된 Paper tape를 읽는다. Optical reading의 처음 적용은 Utility billing, 보험 할증금 통지, 위탁 판매 청구서 등이다.

1287 Optical reader는 손으로 쓴 것, 또는 기계로 인쇄된 숫자와 종이나 카아드 document로부터의 어떠한 알파벳 문자 및 메일의 테이프를 읽을 수 있다.

**Visual Output**

복잡한 정보를 다루기 위한 여러 가지의 크기, 용량,

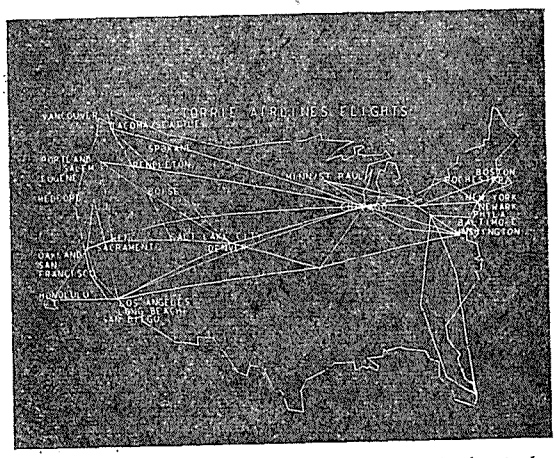



Fig. 2-29. IBM 2250 screen contents. A display including absolute vector graphics, point plotting, and both sizes of alphabetic characters.

속도 및 능력의 Visual display unit는 컴퓨터 사용자가 정상적 인쇄 방법에 의해 제작하려면 몇배의 시간이 걸리는 Graphic Report를 Cathode ray tube 위에 볼 수 있도록 해준다. 시스템 오퍼레이터 콘솔(system operator console)로서 Visual display unit의 사용은 전형적인 적용이다. 다른 것은 전화 문의로 고객의 계정 기록을 수정하고 제시하는 것이다. 기록을 직접 고치고(entry keyboard를 사용하여) 정정된 데이터를 기억 장치로 되돌아 가게 하는 것은 가능하다.

Display 장치는 Tables, Graph, Chart와 알파벳 글자 및 도형을 나타낸다(cathode ray tube screen 위에). IBM 2250 Display unit는 X와 Y좌표축에 의해 번지(address)를 지정 받을 수 있는 백만 개 이상의 점을 가지는 Display area를 가진다. 이것은 모두 12평방 인치 넓이 위에 각각 74개의 문자와 52개의 선을 표시할 수 있다. [Fig. 2-29]는 IBM 2250 위에 표시된 예를 보여준다.

IBM 2250에 연결된 IBM 2840 제어 장치는 컴퓨터로부터 1초당 238,000개의 문자까지 데이터를 받아들이고 저장한다. 1초당 60,000개의 문자나 선들도 표시될 수 있다. 단지 Line의 끝점의 지정으로 X축과 Y축이 그려질 수 있다. 뿐만 아니라, 특수한 형태나 Line의 어떠한 각도 그릴 수 있게 한다. Point는 1점당 16.8 microsecond(100만 분의 1)의 빠른 속도로 표시될 수 있다.

2260 Display unit는 $4''\times9''$의 표시 영역 위에 80개의 문자 12줄을 표시할 수 있다.

2260에 연결된 2848 제어 장치는 컴퓨터로부터 1초당 2,560개의 문자의 비율로 데이터를 받아들이고 저장한다.

〈다음 호에 계속〉