IBM Field Engineering Education
Student Self-Study Course

SYSTEM/360

Introductory Programming
Book 1 – Introduction

Preface

This is Book 1 of the System/360 Introductory Programming Student Self-Study Course.

Course Contents

●	Book 1:	Introduction	R23-2933
	Book 2:	Program Control and Execution	R23-2950
	Book 3:	Fixed Point Binary Operations	R23-2957
	Book 4:	Branching, Logical and Decimal Operations	R23-2958
	Book 5:	Input/Output Operations	R23-2959

Prerequisites

- Systems experience (1400 series with tapes, 7000 series with tapes) or a basic computer concepts course.
- Books 1 through 5 of this course must be taken in sequence.

Instructions to the student and advisor

- This course is to be used by the student in accordance with the procedure in the Instructions to the Student section in Book 1 of this course.
- The course is to be administered in accordance with the procedure in the System/360 Introductory Programming Administrator Guide, Form #R23-2972.

This edition, R23-2933-1, is a revision of the preceding edition, but it does not obsolete R23-2933-0. Numerous changes of a minor nature have been made through the manual. The Introductory Programming Student Guide, R23-2975-0 is incorporated in the "Instructions to the Student" section of this edition.

Issued to: ______________________

Branch Office: ______________ No: ________

Department: ______________________

Address: ______________________

If this manual is mislaid, please return it to the above address.

Copies of this and other IBM publications can be obtained through IBM Branch Offices. Address comments concerning the content of this publication to: IBM, FE Education Planning, Dept. 911, Poughkeepsie, N. Y., 12602

Instructions to the student

This course is designed to be learned on a "self-study" basis. Because each student is different and requires a different type of presentation, there may be parts of this course that are not immediately clear. If you are confronted with this situation, consider these three courses of action:

1. Review the material immediately preceding the problem area.

2. Continue ahead and see if the area is cleared up with additional material.

3. Contact a previously trained CE or SE and ask for help. (Take this step only after steps 1 and 2 have been tried.)

Do not attempt to memorize everything that is mentioned. If you can answer the review questions at the end of the sections and can analyze the programming examples without too much difficulty, you are progressing satisfactorily.

You will probably be able to complete this course in about 48 to 56 hours. If you are a CE with considerable systems experience, you will be able to complete some sections rather rapidly. However, do not skip any material unless you are told to do so. You should not spend more than four hours a day on this course.

Don't expect this to be an easy course. The material is in a self-study format and will require much active participation on your part. Many of the blanks that you will be asked to fill in will require that you figure out a problem on scratch paper or seek additional information. Don't hesitate to refer back to the preceding material or to the reference material.

If you continually look at the answers, before trying to fill in the blanks, you won't retain the material you are learning. Make every effort to fill in the blanks before looking at the answer. However, there will be a few times when you cannot think of the answer and will have to look at the correct answer.

Read the remainder of this Instructions to the Student section before starting the actual self-study text.

Student Materials

System/360 Introductory Programming Course	
System/360 Introduction	R23-2933
System/360 Program Control and Execution	R23-2950
System/360 Fixed Point Binary Operations	R23-2957
System/360 Branching, Logical and Decimal Operations	R23-2958
System/360 Input/Output Operations	R23-2959

Reference Material

System/360 Principles of Operations	A22-6821

Course Objectives

This course is intended to prepare you for further training on the IBM System/360. At the end of this course, you should have a comprehensive knowledge of the System/360 principles of operation. You will be able to work readily with the hexadecimal numbering system. Given a program which uses the Standard Instruction Set with the Decimal Feature, you will be able to analyze it.

Detailed learning objectives are listed at the beginning of each section.

Course Description

This System/360 Introductory Programming course consists of five self-study books. It uses the System/360 Principles of Operation manual (Form A22-6821) as reference material. The course is designed to teach the Standard Instruction Set and the Decimal Feature of IBM System/360. The usual time for completing this course ranges from 48 to 56 hours. However, this is a self-study course and allows you to proceed at your own rate. As such, there is no way to state exactly how long it will take you to complete this course. You should never spend more than four hours a day on it. Therefore, you can expect to spend approximately three calendar weeks on this course.

Each of the five self-study books has an alphabetical index of the topics which they contain. In addition, a comprehensive index covering all five books is located in the front of Book 1 (this book).

You will be given two examinations. After completing the first two books, you will take a mid-course examination. In order to continue with this course, you must achieve a score of 80% or better.

If your score is less than 80%, you will have to review the material and take another quiz. The best way to review is to take the review quizzes at the end of each section in the first two books. If you have trouble with these review questions, then you will have to re-read the text material.

When you have completed all five self-study books, you will be given the final examination. To successfully complete this course, you must obtain a score of 70% or better on this quiz. Both quizzes are of the closed book, multiple-choice type. However, this is not a "memory" course. Included in the final examination will be several pages of reference material.

You have just read the description of the entire Introductory Programming course. A description of each book follows.

Description of Student Materials

Book 1: System/360 Introduction R23-2933

This self-study book contains three sections. In these sections, you will learn the numbering systems used in the System/360, the logical organization of the system, and its data formats. At the beginning of each section is a list of learning objectives. Review questions at the end of each section will help you determine if you have met the objectives of that section. You can use these review questions at any time in the course if you feel a need to review the material. This book usually takes about 8 to 9 hours to complete.

Book 2: System/360 Program Control and Execution R23-2950

This book has four sections in which you will learn the following:

1. Instruction Formats.
2. Control of the sequence in which instructions are executed.
3. System interrupts ("hardware branches").
4. Storage protection feature.

As in the first book, each section in this book has a list of learning objectives and review questions. You will probably take about 9 to 10 hours to complete this book. However, since you are to proceed at your own rate, you may take longer than this. If you do, don't become too concerned. It is not an objective of this course to learn how to be a speed-reader. Rather, it is to learn the System/360 Principles of Operation.

When you have completed this book, you will be given the mid-course examination. This examination will test you on the learning objectives of the first two books. A score of 80% or better is required.

Book 3: System/360 Fixed Point Binary Operations R23-2957

In this book you will learn the instructions that operate on fixed point binary data. The System/360 Principles of Operation manual will be used extensively as reference material. You will first review the binary data and instruction formats. Then, you will learn how to convert IBM card data into the necessary binary data formats. You will study the fixed point binary instructions and the program errors that can result from improper usage. To determine your over-all understanding of the binary operations, you will be given a number of programming examples to analyze. These programming examples can be used to review the material covered. This book usually takes about 10 to 12 hours to complete.

Book 4: System/360 Branching, Logical and Decimal Operations R23-2958

This self-study book also makes extensive reference to the System/360 Principles of Operation manual. The branching instructions will be learned first. Then you will learn the instructions which are used to process logical and decimal data. To test your understanding of these operations, you will be required to both analyze and write a few short programs. This book will usually take about 12 hours to complete.

Book 5: System/360 Input/Output Operations R23-2959

This is the final book of your Introductory Programming course. You will learn the input/output instructions as well as the various control words used during the I/O channel operations. You will be made familiar with some of the I/O devices and with the standard interface between the I/O device and the channel. You will probably complete this book in 10 to 12 hours. You will then be given a final examination. You must obtain a score of 70% or better to successfully complete this course. This examination will test you on the contents of all five self-study books. It consists of fifty multiple-choice questions and you will be given a maximum of two hours to answer them. You will not be allowed to use any reference material other than that which is supplied with the examination. This reference material will include:

1. An alphabetical list of the instructions taught with their mnemonics, formats, and hexadecimal Op codes.
2. The formats of the System/360 control words.
3. The meaning of condition code settings.

Reference Book: System/360 Principles of Operation A22-6821

This manual is your source of reference for all information which concerns the programming aspects of the System/360 Instruction Set. This manual includes a description of each instruction as well as an appendix section which can be used for quick reference.

You will frequently be directed to the Principles of Operation manual. The areas that you are to read will not be referred to by page number. Instead, you will be given the name of the area and will have to use the contents pages of the manual in order to find the actual page numbers that you need. This is done to reduce reference errors which occur when manuals are changed.

When you finish this course, you may keep all material listed. If you go on to further System/360 training, bring the Principles of Operation manual with you.

Alphabetic Index of Books 1 through 5

This index should be used to refer you to a particular area of your self-study books so that you can review those points which are giving you trouble. The index will refer you to a page or group of pages in one of the five books.

NOTE: This index is to be used for the purpose of referring to explanatory material while taking this self-study course. After you have completed the course, it is expected that you will use the Principles of Operation manual (Form A22-6821) for reference purposes.

How to use this book

There are three sections to this text. At the beginning of each section, is a list of Learning Objectives, which you will be expected to learn as a result of studying that particular section. At the end of each section (or subsection) is a list of Review Questions so that you can evaluate your progress. You will go through this book in a serial fashion. That is, you will not be expected to skip or branch around. The answer to each frame is in the next frame. You may find it helpful to use a standard IBM card to cover the answers as you read the frames.

Periodically, as you go through this book, you will be directed to study areas of the System/360 Principles of Operation manual. This will help you to become familiar with the manual so that it may be used as reference material at a later date.

THE CONTENTS OF THIS BOOK

SECTION I   Numbering Systems

It is expected that you would be familiar with some of the numbering systems used in computers because of either your previous experience or your completion of a course in basic computer concepts. In this section you will learn the numbering systems used by the System/360. This will ensure that you are at the proper level to study the System/360 and its data formats.

SECTION II   Organization

This section will introduce you to the logical structure of the System/360. You will learn the basic units and the role they play in a System/360.

SECTION III   Data Formats

In this section you will learn the data formats used in the System/360 with the exception of the Floating Point formats. The Floating Point feature of System/360 is not covered in this self-study course.

ALPHABETICAL INDEX

System/360 Introduction

- Section I: Numbering Systems
- Section II: Organization
- Section III: Data Formats

SECTION I LEARNING OBJECTIVES

At the end of this section, you should be able to:

1. Express any decimal value from 0 to 15 as a four position binary number.

2. Express any decimal value from 0 to 15 as a one hexadecimal digit.

3. Express the complement of any decimal, binary, or hexadecimal number.

4. Add any two decimal, binary, or hexadecimal numbers.

5. Subtract via complement addition one decimal, binary, or hexadecimal number from another.

6. Convert any decimal number to a binary or hexadecimal number.

7. Convert any binary or hexadecimal number to a decimal number.

Numbering Systems

Numbering systems were developed by man so that he could count. Later the simple act of counting was expanded to the four basic mechanics of arithmetic: addition, subtraction, multiplication, and division. A number is basically a string of symbols. Such a number in the decimal system, with which you are quite familiar, is 360. Each symbol in a number has a definite place value. At this point, let's review some general rules and see how they apply to the numbering systems used by System/360.

A number is a sum of terms. Each term is a product of a digit symbol and its place value. The place value of the digit symbol is some power of the base. The power of the base starts with zero and increases by 1 from right to left.

A number is a sum of _______. Each term is a product of a _______ and its _______ _______. The decimal numbering system has a _______ of ten.

---

terms
digit
place value
base

Looking at the decimal number 360 as a sum of terms you can see that:

$$360 = 3 \times 10^2 + 6 \times 10^1 + 0 \times 10^0$$

Digit Base Exponent
or
Power of Base

TERM

This could also have been expressed in this manner:

$$360 = 3 \times 100 + 6 \times 10 + 0 \times 1$$

Notice that $10^0 = 1$. Any value to the power of 0 equals 1.

$2^0 =$ __________ $16^0 =$ __________

1
1

Another rule that you can see from the previous example is that the place value of each digit increases going from right to left. The rightmost digit of a number is called its low-order or least significant position. The leftmost digit is called its high-order or most significant position.

Example:

360

High Order or Most Significant — Low Order or Least Significant

Given the decimal number 479, express it as a sum of terms and indicate the low-order position.

479 = ____

$4 \times 10^2 \quad + \quad 7 \times 10^1 \quad + \quad 9 \times 10^0 \leftarrow$ Low Order

or

$4 \times 100 \quad + \quad 7 \times 10 \quad + \quad 9 \times 1$

Of course, you would not ordinarily express decimal numbers as sums of terms because you are too familiar with the decimal numbering system. However, numbering systems with a base other than 10 can also be expressed as a sum of terms. So as you will see, there are definite similarities between numbering systems regardless of the base.

The System/360 is capable of performing arithmetic instructions involving three different numbering systems. As part of its standard instruction set, the System/360 can do basic arithmetic with binary numbers. With the addition of the decimal feature, it can do arithmetic with binary coded decimal numbers. With the floating point feature, it can do floating point arithmetic operations with hexadecimal numbers.

List three numbering systems used by the System/360.

a. ______________ b. ______________ c. ______________

---

a. Binary
b. Decimal
c. Hexadecimal

You have been working with the decimal system most of your life. It uses the value 10 (ten) for its base. This means that each place in a decimal number represents ten raised to a power.

100,000	10,000	1,000	100	10	1

It uses ten digit symbols (0-9). Each time the highest digit value (9) is exceeded by 1 in any place of the number, the result is zero and there is a carry of 1 to the next higher place value.

Example:

09	=	0 × 10	+	9 × 1
+ 01	=	+ 0 × 10	+	1 × 1
		0 × 10	+	0 × 1
		+ 1 × 10 ← Carry		
10	=	1 × 10	+	0 × 1

The principle illustrated here is true for the other numbering systems as well. Let's see if you know the principle.

When the highest digit value is exceeded by 1, (in your own words) ____.

The result is zero and there is a carry of 1 to the next higher place value.

A numbering system other than decimal which you may be familiar with is the binary numbering system. It uses the base 2 and has only two digit symbols (0 and 1).

Express the binary number 1000 as a sum of its terms.

____

$1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 0 \times 2^0$

or

$1 \times 8 + 0 \times 4 + 0 \times 2 + 0 \times 1$

Notice that a binary number increases by the powers of 2. That is, each added place to a binary number doubles it. Binary 1000 is double binary 100. In decimal, adding a place multiples the number by ten. Decimal 1000 is ten times decimal 100.

Fill in the place values of a six-position decimal number.

Fill in the place values of a six-position binary number.

100,000	10,000	1,000	100	10	1

32	16	8	4	2	1

Express the decimal value of 25 as binary number. ______________

---

1 1 0 0 1

Which of the following is not a binary number?

a. 1011 b. 0000 c. 1200

---

c. 1200

1200 is not a valid binary number because 2 is not a valid symbol. The binary numbering system has only two valid symbols; 0 and 1.

Add 1 to binary number 1001. ______________

1010 Express the decimal values 0-15 as four-position binary numbers.

DECIMAL	BINARY
0	______
1	______
2	______
3	______
4	______
5	______
6	______
7	______
8	______
9	______
10	______
11	______
12	______
13	______
14	______
15	______

DECIMAL	BINARY
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111

Because the binary numbering system uses only two symbols (0 and 1), it is ideally suited for use in computers. Each bit position in a computer can be used to represent a Binary Digit. To represent a decimal digit, four bit positions are needed. For instance, a decimal 9 would be represented like this: 1001.

A third numbering system in use in the System/360 is the hexadecimal numbering system. The hexadecimal system uses the decimal value of 16 as its base.

16^3	16^2	16^1	16^0
4096	256	16	1

The binary system can only count as high as 1 before a carry occurs.

(1) $0 + 1 = 1$
(2) $1 + 1 = 0$ with a carry

The decimal system can count as high as 9 before a carry occurs.

(1) $8 + 1 = 9$
(2) $9 + 1 = 0$ with a carry

In the hexadecimal numbering system you can count as high as 15 before a carry occurs.

(1) $14 + 1 = 15$
(2) $15 + 1 = 0$ with a carry

To express the value 10 to 15, the symbols A to F are used. This will probably be the hardest thing for you to get used to; seeing alphabetic characters in a number.

Express the following hexadecimal number as a sum of terms:

796 = ____

$796 = 7 \times 16^2 + 9 \times 16^1 + 6 \times 16^0$

or

$7 \times 256 + 9 \times 16 + 6 \times 1$

The decimal value 112 can be expressed as the hexidecimal number 70.

Hexadecimal 70 is equal to: $7 \times 16^1 + 0 \times 16^0$

Add 1 to a hexadecimal 9. $9 + 1 =$ ____________

A

Count from a decimal 10 to a decimal 20 in hexadecimal.

Decimal	Hexadecimal
10	____
11	____
12	____
13	____
14	____
15	____
16	____
17	____
18	____
19	____

Decimal	Hexadecimal
10	A
11	B
12	C
13	D
14	E
15	F
16	10
17	11
18	12
19	13

These are the hexadecimal symbols for the decimal values 10 to 15.

Since you think most readily in decimal terms, you will find it very helpful to be able to convert from one numbering system to another. You have already expressed several small decimal values as both binary and hexadecimal numbers. It becomes more difficult as the values get larger. Fortunately, there are a few simple rules to remember for converting any number.

CONVERTING FROM DECIMAL TO HEXADECIMAL AND BINARY

Conversion Rule

1. Divide <u>entire</u> decimal number by the new base (16).
2. Remainder becomes low order of new number.
3. Divide quotient by the new base (16).
4. Remainder becomes next digit of new number.
5. Repeat steps 3 and 4 until a quotient of zero is obtained.

Decimal 456 to hexadecimal

```
              0                                    1    C    8
          16 |1  ---------->Remainder of 1---------^    ^    ^
       16 | 28   ---------->Remainder of 12 or C -------'    |
    16 |  456    ---------->Remainder of 8 ------------------'
```

Using the preceding rules, do the following problems:

Convert 972 to hexadecimal

```
16 |  972 ---->Remainder of 12 ------------+ <--Low Order
  16 |  60 ---->Remainder of 12 -------+    |
    16 |  3 ---->Remainder of 3 --+    |    |
         0                        v    v    v
                                  3    C    C
```

Convert 1248 to hexadecimal

```
16 |  1248 ------>Remainder of 0 -----------+ <--Low Order
  16 |  78 ------>Remainder of 14 ------+   |
    16 |  4 ------>Remainder of 4 --+   |   |
          0                         v   v   v
                                    4   E   0
```

Convert 247,200 to hexadecimal

```
16 |  247200 ---->Remainder of 0 ------------------+ <--Low Order
  16 |  15450 ---->Remainder of 10 ------------+   |
    16 |  965 ---->Remainder of 5 ---------+   |   |
      16 |  60 ---->Remainder of 12 ---+   |   |   |
        16 |  3 ---->Remainder of 3 -+ |   |   |   |
              0                      v v   v   v   v
                                     3 C   5   A   0
```

The rules for converting from decimal apply to binary as well as hexadecimal. The only difference is that the "new base" is 2 rather than 16.

Convert decimal 47 to binary

```
2 |  47 ---->Remainder of 1 ---------------+ <--Low Order
 2 |  23 ---->Remainder of 1 ------------+ |
  2 |  11 ---->Remainder of 1 ---------+ | |
   2 |  5 ---->Remainder of 1 ------+  | | |
    2 |  2 ---->Remainder of 0 ---+ |  | | |
     2 |  1 ---->Remainder of 1 -+ | |  | | |
           0                     v v v  v v v
                                 1 0 1  1 1 1
```

As you can imagine, converting larger numbers to binary would take quite some time. The usual procedure is to convert large numbers to hexadecimal. Then the hexadecimal number is easily converted to binary. The base of the binary system is 2^1 while the hexadecimal system uses a base of 16 or 2^4. You can see that there is a direct 4-to-1 relationship (2^4 to 2^1) between the two bases. Every hexadecimal digit becomes four binary digits. Every four binary digits in turn can be converted to a single hexadecimal digit.

Convert hexadecimal 4E0 to binary

4	E	0
↓	↓	↓
0100	1110	0000

Convert binary 010011100000 to hexadecimal

0100	1110	0000
↓	↓	↓
4	E	0

Besides using the hexadecimal numbering system for floating point calculations, the System/360 also uses the hex system in most printed material to express long binary numbers. An example of this is expressing the 24-bit binary addresses of main storage as six hexadecimal digits. The six hex digits can be easily converted to binary if it is necessary to find the actual machine language address.

"Hex" Address ——————→ 0 0 0 4 E C

Binary Address ——→ 0000 0000 0000 0100 1110 1100

If it is desired to find the decimal byte location, the hex address can be converted to decimal.

Prior to seeing how to convert from hexadecimal back to decimal, let's do another conversion problem.

Convert decimal 147,332 to binary by first converting to hexadecimal.

```
16 |_147332 ---------> Remainder of 4 <--------- Low Order
     |_9208 ---------> Remainder of 8
       |_575 --------> Remainder of 15
         |_35 -------> Remainder of 3
           |_2 ------> Remainder of 2
             0
```

Hex	2	3	F	8	4
↓	↓	↓	↓	↓	↓
Binary	0010	0011	1111	1000	0100

CONVERTING FROM HEXADECIMAL TO DECIMAL

Conversion Rules

1. Multiply the high-order digit of the number by the old base (16).
2. Add next digit to product.
3. Multiply sum by the old base (16).
4. Repeat steps 2 and 3.
5. Stop at step 2 when low-order digit has been added.

"Hex" 1C8 to decimal

```
    1      C      8
 × 16      |      |
   16      |      |
 + 12 <----+      |
   28             |
 × 16             |
  168             |
   28             |
  448             |
 +  8 <-----------+
  456
```

Using the preceding rules, do the following hexadecimal to decimal conversion problems.

Convert 3CC to decimal

972

Convert 4E0 to decimal

```
      3   C   C
×    16   |   |
     48   |   |
+    12 <-'   |
     60       |
×    16       |
    360       |
     60       |
    960       |
     12 <-----'
    972
```

1248

Convert 3C5A0 to decimal

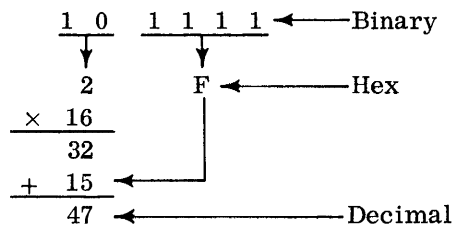
```
      4   E   0
×    16   |   |
     64   |   |
+    14 <-'   |
     78       |
×    16       |
    468       |
     78       |
   1248       |
+     0 <-----'
   1248
```

247,200

The rules for converting to decimal apply to binary numbers as well as hex numbers. The only difference is that the old base is 2 rather than 16.

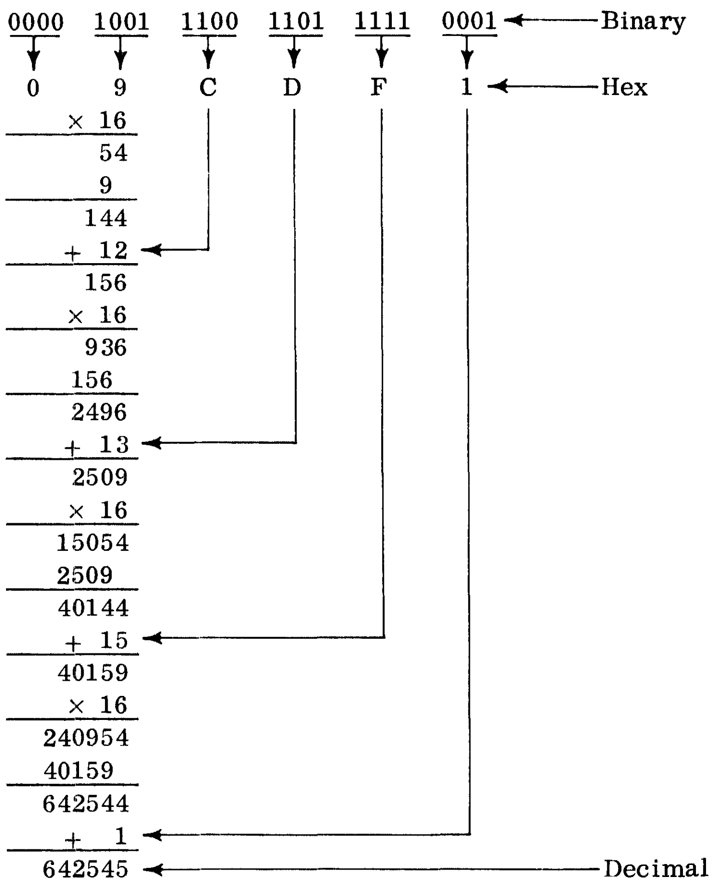
```
Convert 101111 to decimal
1   0   1   1   1   1
× 2 |   |   |   |   |
  2 |   |   |   |   |
+ 0 <-' |   |   |   |
  2     |   |   |   |
× 2     |   |   |   |
  4     |   |   |   |
+ 1 <---'   |   |   |
  5         |   |   |
× 2         |   |   |
 10         |   |   |
+ 1 <-------'   |   |
 11             |   |
× 2             |   |
 22             |   |
+ 1 <-----------'   |
 23                 |
× 2                 |
 46                 |
+ 1 <---------------'
 47
```

As you can see, the direct conversion from binary to decimal can be rather lengthy. It is much better to convert from binary to hexadecimal and then to decimal.



Given the following 24 bit binary address, what is the decimal byte location?

0 0 0 0 1 0 0 1 1 1 0 0 1 1 0 1 1 1 1 1 0 0 0 1



Conversion of small decimal and hexadecimal numbers can also be accomplished by using a reference table. Go to the IBM System/360 Principles of Operation manual and briefly study the Hexadecimal-Decimal Conversion Table that is located in the Appendix.

So far you know how to count in decimal, binary, or hexadecimal. You can also convert from one numbering system to another. But can you add or subtract with these numbers? Keep going and you will find out.

You certainly can add and subtract with decimal numbers. But let's review some of the rules of algebra concerning signed numbers.

When adding two numbers with like signs, the numbers are a________ and the s______ is retained.

When adding two numbers with unlike signs, s__________ the smaller number from the larger and use the sign of the l________ number.

When subtracting A from B, c________ the sign of _____. Now follow the rules of addition.

---

added sign subtract larger change A	Algebraic Rules:

1. Subtract +17 from +51

```
 +51
 -17
 ---
 +34
```

2. Subtract -17 from +51

```
 +51
 +17
 ---
 +68
```

3. Subtract -51 from +17

```
 +17
 +51
 ---
 +68
```

4. Subtract +51 from +17

```
 +17
 -51
 ---
 -34
```

ADDITION OF BINARY AND HEXADECIMAL NUMBERS

Addition is similar to counting. The following illustrates a principle that YOU learned earlier.

Add 1 to a binary 1. ____________________

Add 1 to a hex 9. ____________________

Add 1 to a hex F. ____________________

Now add 2 to a hex F. ____________________

---

0 with a carry of 1
A
0 with a carry of 1
1 with a carry of 1

Adding is a fast method of counting. Of course, with binary numbers, adding is very simple. The following illustrates the rules of binary addition:

```
    0 + 0 = 0
    1 + 0 = 1
    0 + 1 = 1
    1 + 1 = 0 with a carry of 1
1 + 1 + 1 = 1 with a carry of 1
```

Do the following binary additions:

```
a.  1001     b.  0111     c.  1111
    0111         0101         1111
    ----         ----         ----
```

---

a. 0000 with a carry of 1 out of the high order.
b. 1100
c. 1110 with a carry of 1 out of the high order

Since the binary numbering system uses only two symbols, it is easy to state all the possible rules of binary addition.

a. 0 + 0 = ______
b. 1 + 0 = ______
c. 0 + 1 = ______
d. 1 + 1 = ______
e. 1 + 1 + 1 = ______

---

a. 0
b. 1
c. 1
d. 0 with a carry
e. 1 with a carry

Hexadecimal arithmetic has too many possible conditions because it has 16 different symbols. However, hexadecimal digits are added just like decimal digits. The difference is that hex addition doesn't result in a carry until the decimal value of 15 is exceeded.

Decimal ——→ 9 + 6 = 15
Hex ——→ 9 + 6 = F
Hex ——→ 9 + 7 = 10

Do the following hexadecimal additions:

a. 9 + 5 = ________
b. 8 + 7 = ________
c. 8 + 8 = ________
d. A + 5 = ________
e. A + A = ________
f. B + B = ________
g. F + F = ________

SUBTRACTION OF BINARY AND HEXADECIMAL NUMBERS

---

a. E
b. F
c. 10
d. F
e. 14
f. 16
g. 1E

Just as addition is a form of counting, subtraction is a form of <u>dis</u>counting or counting backwards.

In binary, 1 + 0 = 1. Therefore, if 1 is taken away from 1, the result must be 0 (1 - 1 = 0). In like manner, 1 + 1 = 0 with a carry of one. Therefore, if 1 is taken away from 0, a 1 must be borrowed from the next digit. Then the result will be 1.

Example:

```
   01      10
 + 01    - 01
 ----    ----
   10      01
```

In like manner, all rules of binary subtraction may be derived from those of binary addition.

a. 0 + 0 = ______
b. 0 - 0 = ______
c. 0 + 1 = ______
d. 1 - 0 = ______
e. 1 - 1 = ______
f. 1 + 1 = ______
g. 0 - 1 = ______

---

a. 0
b. 0
c. 1
d. 1
e. 0
f. 0 with a carry
g. 1 with a borrow

Earlier you reviewed the rules for adding or subtracting signed numbers. Using those rules, do the following problems in binary addition and subtraction.

```
   +1001
 + +0101
 -------
```

+1110

```
   +1001
 + -0101
 -------
```

+0100 Notice that in the last problem you had to borrow from the high-order position. In doing so, it left the high order with a 0.

```
 _ +1001
   +0101
 ________
```

+0100 Here is the rule that should have been used: Change the sign of A and follow the rules of addition. Once the sign of A has been changed, the problem becomes identical to the problem preceding it.

```
   +0101
 + -1001
 ________
```

-0100 This particular problem involving adding unlike signs was solved by subtracting the smaller from the larger and using the sign of the larger like this:

```
 _(-) 1001
      0101
 __________
     -0100
```

Subtraction in hexadecimal is just like decimal subtraction. However, whenever you borrow from the high order, you are borrowing 16 rather than 10.

Example:

```
Decimal
  17                0 (17)
- 08   becomes     -0   8
                     09

Hexadecimal
  17                0 (23) <--- decimal value
- 08   becomes     -0   8
                       0F
```

Do the following hexadecimal subtraction problems:

```
a.    9A          b.    AB
     -57               -9A
     ___               ___
```

---

a. 43
b. 11

Just as a check on the previous problem, let's convert both numbers to decimal, subtract decimally, and convert the result back to hexadecimal.

1st Number	2nd Number
A B	9 A
× 16	× 16
160	144
+ 11	+ 10
171	154

Subtraction

171
\- 154
017

Reconversion

16 | 017 Remainder of 1
1 Remainder of 1
0

Final Result = 1 1

---

a. F9F
\- A8F

b. F9A
\-A8F

---

a. 510
b. 50B

In problem b of the preceding frame, you had to borrow in order to subtract a hex F from the units position. Since you borrowed 16, the problem became this:

F	8	(26)	← Decimal Value
A	8	F	
5	0	B	

F0A
\-A8F

47B

That was a real toughie! You had to borrow from two places over to subtract from the units position. It worked like this:

```
                          Decimal Value
                        ↙             ↙
 F0A  ──────►  E(16)A  ──────►  EF(26)
- A8F         - A  8  F         - A8F
                                  47B
```

Actually this is just like <u>decimal</u> borrowing; for example:

<u>Decimal</u>

```
 807 ──────►  7 (10) 7 ──────► 79(17)
- 128        -1   2  8         - 12 8
                                 679
```

You may have had trouble with the last hexadecimal problem, so do another just like it.

```
  D00A
- 170B
```

B8FF

If you followed the proper procedure for borrowing, you should have arrived at the correct answer. Let's make sure you know the proper procedure for borrowing from hexadecimal numbers.

Each time you borrow 1 from a hex digit, you bring over a decimal value of ______.

16

Let's check the answer to the last hexadecimal subtraction problem by doing it in decimal.

1. Convert operands to decimal

```
   D  0  0  A            1  7  0  B
 × 16                  × 16
  208                     16
 +  0 <-- (0)          +   7 <-- (7)
  208                     23
 × 16                  × 16
 3328                    368
 +  0 <-- (0)          +   0 <-- (0)
 3328                    368
 × 16                  × 16
53248                   5888
 + 10 <-- (A)          +  11 <-- (B)
53258                   5899
```

2. Subtract decimally

```
  53258
 - 5899
  47359
```

3. Convert result to hexadecimal

```
16 | 47359 ---> Remainder of 15
   |  2959 ---> Remainder of 15
   |   184 ---> Remainder of 8
   |    11 ---> Remainder of 11
         0
                              B  8  F  F
```

You have not only checked the validity of the answer; you have also seen that you can solve hexadecimal calculations by converting the numbers to decimal. The choice of using decimal or hexadecimal to solve problems is yours. Normally it is faster to solve hex problems in hexadecimal.

So far you have been adding and subtracting signed and unsigned numbers. Actually you should realize by now that unsigned numbers are treated mathematically as if they had plus signs. The numbers you have been working with have been decimal, binary, and hexadecimal numbers. Before you can continue and can see the data formats used in the System/360, there is one last item to be learned concerning numbers. That item is Complement Addition.

Complement addition is the way most computers (System/360 included) perform subtraction.

A complement number is defined as that quantity which, when added to a number, would result in a zero answer and a carry out of the high-order position.

> The quantity 544 when added to the number 456 would result in an answer of 000 with a carry out of the high-order position.

In the preceding example, the quantity 544 is considered the ____________ of 456. Conversely, 456 could be considered the ____________ of 544.

---

complement
complement

The procedure for complementing any number (decimal or otherwise) is the same. Subtract each digit of a number from the highest digit value of the numbering system and add 1 to the low-order position.

For decimal numbers, this means to subtract each digit from 9 and add 1 to the low order.
Example:

```
    To Complement 456        999
                           - 456
                             543
                           +   1
    Complement of 456 -----> 544
```

The complement of a decimal number is usually called the "tens" complement.

What is the "tens" complement of 968? ______________

032

What is the "tens" complement of 999? ______________

What is the "tens" complement of 500? ______________

What is the "tens" complement of 000? ______________

---

001 500 000 cannot be complemented	Example of why 000 cannot be complemented:

```
   999
 - 000
   999   Each digit can be subtracted from 9

   999
 +   1   However, when 1 is added to low order,
 ↙000    the result goes back to zero.
Carry
```

For complementing hexadecimal numbers, each digit is subtracted from F(15) and 1 is added to low order.

To complement a hex 1 C 8

```
   FFF
 - 1C8
   E37
 +   1
   E38 ←—— Complement of 1C8
```

The complement of a hexadecimal number is usually called the "sixteens" complement.

E38 is the "__________" complement of 1C8.

"sixteens"

What is the complement of the following hexadecimal numbers?

a. 4 E 8 __________ b. A B C __________

---

a. B 1 8
b. 544

a. D 3 A __________ b. F F F __________

---

a. 2 C 6
b. 001

a. 8 0 0 __________ b. 0 0 0 __________

---

a. 8 0 0 b. Just as in decimal the quantity zero cannot be complemented.	To complement binary numbers, subtract each digit from 1 and add 1 to the low order. Another way of saying this is to say: Invert each binary digit and add 1. To complement the binary number 0 0 0 1 1 1 0 0 1 0 0 0

```
  1 1 1 1 1 1 1 1 1 1 1 1
- 0 0 0 1 1 1 0 0 1 0 0 0
  -----------------------
  1 1 1 0 0 0 1 1 0 1 1 1
                    +   1
  -----------------------
  1 1 1 0 0 0 1 1 1 0 0 0
```

The complement of a binary number is called the "twos" complement.

To obtain the "twos" complement of a binary number ________ each digit and add _____ to the low order.

---

invert 1	Obtain the "twos" complement of the following binary numbers: a. 0 0 1 0 1 1 1 1 b. 1 1 1 1 1 1 1 1 a. ______________ b. ______________

---

a. 1 1 0 1 0 0 0 1 b. 0 0 0 0 0 0 0 1	a. 1 1 0 0 0 0 0 0 b. 0 0 0 0 0 0 0 0 a. ______________ b. ______________

---

a. 0 1 0 0 0 0 0 0 b. Again the quantity zero cannot be complemented. The operation always results in a zero answer.	Now that you can obtain the complement of any number, what does this mean to you? Earlier it was stated that most computers perform subtractions by means of complement addition. This means that, instead of subtracting a number, most computers derive the correct result by <u>adding</u> the <u>complement</u> of a number. An example of this follows.

Example:

To subtract a decimal 456 from 847:

1. You do it this way:

```
  847
- 456
  391
```

2. The computer does it this way:

```
  847          847
- 456 ------> +544
              391
             /
Carry out of high order
```

Notice that the answer in both cases is the same. The act of complement addition will always result in the same answer as subtraction. Computers usually use adders in their Arithmetic and Logical Units. Subtracting by complement addition allows the computer to use its adder for both addition and subtraction. Of course, there is some subtraction involved in complementing a number. In complementing, however, a number is always subtracted from the same value (the base minus 1). This can be handled by a minimum amount of circuitry on the input to the adder.

Computers usually subtract by means of ____________ ____________.

---

complement addition

Both addition and subtraction are usually done in the computer's ALU by the _________.

---

adder

The statement was made earlier that complement addition will always result in the same answer as subtraction. The result of complement addition, however, will be in one or two forms: True or Complement. Let's take a look at the previous example and explore this further by first looking at regular subtraction.

```
                847
Subtraction   - 456
                391
```

Subtraction always gives a _______ answer.

---

true

Now let's look at complement addition.

```
Complement Addition        847
                          +544
                           391
Carry out of high order
```

By inspection, you can tell that the answer is in true form. But how can a computer tell whether it is in true or in complement form?

If you will notice, there was a carry out of the high order when complement adding in our example. This carry is a signal to the computer that the answer is in true form.

The results of complement addition may be in ______ or in __________ form.

---

true
complement

The computer is signaled that the result is in true form by (in your own words) ________________________________.

---

carry out of high order

Since a carry out of the high order indicates that the answer is true, the absence of a carry indicates (in your own words) ________________________________.

---

that the answer is in complement form

Now let's reverse our numbers and subtract 847 from 456.

```
Subtraction      456       847
                -847      -456
                          -391
```

The -391 is called the ______ answer.

---

true

```
Complement Addition      456       456
                        -847      +153
                                   609
```

The absence of a high-order carry indicates (in your own words) ________________________________.

---

that the answer is in complement form

To obtain the true answer, the computer must do two things:

1. Complement the complement answer. This is known as "Re-complementing."

2. Change the sign of the result field. In the previous example, the unsigned field (847) was considered plus. As a result, the true answer will be minus.

Complementing a complement answer is known as ______________.

---

re-complementing

Re-complementing also involves (in your own words) ______________ of the result field.

---

changing the sign

Re-complementing

Subtraction	Complement Addition	Re-complementing
456	456	999
- 847	+ 153	- 609
-391	609	390
		+ 1
		-391

Solve the following decimal subtraction problems by complement addition. Re-complement if necessary to obtain a true answer.

```
 789
-760
```

Complementing	Adding
999	789
-760	+ 240
239	C←029
+ 1	
240	

The true answer is +029. Re-complementing was not necessary.

```
 247
-821
```

Complementing	Adding	Re-complementing
999	247	999
821	+ 179	- 426
178	426	573
+ 1		+ 1
179		- 574

The true answer is -574.

In the answers to the preceding problems, the complementing of the fields and subsequent addition are shown as two separate operations. Actually, in computers the complementing is done as the field is being sent to the adder. The complementing and any subsequent re-complementing is done automatically by the computer.

You have just done some complement additions with decimal fields. Since you already know how to complement binary and hexadecimal fields, go ahead and solve the following binary problems by complement adding.

```
 1 1 1 0 1 0 0 1
-0 1 1 0 1 0 1 1
```

Complement Addition

```
   1 1 1 0 1 0 0 1
   1 0 0 1 0 1 0 1
C←0 1 1 1 1 1 1 0
```

The true answer is +0 1 1 1 1 1 1 0.

```
 0 1 0 1 1 0 0 1
-0 1 1 0 0 1 1 1
```

Complement Addition

```
   0 1 0 1 1 0 0 1
 + 1 0 0 1 1 0 0 1
   1 1 1 1 0 0 1 0
```

There was no carry, so re-complement

```
   1 1 1 1 1 1 1 1
 - 1 1 1 1 0 0 1 0
   0 0 0 0 1 1 0 1
               + 1
 - 0 0 0 0 1 1 1 0 ←—— The true answer
```

You have just done two problems of subtraction with binary numbers. You solved them by complement addition. Later on when you study data formats, you will see that the System/360 does its binary calculations in a unique fashion. For now, solve the following hexadecimal problems with complement addition.

```
  E 7 A 4
- A 4 8 E
```

Complement	Addition
F F F F	E 7 A 4
-A 4 8 E	+5 B 7 2
5 B 7 1	C ← 4 3 1 6
+ 1	
5 B 7 2	

4 3 1 6 is the true hexadecimal answer.

```
  A B C D
- E D C B
```

Complement	Addition	Re-complement
F F F F	A B C D	F F F F
-E D C B	1 2 3 5	- B E 0 2
1 2 3 4	B E 0 2	4 1 F D
+ 1		+ 1
1 2 3 5		- 4 1 F E

The true answer is - 4 1 F E.

In summary, complement addition is the method most computers use to subtract. The result of complement addition is in true form if there is a carry out of the high-order position. The absence of a carry indicates that the answer is in complement form. To obtain the true answer, the computer must re-complement the answer and change the sign.

After doing the review questions on the following pages, you will be ready to study the System/360 organization.

REVIEW QUESTIONS ON NUMBERING SYSTEMS

- Try to answer the questions without referring to the material. However, if you do require aid, refer to this book and/or the System/360 Principles of Operation manual and consider reviewing the area where aid is required.

1. Express the decimal values 0-15 as a four position binary number and as one hexadecimal digit.

Decimal	Binary	Hexadecimal
0	________	________
1	________	________
2	________	________
3	________	________
4	________	________
5	________	________
6	________	________
7	________	________
8	________	________
9	________	________
10	________	________
11	________	________
12	________	________
13	________	________
14	________	________
15	________	________

2. Write the "tens" complement of the following decimal numbers:
   a. 705 ________ b. 671 ________

3. Write the "twos" complement of the following binary numbers:
   a. 1 1 0 1 1 0 1 1 b. 1 0 0 0 0 1 1 0

   ________ ________

4. Write the "sixteens" complement of the following hexadecimal numbers:
   a. F A D E ________ b. D E A F ________

5. Add the following:

	Decimal	Binary	Hexadecimal
	a. 705	b. 1 1 0 1 1 0 1 1	c. FADE
	+ 671	+1 0 0 0 0 1 1 0	+ DEAF

6. Subtract the following using complement addition:

Decimal	Binary	Hexadecimal
a. 705 -671	b. 1 1 0 1 1 0 1 1 -1 0 0 0 0 1 1 0	c. FADE -DEAF
d. 671 -705	e. 1 0 0 0 0 1 1 0 -1 1 0 1 1 0 1 1	f. DEAF -FADE

7. Convert the following decimal numbers to hexadecimal and binary numbers:

a. 705

b. 671

8. Convert the following binary numbers to decimal numbers:

a. 1 1 0 1 1 0 1 1 b. 1 0 0 0 0 1 1 0

9. Convert the following hexadecimal numbers to decimal numbers:

a. FADE b. DEAF

10. Express the following decimal numbers in binary and hexadecimal.

Decimal	Binary	Hexadecimal
16		
70		
161		

ANSWERS TO REVIEW QUESTIONS

1.

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

2. a. 295
   b. 329

3. a. 0 0 1 0 0 1 0 1
   b. 0 1 1 1 1 0 1 0

4. a. 0522
   b. 2151

5. a. 1376
   b. 1 0 1 1 0 0 0 0 1
   c. 1D98D

6. a.
```
   705             705
  -671     =      +329
          Carry<--034<--True Answer
```

b.
```
   1 1 0 1 1 0 1 1        1 1 0 1 1 0 1 1
  -1 0 0 0 0 1 1 0   =   +0 1 1 1 1 0 1 0
                    Carry<0 1 0 1 0 1 0 1<--True Answer
```

c.
```
   FADE            FADE
  -DEAF     =     +2 1 5 1
          Carry <--1 C 2 F<--True Answer
```

6. d. 671 − 705 = 671 + 295

```
   671              671
 - 705     =      + 295
                    966 ←— Complement Answer

                  - 034 ←— True Answer
```

e.

```
   1 0 0 0 0 1 1 0          1 0 0 0 0 1 1 0
 - 1 1 0 1 1 0 1 1    =   + 0 0 1 0 0 1 0 1
                            1 0 1 0 1 0 1 1 ←— Complement Answer

                          - 0 1 0 1 0 1 0 1 ←— True Answer
```

f.

```
   DEAF              DEAF
 - FADE     =        0522
                     E3D1 ←— Complement Answer

                   - 1C2F ←— True Answer
```

7. a.

```
16 | 705 ——————————→ 1 ←— Hexadecimal
  16 | 44 ————————→ C
    16 | 2 ————→ 2
         0

0 0 1 0   1 1 0 0   0 0 0 1 ←— Binary
```

b.

```
16 | 671 ——————————→ F ←— Hexadecimal
  16 | 41 ————————→ 9
    16 | 2 ————→ 2
         0

0 0 1 0   1 0 0 1   1 1 1 1 ←— Binary
```

8. a.

```
      1 1 0 1 1 0 1 1
×     2
      2
+     1
      3
×     2
      6
+     0
      6
×     2
     12
+     1
     13
×     2
     26
+     1
     27
×     2
     54
+     0
     54
×     2
    108
+     1
    109
×     2
    218
+     1
    219
```

b. Let's do this one by first converting to hexadecimal.

```
1 0 0 0  0 1 1 0
      8        6
×    16
     48
      8
    128
+     6
    134
```

9. a.

```
      (15) (10) (13) (14)
        F    A    D    E
  ×    16
       90
      15
      240
  +    10 ← A
      250
  ×    16
     1500
     250
     4000
       13 ← D
     4013
  ×    16
    24078
    4013
    64208
  +    14 ← E
    64222
```

b.

```
      (13) (14) (10) (15)
        D    E    A    F
  ×    16
       78
      13
      208
  +    14 ← E
      222
  ×    16
     1332
     222
     3552
  +    10 ← A
     3562
  ×    16
    21372
    3562
    56992
  +    15 ← F
    57007
```

10.

Decimal	Binary	Hexadecimal
16	10000	10
70	1000110	46
161	10100001	A1

SECTION II LEARNING OBJECTIVES

At the end of this section, you should be able to:

A. With reference to main storage:

1. Define: Byte, Halfword, Word, and Doubleword.
2. State that each and every byte in main storage is individually addressable with a 24 bit (3 byte) binary address.
3. Referring to the above terms, state the boundary restrictions.

B. With reference to the General Registers:

1. State the number of general registers.
2. State that each register is addressable with a 4 bit binary address.
3. State that in some operations, an even-odd address pair of registers is used. In these cases, the high-order (even address) register is used for addressing purposes.
4. State that the registers are used to hold:
   a. Operands (Accumulated Data)
   b. Indexing Factors
   c. Base Address

C. State that the System/360 can do the following data operations:

1. Register to register with fixed length operands.
2. Storage to register with fixed length operands.
3. Storage to storage with variable length operands.

D. With reference to the I/O Channels:

1. Describe the difference between a Selector and a Multiplexor Channel.
2. Describe the difference between burst and multiplex modes.

Organization

The System/360 is a general purpose computer system. By this we mean it is designated to be used for commercial, scientific, and communications applications. In the past, these applications were handled by separate computer families.

Growth ↑	Scientific	Commercial
	7090	7080
	709	705 III
	704	705 II
	701	702

One scientific computer family and its comparable commercial equivalent.

The scientific computers were usually fixed word length machines and used a pure binary form of coding. On the other hand, the commercial computers were usually variable word length (character oriented) machines and used a binary coded representation of decimal information. The System/360 uses binary as well as BCD and has both fixed and variable length fields.

To fit the cost and volume needs of computer users, the IBM System/360 is available in several models. For instance, to suit the demands of users who need a minimum number of answers per month, a model 30 is available at a minimum cost. A model 70, however, will give approximately 50 times as many answers per month. Both models (30 and 70) are, however, program compatible. That is, a program written for a model 30 can run on a model 70 and vice versa. The answers will be the same; the numbers of answers per month will be different.

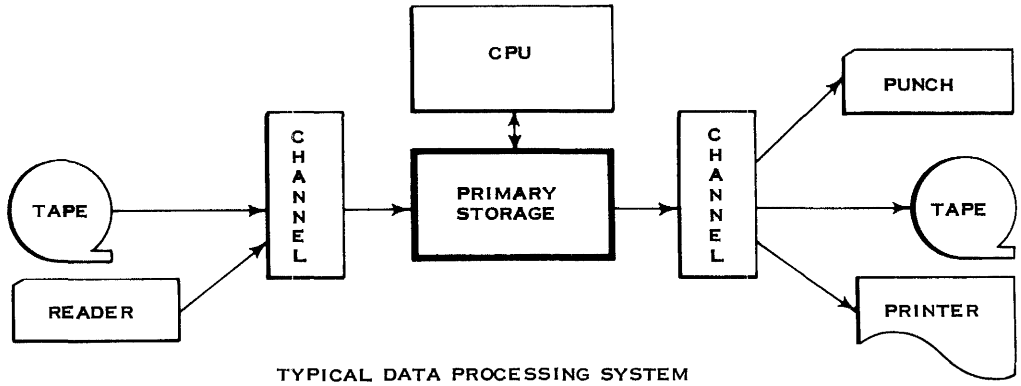
A machine language program written for one model of the System/360 can run on any other model. __________ (True/False)

True

The System/360 also uses a new technology known as Solid Logic Technology. This new technology is commonly referred to as SLT. Basically, it consists of printed circuitry instead of physical wiring on the back panel. It also uses packaged logic circuits. This new technology reduces manufacturing costs, increases reliability and reduces maintenance time. The details of SLT will not be covered in this course.

SLT stands for _______ ________ _____________.

Solid Logic Technology



TYPICAL DATA PROCESSING SYSTEM

In the preceding figure, you can see the components that make up a data processing system. You should be familiar with these components either from past experience or because of a basic computer systems principles course.

Let's learn about these components as they apply to the System/360!

The primary storage is that section of a DP system that contains the program to be executed as well as the data to be processed. All data entering the system goes into the primary storage before it can be processed. After processing, the data must be placed back into primary storage before it can be sent to an output device.

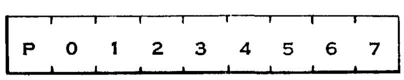
Primary storage is sometimes referred to as main storage. Most computers use ferrite cores as their primary storage device. The System/360 also uses ferrite cores for its main storage.

The type of storage used for primary storage in the System/360 is _______ storage.

---

core

The smallest addressable unit of main storage in the System/360 is called the byte. The byte consists of eight data bits and one parity bit.



THE BYTE

As can be seen above, the leftmost bit of a byte is the parity bit. System/360 uses odd parity. That is, an odd number of bits in every byte will be set (in the 1 state). The remaining bits will be reset (in the 0 state). If an even number of bits are set, a machine check (error) will be indicated.

The smallest addressable unit of main storage is called a ______ . It consists of _______ data bits and one ________ bit. The leftmost bit is the ________ bit. A machine error will be indicated if a byte has an _______ number of bits set.

---

byte
eight
parity
parity
even

As would be expected, the faster models of System/360 would need more storage bytes than the slower models. Also each model of the system would have as an option several sizes of main or core storage. As can be seen from the following figure, the model 30 comes in four sizes from approximately 8K bytes to 65K bytes. The model 70, on the other hand, can have either 262K or 524K bytes in main storage.

MAIN STORAGE
(USUALLY LOCATED IN THE PROCESSING UNIT)

CAPACITY (BYTES)	SYSTEM MODEL					
8,192	C30					
16,384	D30	D40				
32,768	E30	E40				
65,536	F30	F40	F50			
131,072		G40	G50	G60		
262,144		H40	H50	H60	H62	H70
524,288				I60	I62	I70
BYTES / ACCESS	1	2	4	8	8	8 ← (3)
MAIN STORAGE SPEED IN USECS	1.5	2.5	2	2	1	1 ← (1)

MAIN STORAGE IN MODELS 60 – 70 IS HOUSED EXTERNALLY TO THE PROCESSING UNIT. ← (2)

PROCESSING UNIT

Besides the byte size of each model of the System/360, there is other information available from the preceding figure.

1. The time required to take a storage cycle varies between models of the System/360.

2. In all but the models 60-70, the main storage is housed in the same physical structure as the processing unit.

3. The number of bytes accessed during each storage cycle varies with each model of the System/360. A storage cycle is the period of time during which information is read out of main storage. The information that is read out is either regenerated or new information is placed back into main storage.

The smallest addressable unit of main storage is called a _______.

---

byte

Use the preceding figure to answer the following:

A model 40 of the System/360 can have as few as _________ bytes or as many as _________ bytes.

Every time a model 30 takes a storage cycle, one byte is accessed. Every time a model 50 takes a storage cycle, ______ bytes are accessed.

16,384
262,144
four

One thing you should understand now is that the byte is the smallest addressable unit of main storage. This means that, regardless of which model we are discussing, each and every byte of main storage is individually addressable. To read out the first eight bytes of main storage, the model 30 would take eight storage cycles. For each cycle, the model 30 would change its storage address by 1, using addresses 0-7. The model 50 on the other hand, would need to take only two storage cycles. To access bytes 0-3, the storage address would be 0000. For the next four bytes (4-7) the address would be 0004 and not 0001. Actually, to access bytes 0-3 on a model 50, any of the 4 addresses (0000-0003) could be used. Later on you will learn that in certain cases, special restrictions are placed on the addresses used.

It is desired to read out the first ten bytes (0-9) of main storage on a model 40. How many storage cycles and what addresses would be used? Use the preceding figure for reference. ______________________________

---

5 storage cycles with addresses 0000, 0002 0004, 0006, 0008

You should now realize that main storage addresses start with 0000 for the first byte and increase by 1 for each byte in the particular main storage unit. Valid storage addresses for a model 30 would start with 0000 and continue up to 65,535. On a model 70, valid main storage addresses start with 0000 and continue up to 524,287. To allow for program compatibility as well as for future growth, the System/360 uses a 24-bit binary address to address main storage. A 24-bit binary number allows us to go as high as 16,777,215 for an address. You can see the future growth that is possible here! A binary rather than a binary coded decimal address is used because it is more efficient with large addresses.

Write the 24-bit binary address that would be used to address byte location 0007. ______________________________

000000000000000000000111

You should be familiar enough at this point with the binary numbering system to have done the preceding question without much difficulty. Of course, you might have a slight case of writer's cramps from writing out a 24-bit address. Normally, machine addresses are expressed hexadecimally. Hexadecimal is another numbering system you are familiar with. Binary uses a base of two (2^1) while hexadecimal uses a base of sixteen (2^4). There is a direct 4-to-1 ratio between binary and hexadecimal. Each four binary bits can be expressed as one hexadecimal digit. Address 0007 could be expressed as six hexadecimal digits:

000000000000000000000111 ——→ 000007

How would the highest 24 bit binary address be expressed hexadecimally?

____

FFFFFF

Each main storage address refers to an individual ______. Every main storage can be located by a ___-_____ binary address.

---

byte
24-bit

Every byte has ________ data bits and one ________ bit. The leftmost bit is the ______ bit and is used to give every byte an _________ number of bits set (in the 1 state).

---

eight, parity
parity, odd

How many bytes are read out during a storage cycle on a model 30? _____

One

Express the decimal value 12 as a hexadecimal digit. ____________

C

You are probably a little perplexed about this byte by now. You know that a byte consists of eight data bits and a parity bit! You know that each byte is individually addressable by a 24-bit binary address! You know that main storage size can vary from approximately 8K bytes on a model 30 to over 500K bytes on a model 70! You know that the model 30 accesses one byte per storage cycle while a model 70 accesses eight bytes per storage cycle! However, you are probably asking yourself:

Is the byte a character?
Is it a binary number?
Just what is it?

The answer to these questions is simple. The eight data bits of a byte can be coded to represent characters, binary numbers, or anything you want them to be. The instructions of the System/360 are many and varied. Some of the instructions treat bytes as characters. Some instructions treat bytes as part of a binary number. So the answer to the question, "What does a byte represent?" is that it depends on the particular instruction being executed at the time. This question will be answered more to your satisfaction after you study the data formats and some of the instructions.

As was previously stated, the System/360 is a general purpose data processing system. As such it is designed to operate with fixed length as well as variable length data. The byte as you have already learned is a very versatile unit. It is individually addressable. By further specifying the number of desired bytes, we can have a variable length field in main storage starting and ending at any byte address.

The System/360 can operate with variable length data. Variable length data can start at _____ byte address.

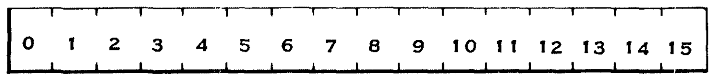
---

any

To be of truly general purpose, the System/360 most also be capable of operating with fixed length data. Whereas variable length data has a variable number of bytes, fixed length data always has a fixed number of bytes. The simplicity of this last statement almost scares you! So let's go on and define these fixed length fields.

A halfword is two bytes in length.

The data bit positions of a halfword are numbered 0-15 from left to right.



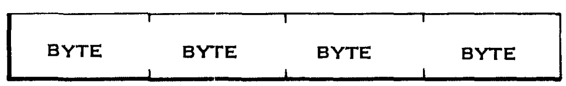
Notice that the parity bits are not shown. They will not be shown from here on, since they do not represent data. However, remember that every byte does contain a parity bit for checking purposes.

Two bytes are contained in a __________. Its data bit positions are numbered ___ to ____, left to right. Each halfword has _____ parity bits associated with it.

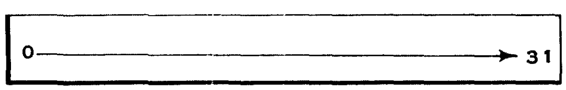
---

halfword
0 to 15
two

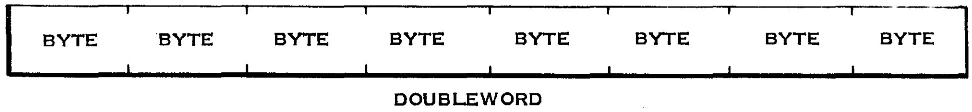
A word is 4 bytes long.



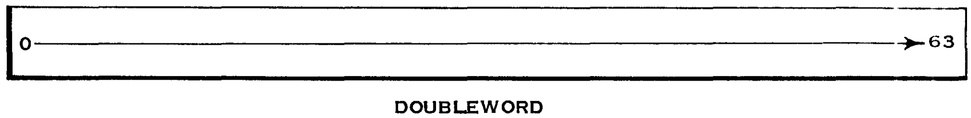
The data bit positions of a word are numbered 0-31 from left to right.



A doubleword is 8 bytes long.



The data bit positions of a doubleword are numbered 0-63 from left to right.



Remember now that each byte of a halfword, word, or doubleword carries its own parity bit.

A byte contains __________ data bits and one _______ bit.

---

eight, parity

A _______________ is two bytes in length.

A _______________ is four bytes in length.

A _______________ is eight bytes in length.

---

halfword
word
doubleword

The data bit positions of fixed length data are numbered from ______ ___________ (right to left/left to right) starting with bit position 0. Each ______ of fixed length data contains a parity bit.

---

left to right
byte

Remember now that it is the instruction being executed that determines whether to consider data as variable or fixed. The Op code of the instruction will also determine, in the case of fixed length data, whether it is a halfword, word, or doubleword.

Before leaving the definitions of fixed length data, you must learn the restrictions placed on the use of fixed length data.

The rule is that fixed length data must reside on the correct boundaries in main storage.

BYTE 0000	BYTE 0001	BYTE 0002	BYTE 0003	BYTE 0004	BYTE 0005	BYTE 0006	BYTE 0007	BYTE 000
HALFWORD		HALFWORD		HALFWORD		HALFWORD		HAL
WORD				WORD				
DOUBLEWORD								

Fixed length data is addressed by the high-order byte (leftmost byte) of the field.

For halfwords, this address must be divisible by two.

For words, this address must be divisible by four.

For doublewords, this address must be divisible by eight.

Another way of stating this rule is to say that the 24-bit binary address:

1. Of a halfword must have one low-order zero bit.
2. Of a word must have two low-order zero bits.
3. Of a doubleword must have three low-order zero bits.

A fixed length data field is addressed by its ________ (low/high) order byte.

---

high

The binary address of a word must contain __________ low-order zero bits.

The binary address of a doubleword must contain __________ low-order zero bits.

---

two three	The boundary restriction placed on the use of fixed length fields is a restriction placed on the user. If you violate these rules, it is not a machine check. Instead it is, and rightfully so, considered a program check. Starting a halfword data field at an odd address (such as 000001) will result in a ________ check. Incorrect parity in a halfword data field will result in a ________ check.
program machine	As there are other restrictions placed on the programmer, you should be able to identify program checks by type. The type of program check caused by a violation of fixed length boundaries is known as a specification exception. Another exception to valid programming is addressing a byte location that is not available on your particular model of System/360. The largest size main storage available on the model 30 is 65,536 bytes. Any address higher than this would result in a program check. This type of check is known as an addressing exception. What two types of program check exceptions could occur when addressing main storage? ________ ________
Specification Addressing	If the binary address of a word does not contain two low-order zero bits, the program check that occurs is identified as a ________ exception.
specification	

REVIEW QUESTIONS ON MAIN STORAGE

- Try to answer the questions without referring to the material. However, if you do require aid, refer to this book and/or the System/360 Principles of Operation manual and consider reviewing the area where aid is required.

1. The byte consists of __________ data bits and one ________ bit.

2. If a byte has an _______ number of its bits set, a machine check will occur.

3. Each main storage address refers to a unique ______ location.

4. Data field bit positions are numbered starting with 0 from ______ to ___________.

5. Data fields are addressed by their ________-order byte location.

6. A _______________ is two bytes long.

7. A _______________ is four bytes long.

8. A _______________ is eight bytes long.

9. What two program check exceptions could occur when addressing main storage? __________________ ______________________

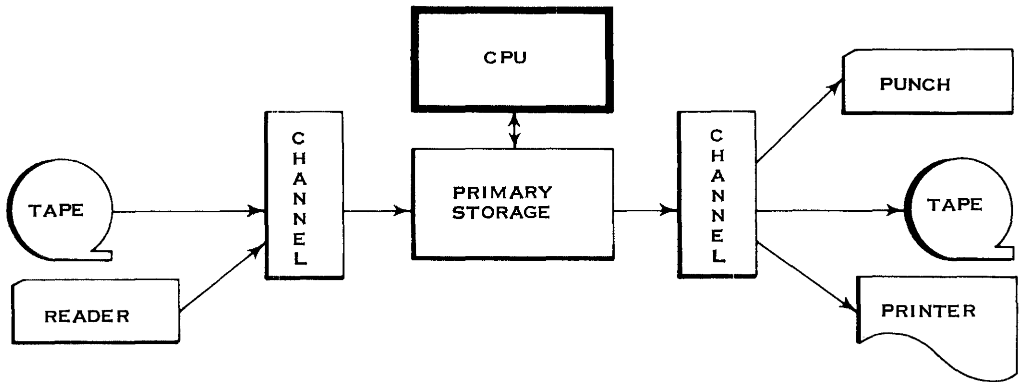
10. The address of fixed length data fields must be divisible by the number of _________ in the field or a ___________________ exception will occur.

ANSWERS TO MAIN STORAGE REVIEW QUESTIONS

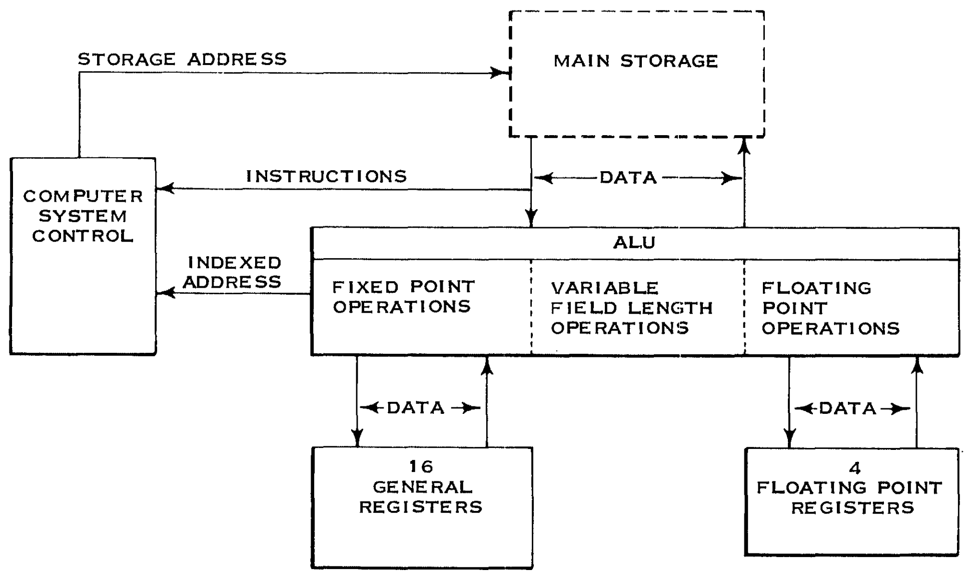
1. eight, parity
2. even
3. byte
4. left, right
5. high
6. halfword
7. word
8. doubleword
9. Specification, Addressing
10. bytes, specification

CENTRAL PROCESSING UNIT

Now that you know the primary storage capabilities of the System/360, let's explore those of the Central Processing Unit (CPU).



TYPICAL DATA PROCESSING UNIT



CENTRAL PROCESSING UNIT LOGIC FLOW

In the preceding frame, you can see the logical structure of the CPU for the System/360 and its relationship to the main storage.

As you know, there are two main sections in CPU. They are: 1) the control section, and 2) the arithmetic and logical section (called ALU).

From the illustration on the facing page, you should be able to see some of the functions of the control section. They are:

1. All references to main storage, whether for instructions or for data, are made by the control section.
2. During I time of any instruction, the control section addresses main storage and causes the instruction to be fetched and sent to the control section. The instruction is then decoded by the control section and executed during E time.

During I time of an instruction, the instruction is brought out of main storage to the ________ section. The control section decodes the ____________.

All addresses are supplied to the main storage by the ________ section.

During E time of an instruction, the instruction is __________.

---

control
instruction
control
executed

In general, the arithmetic and logical section of a computer contains the circuits necessary for adding and comparing data fields as well as the other circuits necessary for operating on data fields.

As can be seen from the CPU Logic Flow illustration, the ALU can do:

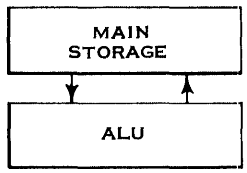
1. Variable field length operations.
2. Fixed point operations involving fixed length fields.
3. Floating point operations.

In your own words, what is the function of the arithmetic and logical unit of a computer? ____

____

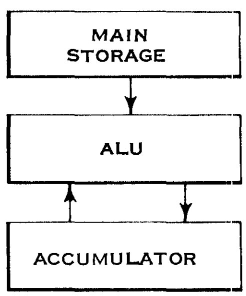
After the instruction has been fetched and decoded in the control section, the data fields are brought out to the ALU, and the operation (such as add or subtract) is executed.

In looking at the ALU, let us first consider variable length fields as used in many commercial computers of the past. Two main concepts were used. The storage-to-storage concept was used by computers of the 1401 family. In it the data fields were brought out of main storage, operated upon, and the results went back into main storage.



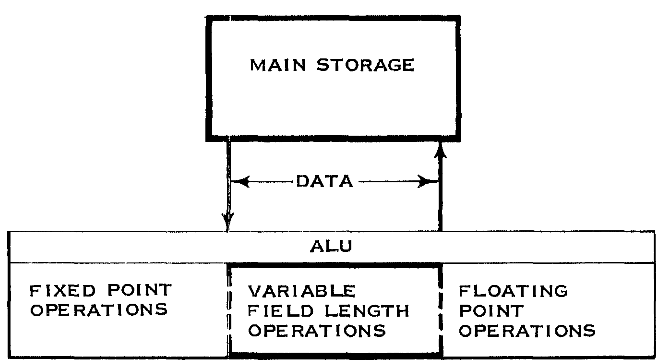
STORAGE–TO–STORAGE CONCEPT

Other computers such as those of the 702 - 705 family used a storage-to-accumulator concept. The accumulator was a small storage device. The storage medium could be core storage, vacuum tube or transistorized registers. In the storage-to-accumulator concept, one of the data fields would be in main storage and the other would be in an accumulator. Both fields would be brought out to the ALU, operated upon, and the result would go back into the accumulator.



STORAGE–TO–ACCUMULATOR CONCEPT

For its variable length operations, the System/360 uses the storage-to-storage concept.

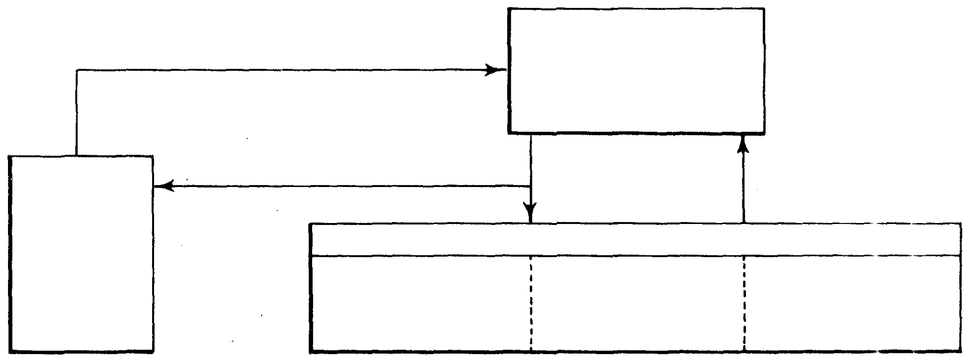


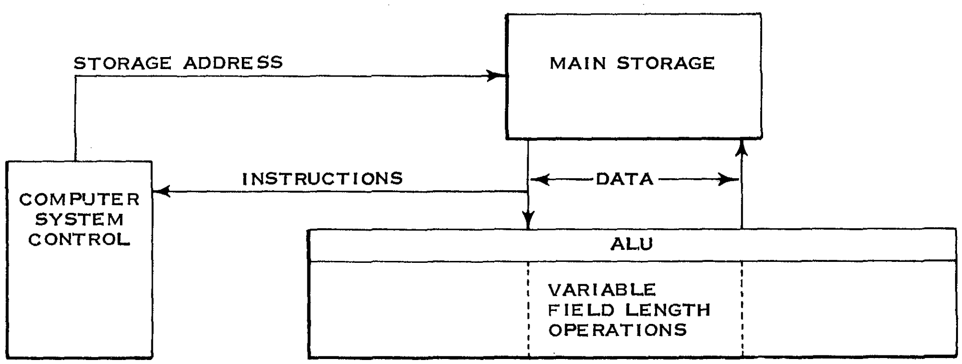
Sections of the System/360 necessary for a variable length operation, including I time, are shown in this frame. Label the blocks as to:

Control Section
Main Storage
ALU
Variable Field Length Operations

On the lines connecting the blocks, indicate whether they are:

Addresses
Instructions
Data





Fields of data (fixed or variable length) are often referred to as operands.

Instructions usually contain an Op code, the address of a first _________ and the address of a second _________.

---

operand
operand

As you have previously learned, variable length fields can start at any byte location in main storage. They are not restricted by storage boundaries as are fixed length operands. However, there must be some way of indicating to the system the length of the fields. In computers of the past, this was done several ways. The 1401 used a special word mark bit over the high order. The 705-II used zone bits. In the System/360, variable length operations use binary and decimal operands. In order to be code independent, System/360 specifies the length of these fields by a length code in the instruction.

Variable length fields can start at _____ byte location in main storage. Their length is specified by (in your own words) ________________________________.

---

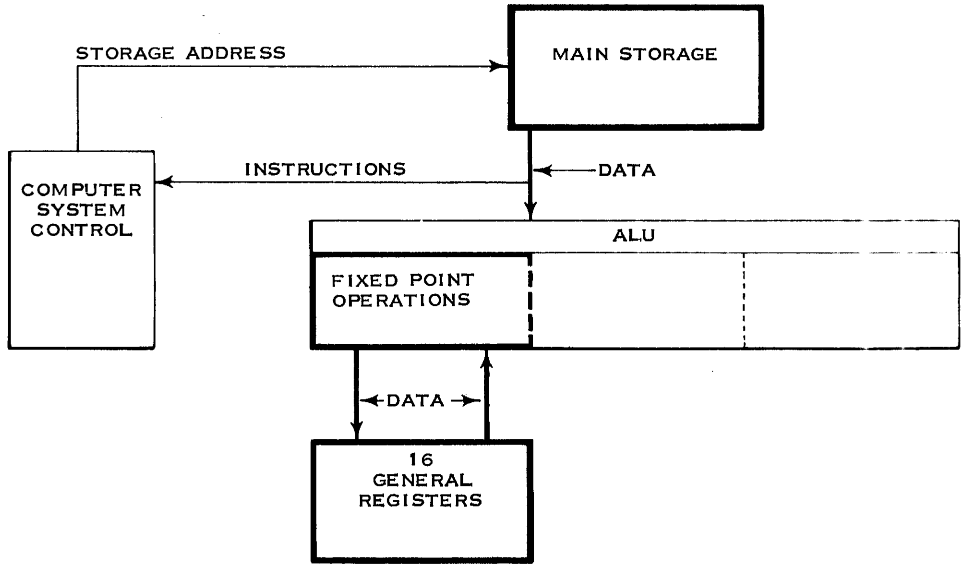
any
a length code in the instruction.

The length code can be either 4 or 8 bits long, depending on the instruction. The length code is in binary. As a result, the maximum length can be either 16 or 256 bytes. The value of the code is one less than the total number of bytes.

Length code of 0000 = 1 Byte
Length code of 1111 = 16 Bytes
Length code of 11111111 = 256 Bytes

A length code of 0111 would specify a variable field length of how many bytes? ____________

Eight (8)



When operating on fixed length fields (such as halfwords, words, or doublewords), the System/360 uses the storage-to-accumulator concept. These fixed length operations use binary operands. For use as accumulators, the System/360 has ____ registers available to the programmer. As these registers can be used for purposes other than accumulating, they are called ________ __________.

16
general registers

When working with fixed length operations, the System/360 uses a s________ -to- a__________ concept.

---

storage
accumulator

For use as accumulators, the programmer has available 16 ________ __________. These registers are numbered 0-15 and are addressed in an instruction by a 4 bit binary address field.

---

general registers

To use general register 0 as an accumulator, what address is given?

______

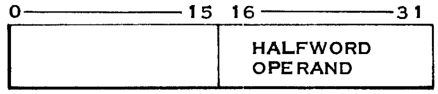
0000

General registers 0-15 are all one word in length. How many bytes may be contained in a general register? ______

---

four

Being a word in length, a general register can be used to contain a halfword data field. Data fields are sometimes referred to as operands.



GENERAL REGISTER

As can be seen in the preceding figure, the bits of a general register are numbered left to right starting with the number 0. Also, we can see that a halfword operand is placed in the low-order bits (16-31) of a general register.

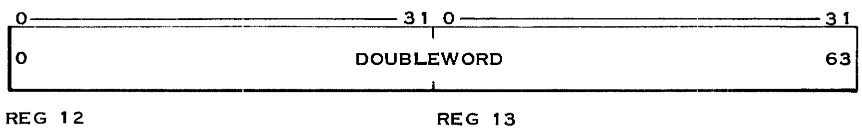
None of the general registers 0-15 can contain a doubleword. For those operations that use a doubleword operand, such as fixed length divide, a pair of adjacent registers is used. In these cases, an even-odd pair of registers (such as 0-1 or 6-7) is used, and the even register is addressed.

With general register address 1100 specified, which two general registers would be used in a fixed length divide operation? ____________

---

12 and 13

In the preceding question, bits 0-63 of the doubleword would be in the registers as shown below.

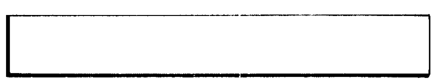


Fixed length operands in main storage must be on integral boundaries or a program c______ will occur indicating a s____________ e__________.

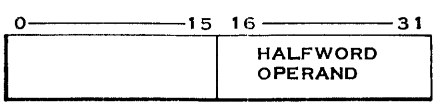
---

check
specification exception

Number the bit positions of the general register below. Also show where a halfword operand would be placed.

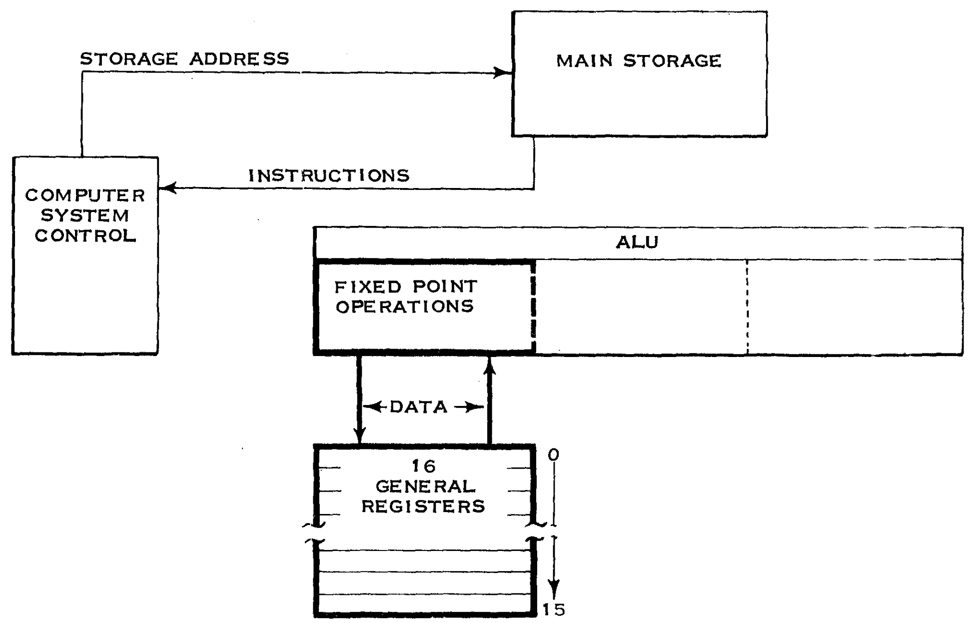


GENERAL REGISTER



GENERAL REGISTER

With sixteen general registers, sometimes both fixed length binary operands will be in the general registers. In these cases, another data flow concept is used. The System/360 can do a register-to-register (accumulator-to-accumulator) operation.

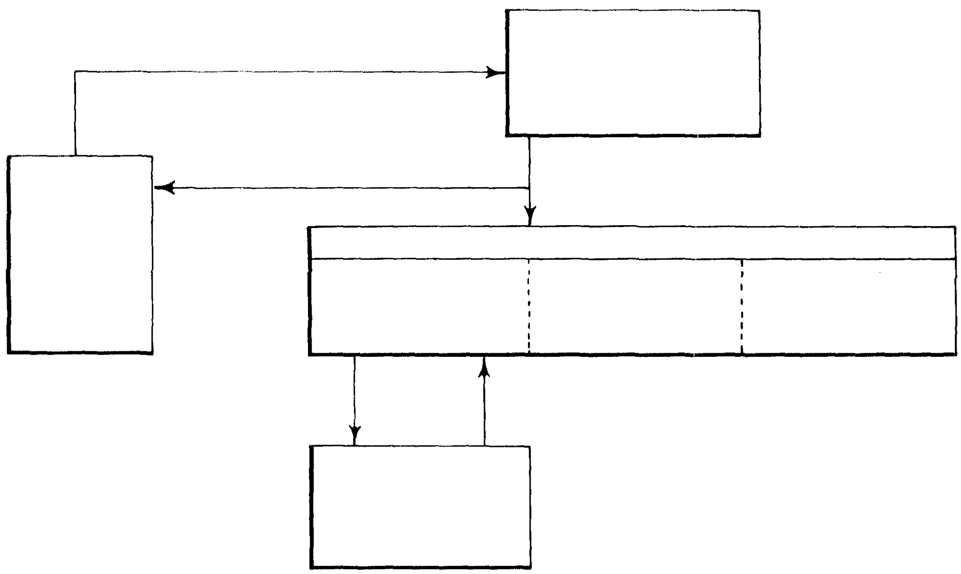


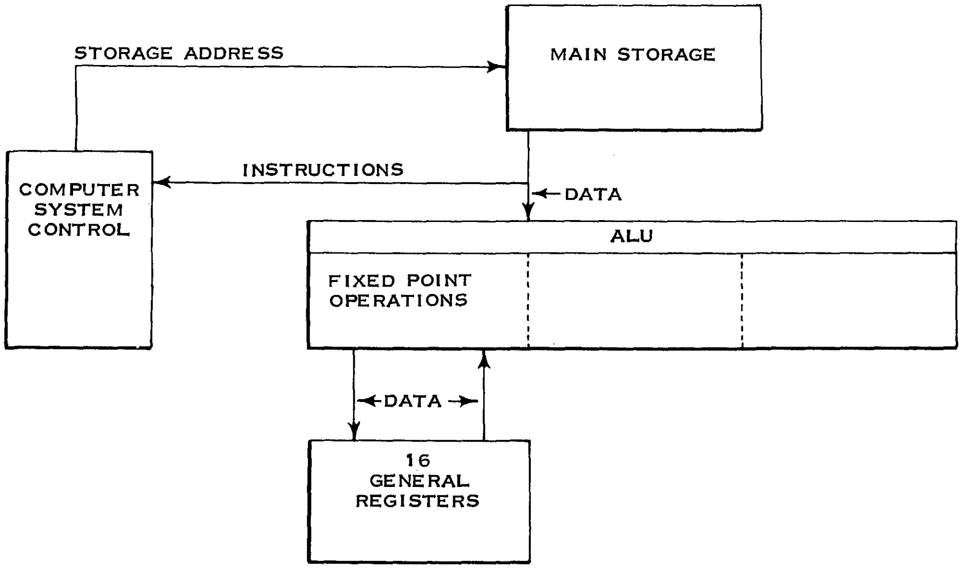
Sections of the System/360 necessary for fixed length operations, including I time are shown in this frame. Label the blocks as to:

Control Section
ALU
Main Storage
Fixed Length Operations
General Registers

On the lines connecting the blocks indicate whether they are:

Addresses
Instructions
Data





The general registers are also used for purposes other than accumulating. Two other main uses are as Index Registers and Base Registers. Indexing is a form of indirect addressing. An increment contained in an index register is added to the data address in the instruction to form an effective main storage address. Neither the index register nor the instruction in storage is changed by indexing. The use of the general registers as index and base registers will be explained later in this course. Base registers are similar to index registers.

List three main uses of the general registers. 1. ____________
2. ____________ 3. ____________

---

1. Accumulators
2. Index Registers
3. Base Registers

The floating point feature is not an objective of this course. Some information however, is necessary to acquaint you with the term "floating point." Floating point is the term given to arithmetic operations involving a fraction and an exponent. For instance:

217,000 can be expressed as $.217 \times 10^6$

296,000 can be expressed as $.296 \times 10^6$

Fixed point arithmetic would add the numbers as follows:

$$\begin{array}{r} 217{,}000 \\ +\ \underline{296{,}000} \\ 513{,}000 \end{array}$$

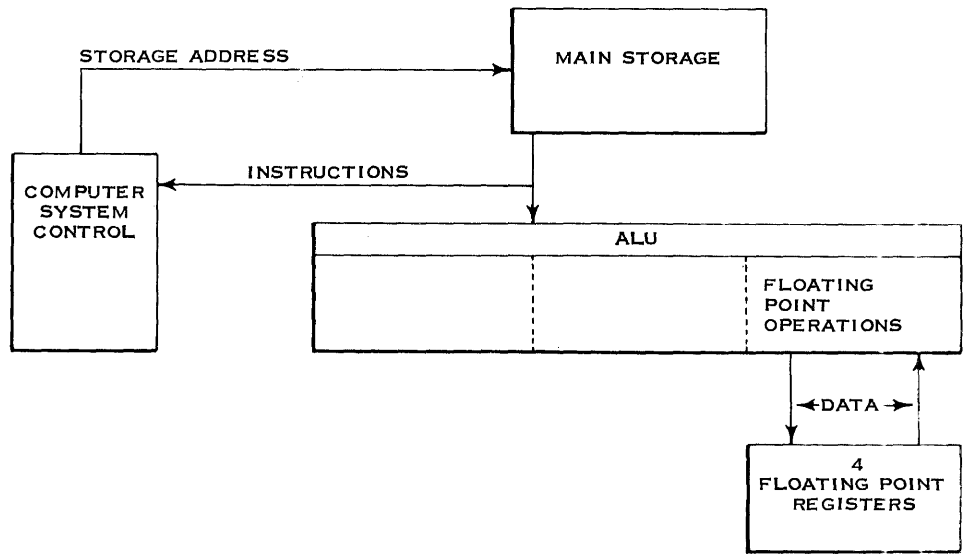
Floating point arithmetic would do it like this:

$$\begin{array}{r} .217 \times 10^6 \\ +\ \underline{.296 \times 10^6} \\ .513 \times 10^6 \end{array}$$

Add fraction → (fraction) Retain exponent → (exponent)

The example shown is an example of decimal floating point. The System/360 uses hexadecimal floating point. For instance:

$$\begin{array}{r} .7F \times 16^6 \\ +\ \underline{.1F \times 16^6} \\ .9E \times 16^6 \end{array}$$



Floating point arithmetic is most useful for expressing very large numbers and operating on them with much precision. To do floating point arithmetic, the System/360 has ______ floating point registers.

---

four

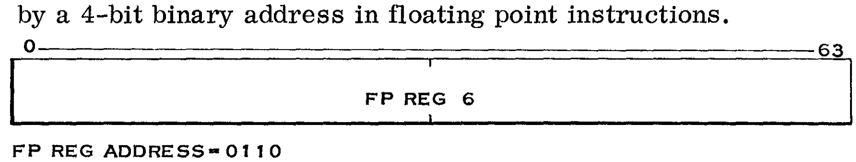
The four floating point registers are numbered 0, 2, 4, 6. These are not the same as general registers 0, 2, 4, 6. The floating point registers are separate registers used only as accumulators during floating point operations.

There are _______ floating point registers numbered _______, ______, _______, _________. The floating point registers are not the same as (in your own words) ____.

---

four
0
2
4
6
general registers
0, 2, 4, 6

The floating point registers are doubleword registers and are addressed by a 4-bit binary address in floating point instructions.



Floating point registers are _____ bits long and can contain a __________________.

64
doubleword

REVIEW QUESTIONS ON CENTRAL PROCESSING UNIT

- Try to answer the questions without referring to the material. However, if you do require aid, refer to this book and/or the System/360 Principles of Operation manual and consider reviewing the area where aid is required.

1. Instructions are decoded by the ________ ________ of CPU.

2. For its variable field length operations, the System/360 uses the ________ to ________ concept.

3. Variable length fields can start at _____ byte location in main storage.

4. The length of variable length fields is specified by (in your own words) ____.

5. For fixed length operations, the System/360 uses a ________ to ____________ concept.

6. For use as accumulators, the programmer can address ________ ________ __________.

7. Number the bit positions of the general register below. Also show where a halfword operand would be placed.

8. With general register address 1100 specified, what two general registers would be used in a fixed length divide operation? ________

9. List three main uses of the general registers.

    1. ________________

    2. ________________

    3. ________________

10. When both fixed length operands are in general registers, a ________ to ___________ concept may be used.

11. For floating point operations, the System/360 has ______ floating point registers.

12. System/360 uses the ____________ numbering system for its floating point expressions.

13. The floating point registers are _____ bits long.

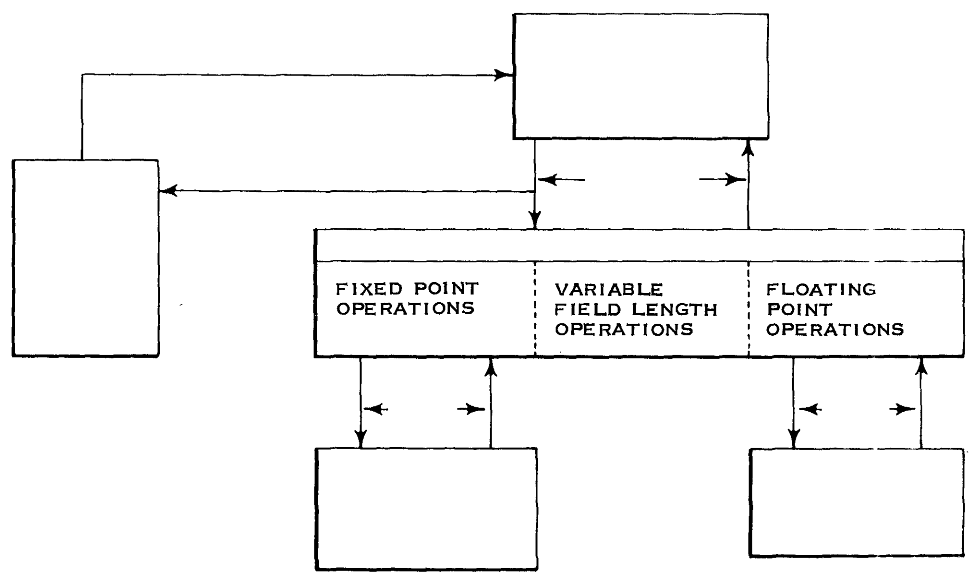
14. Shown below are the blocks that make up the System/360 CPU as well as main storage.

Identify the blocks as to:

Main Storage
Control Section
General Registers
ALU
Floating Point registers

Identify the lines connecting the blocks as to:

Addresses
Instructions
Data



ANSWERS TO CENTRAL PROCESSING UNIT REVIEW QUESTIONS

1. control section

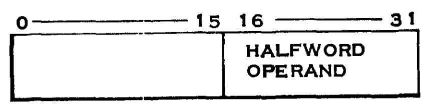
2. storage-to-storage

3. any

4. length code in the instruction

5. storage-to-accumulator (or register)

6. sixteen general registers

7. 

8. General Registers 12 and 13

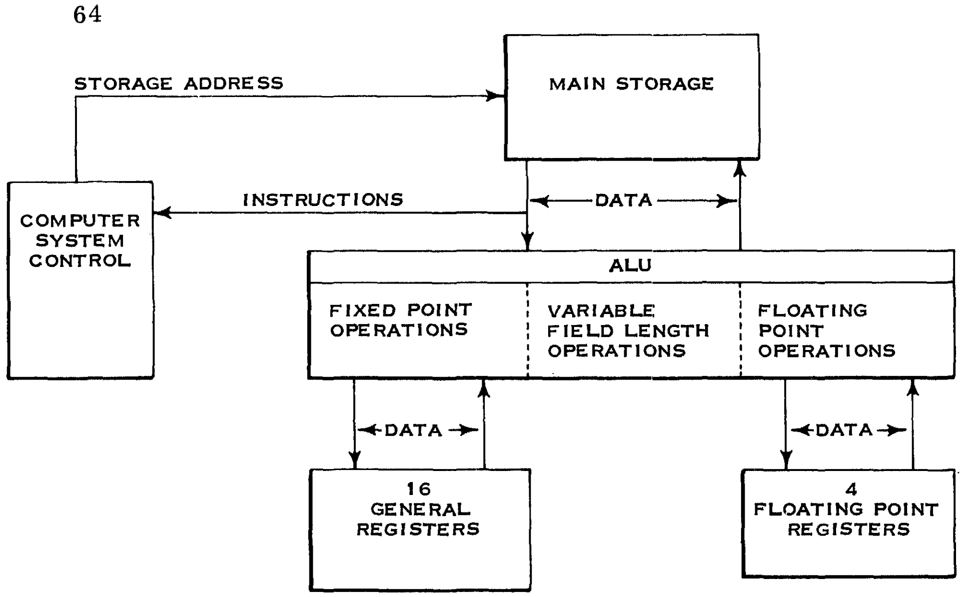
9. 1. Accumulators
   2. Index Registers
   3. Base Registers

10. register-to-register or accumulator-to-accumulator

11. four

12. hexadecimal

13. 64

14. 

The organization of the System/360 which you have been learning is its logical structure. By this we mean that this is the way the System/360 appears to the programmer. The manner in which this logical organization is implemented will vary between the different models of the System/360.

For example this is how the registers are implemented:

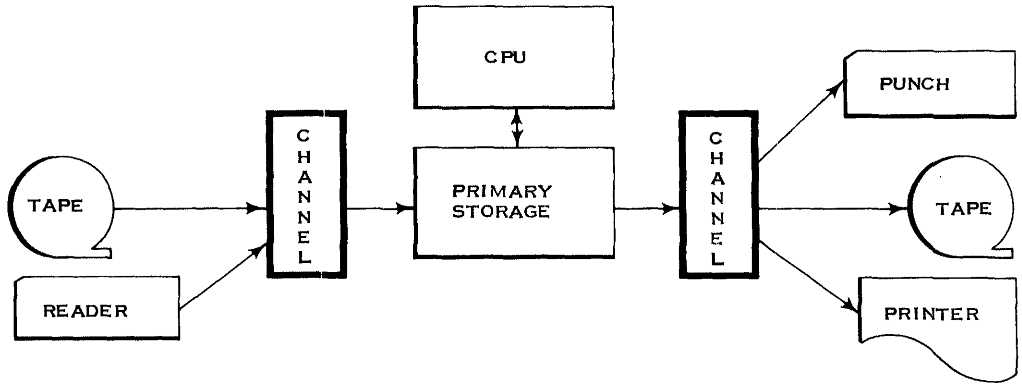
1. In models 60, 62, 70 of the System/360, the general and floating point registers are conventional transistor registers as used in past computers.

2. In models 40 and 50 a core array is used for the general and floating point registers. This array is similar to main storage but is a separate physical entity. It is called Local Store.

3. In model 30 the general and floating point registers are located in the main storage unit. However, they do not use any of the available main storage addresses. The area of the main storage unit used for registers is called Bump Storage.

Another example of hardware differences is in the control section of the System/360. In the model 70 the control section is made up of high-speed conventional transistorized circuits. However, other models of System/360 use a capacitor or a transformer storage device for most of their control functions. This device is called Read Only Storage (ROS). The ROS is a storage device but cannot be changed by the programmer. It is strictly a hardware control device.

In this section of the course your objective is to learn the logical organization of the System/360 and to be able to program it. Let's go on and learn the logical organization of System/360 channels!



From your knowledge of basic computer systems principles, you should realize the importance of input-output channels in any computer system. Their main function is to act as an intermediary between the I/O devices and the main storage unit.

Before input data can be processed in the ALU, it must first reside in ____________ ____________.

---

main storage

Before processed data can be sent to an output device, it must be placed in ____________ ____________.

---

main storage

All data flow between I/O devices and the main storage passes through the ________________.

---

channels

One of the main functions of a channel is to handle I/O requests for a main storage cycle. The channel receives data from the System/360 I/O devices one byte at a time. When enough data has been received to justify the use of main storage, the channel will request a storage cycle. The amount of data required will vary from one to eight bytes depending on the particular model of System/360. After the data has been placed in main storage the channel will wait for additional information from the input device. For an output device the procedure reverses. The channel requests a main storage cycle and brings out data. It passes this data to the output device one byte at a time. The requesting of a main storage cycle by the channel for I/O data is commonly referred to as a "Break-In."

The channel receives ________ ________ of data at a time from an I/O device. Requesting of a main storage cycle by the channel is known as a "________-________".

---

one byte
"break-in"

Since the channel is taking care of main storage cycles for the I/O device, the central processing unit now is logically free to continue processing instructions. We say that processing is "overlapped" with the I/O operation.

This simultaneous operation of an I/O device and the processing of instructions is known as __________________.

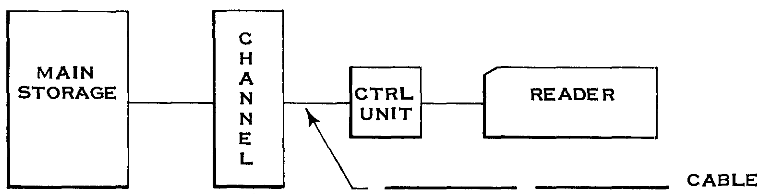
---

overlap

On some models of the System/360, overlapping the channel with CPU operations is not allowed at certain times. Once the CPU has started a channel operation, it has to wait for the channel operation to finish before it can continue processing instructions.

All data and control information are communicated between the System/360 channel and its I/O devices via a Standard Interface cable. More on this later!

Each I/O device logically ties into the System/360 channel's Standard Interface through a control unit.



Fill in the blanks in the illustration.

---

standard interface

The I/O device logically ties into the channel's Standard Interface through a __________ ______________.

---

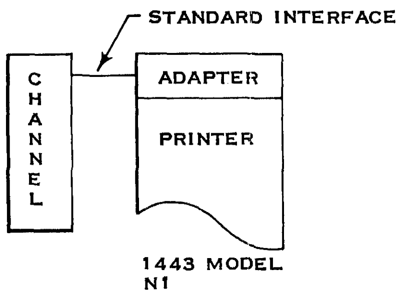
control unit

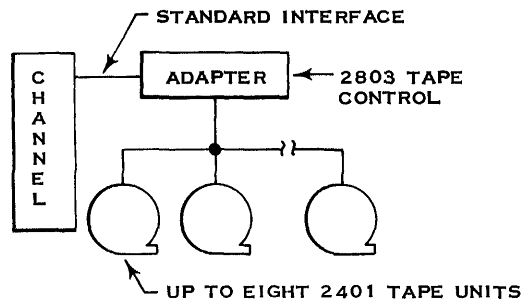
Another name for a control unit is adapter. For some I/O devices, the control unit or adapter is built into the device. For other devices, the control unit is external to the device.

The control unit, or ____________, may be housed in the ______ ___ or may be external to it.

---

adapter
device





Some adapters can control only one I/O device while others can control a number of similar I/O devices. The 1443 Printer Model N1 is an example of an I/O device with a self-contained adapter which controls only one printer. The 2803 tape control is an example of a stand-alone adapter which can control up to eight 2401 magnetic tape units.

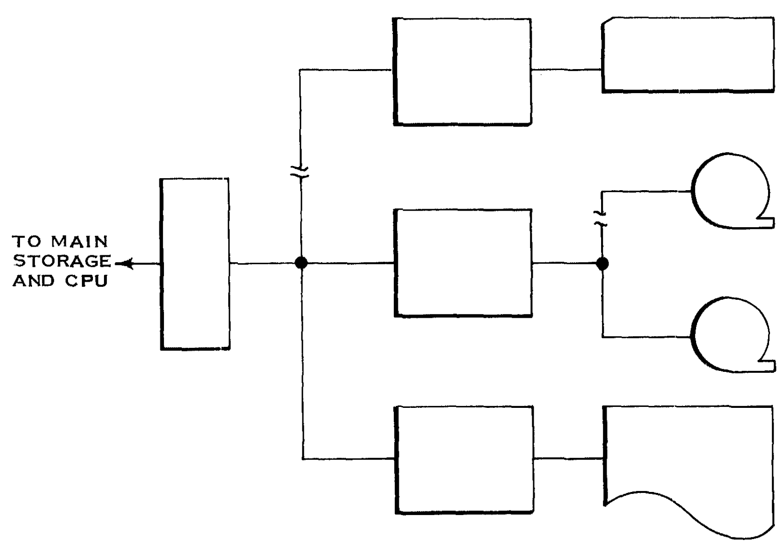
Each channel of the System/360 has the ability to select up to 256 I/O devices. There are physical limitations, of course. One of these is in the standard interface between the channel and the I/O device. There can only be up to <u>eight</u> control units tied into the standard interface cable.

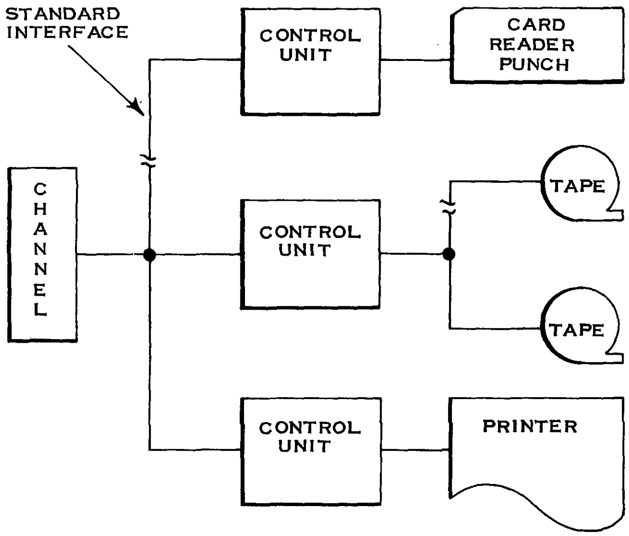
The communication lines between the channel and its I/O devices are known as the __________ ___________. The maximum number of control units that can tie into the standard interface cable is ______.

---

standard interface
eight

Label each block of the channel organization shown below as to channel, control unit, or type of device. Indicate which line is the standard interface.





Channels are logical concepts in I/O operations. In the System/360, they may be stand-alone units as in the model 70 or may be packaged along with main storage in the CPU housing as in the model 30. In the lower models of the System/360, many of the processing units' circuits are used by the channels for their functions. There are two types of channels used by System/360: 1) selector channels, and 2) multiplexor channels. Let's discuss the selector channels first!

SELECTOR CHANNELS

Selector channels are available on all models of the System/360. The maximum number per model varies from two for a model 30 to six for a model 70. The selector channel is so named because only one I/O device can be selected on the channel at any one time. Once selected, a complete record is transferred over the standard interface one byte at a time.

On a selector channel, only one I/O device can be __________ at a time. Once selected, a __________ ________ is transferred over the _________ __________ one byte at a time.

---

selected
complete record
standard interface

Once the record has been transferred, the channel is free to select another I/O device. When a channel is transferring an entire record between main storage and an I/O device, it is said to be operating in "Burst Mode."
Since a selector channel always transfers an entire record, it can only operate in burst mode.

The operation of a channel with only one I/O device for the entire record is known as _______ ______ . Burst mode is the only mode in which a __________ channel can operate.

---

burst mode
selector

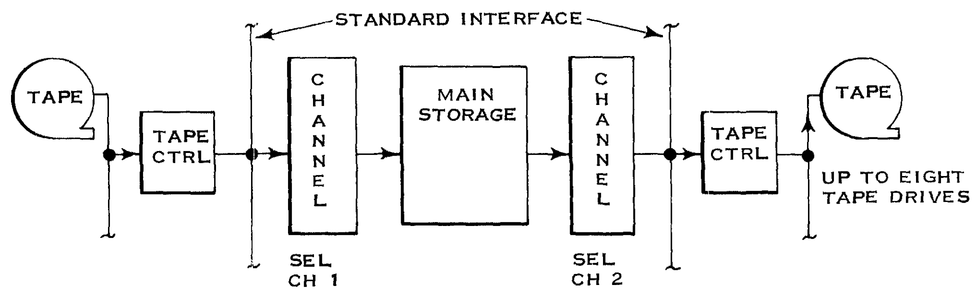
In summary then, on a selector channel, one I/O device transfers an entire record over the channel's standard interface. During this time no other I/O device can be using the channel. The other I/O devices could, however, be in the process of a feed cycle involving no data transfers over the channel. This is most apparent in those I/O devices which have buffers in their control units.

On a selector channel, one I/O device transfers an _______ ________ over the channel. This mode of channel operation is known as ______ ________. During this period of time no other I/O device can be using the _______________.

---

entire record
burst mode
channel

Although only one I/O device can be operating on a selector channel at any one time, multiple selector channels can be operating simultaneously. The following illustration shows an input record being read in from tape over selector channel 1 at the same time as an output record is being transferred over selector channel 2. All channels have their own individual standard interface cable.



Each channel has its own _______ ___________ _________.

All channels can be in operation (your own words)______________________.

---

standard interface cable
simultaneously

Selector channels are designed to operate with high data rates. I/O devices such as magnetic tape, disk units, drums, and buffered card devices are the devices most likely to operate on a selector channel. For operating with communication terminals in a real time application and with low data rate devices like an unbuffered card punch unit, a multiplexor channel is used. A multiplexor channel is available on models 30, 40, 50 of System/360.

I/O devices that operate at high data rates usually use ____________ channels which can operate only in ___________ mode.

For operation with low-speed or real-time I/O devices, a ____________ channel is available on System/360.

---

selector
burst
multiplexor

A Selector channel is designed to operate with only one I/O device at a time on an entire record basis. A Multiplexor channel is designed to operate with a number of I/O devices simultaneously on a byte basis. That is, several I/O devices can be transferring records over the multiplexor channel, time-sharing it on a byte basis.

A number of I/O devices can be operated simultaneously with a ________ channel. The I/O devices time-share the multiplexor channel on a _______ basis.

---

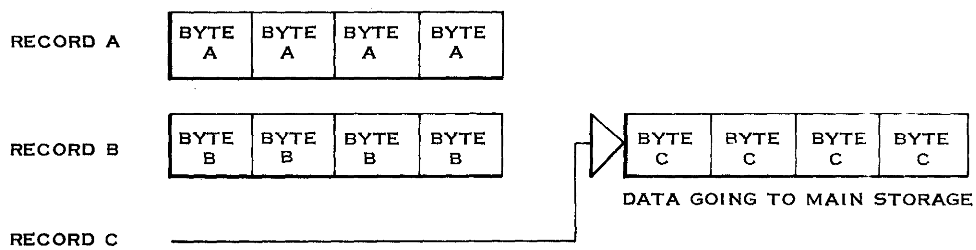
multiplexor
byte

This time-sharing mode of operation is known as "Multiplex" mode. Selector channels can only operate in ________ mode. Multiplexor channels can operate in __________ mode.

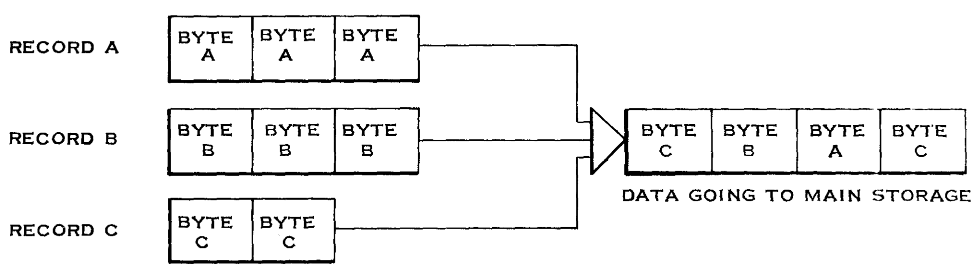
---

burst
multiplex

A comparison of burst versus multiplex mode can be seen below.



BURST MODE



MULTIPLEX MODE

To handle data flow from an I/O device, the channel needs to know certain information such as:

1. In which direction does data flow (input versus output)?

2. Where in main storage should data be placed or taken out of?

3. How many bytes should be sent to an output device or accepted from an input device?

Information of this type is contained in the I/O command addressed to a particular I/O device. For a selector channel, which operates with only one I/O device at a time, the information may be placed in the channel registers and left there to control the operation.

Information necessary to control a selector channel operation is contained in the channel ___________. This information was contained in the I/O ______________ addressed to a particular I/O device.

---

registers
command

On a multiplexor channel, it is possible to have up to 256 I/O devices operating simultaneously. The actual maximum number varies with the particular model and main storage of the System/360. In any case, it is not feasible to have all this information sitting in the multiplexor channel's registers. A set of registers would be necessary for each I/O device! Instead, the multiplexor channel keeps this information in a compact storage area. As a byte of data comes in from a particular I/O device, the multiplexor channel brings the necessary information out of the compact storage area and places it in its registers. After the byte of data from the I/O device has been serviced, the information in the registers is automatically put back into the compact storage area.

A number of I/O devices can operate on a ___________ channel simultaneously. The control information necessary for each I/O device is kept in a compact _________ _________.

---

multiplexor
storage area

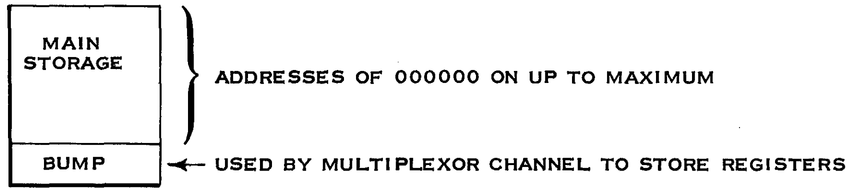
As a byte of data comes in from an I/O device, the control information is brought out and placed in the channel ___________.

After the byte of data has been serviced, the control information is placed back into the compact ________ __________.

---

registers
storage area

The compact storage area used by the multiplexor channel is known as Bump storage. Bump storage is part of the physical core array used for the main storage unit.



As can be seen above, the bump storage does not use any of the main storage addresses. It is a physical part of the core array used by the main storage unit. However, logically it is separate from it and has separate addressing lines. On the model 30, part of the bump storage available to the hardware is also used to contain the sixteen general registers.

The control information necessary for each I/O device on a multiplexor channel is contained in ______ ________. The control information in bump storage comes from the original I/O c________ addressed to a particular I/O device.

---

bump storage
command

Each I/O device has an area in ______ ________ to contain its control information from the original I/O ________.

---

bump storage
command

Bump storage does not use any of main storage (your own words) ______
____.

---

addresses or available area or equivalent

On the model 30, bump storage is also used to contain the ________
________ __________.

---

sixteen general
registers

Each I/O device has an area of bump storage for its own individual use when operating on a multiplexor channel. In effect then, a multiplexor channel is comprised of a number of subchannels. Each subchannel has its own area of bump storage. All subchannels (I/O devices) can be transferring records simultaneously. However, the multiplexor channel registers can be used with only one subchannel at a time. When the subchannel has finished servicing a byte of data for its I/O device, its control information is placed back into its area of bump storage. The multiplexor channel registers are now free to be used by another (or possibly the same) subchannel.

The multiplexor channel can be said to be a number of ____________. The multiplexor channel registers can contain at any one time the control information for only one ____________.

---

subchannels
subchannel

When not being used to service data bytes, the subchannel information is contained in ______ _________.

---

bump storage

Operating several I/O devices simultaneously on a multiplexor channel and servicing their data bytes as needed is known as __________ mode.

---

multiplex

__________ channels can operate with only one I/O device at a time. This mode of operation is called _______ _______.

Selector
burst mode

Multiplexor channels can also operate in burst mode if necessary. Burst mode can be forced on the multiplexor channel by the I/O device. This is done if high-speed devices are placed on the multiplexor channel.

Multiplexor channels can operate in two modes: __________ ______ and ________ ________.

---

multiplex mode
burst mode

The normal mode of operation for a multiplexor channel is ____________ ________. Burst mode can be forced on a multiplexor channel by the ______ _________.

---

multiplex mode
I/O device

Go to the IBM System/360 Principles of Operation manual and briefly study the following areas of the System Structure section:

- Main Storage
  - Information Formats
  - Addressing
  - Information Positioning
- Central Processing Unit
  - General Registers
  - Floating Point Registers
- Arithmetic and Logical Unit

REVIEW QUESTIONS ON CHANNELS

- Try to answer the questions without referring to the material. However, if you do require aid, refer to this book and/or the System/360 Principles of Operation manual and consider reviewing the area where aid is required.

1. All data flow between I/O devices and the main storage passes through the ________.

2. The channel receives _____ ______ of data at a time from an I/O device.

3. The requesting of a storage cycle by the channel is known as ________.

4. The simultaneous operation of an I/O device on the channel and the processing of instructions in the CPU is known as ________.

5. The I/O device communicates with its channel via a ________ ________ cable.

6. The I/O device logically ties into the standard interface through a ________ ______.

7. The operation of a channel with only one I/O device for the entire record is known as ______ ______.

8. Each channel has its own ________ ________ cable.

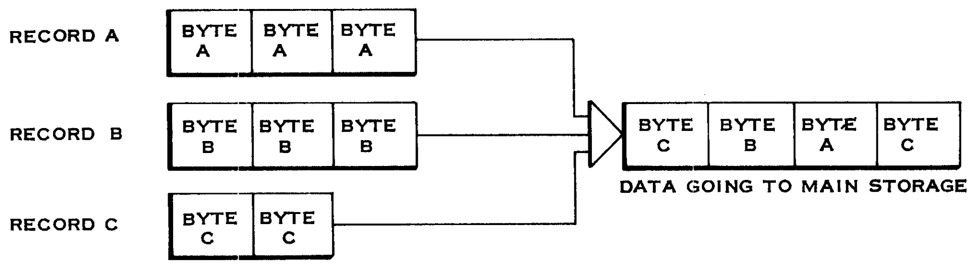
9. ________ channels are designed to operate at high data rates and can operate only in _______ mode.

10. A __________ channel is designed to operate with a number of I/O devices simultaneously.

11. The control information necessary for each I/O device in operation on a multiplexor channel is contained in ______ ________.

12. Multiplexor channels can operate in two modes: _______ _______ and ________ ______.

13. The following illustration shows a ____________ mode operation.



ANSWERS TO CHANNEL REVIEW QUESTIONS

1. channels
2. one byte
3. break-in
4. overlap
5. standard interface
6. control unit
7. burst mode
8. standard interface
9. Selector, burst
10. multiplexor
11. bump storage
12. burst mode, multiplex mode
13. multiplex

SECTION III LEARNING OBJECTIVES

At the end of this section, you should be able to:

A. Decimal Formats

1. Show the Extended BCD Interchange Code for alphameric characters.
2. Show a numeric field in the packed and unpacked format.
3. Add and subtract packed decimal operands.
4. State the rules for determining a decimal overflow.

B. Binary Formats

1. Show a binary number in both the halfword and word formats.
2. State that negative binary operands appear in "twos" complement form.
3. Add and subtract binary operands.
4. State the rules for determining a fixed point overflow.

Data Formats

It is assumed that you have had experience with the IBM card code or, as it is sometimes called, the Hollerith Card Code. From your previous computer experience or from a basic computer systems principles course, you are also familiar with the Standard BCD (Binary Coded Decimal) code.

The seven bits of the standard BCD code are ____________________.

C, B, A, 8, 4, 2, 1

Correct parity is established by using the ___ bit.

C

The zone bits are the ___ and ___ bits.

A
B

The standard BCD code equivalent of a twelve hole punch in an IBM card is the _____________.

B and A bit

Assuming odd parity, the character J would consist of the ________ bits in the standard BCD code.

C, B, 1

The basic unit of information in the System/360 is the byte. Just as each card column can be contained as a character in the Standard BCD code, it can also be contained as a character in the System/360 byte.

The byte in the System/360 can be used to contain a ___________.

---

character

The character code used in the System/360 is known as the Extended BCD Interchange code. Neglecting parity for now, the extended BCD code uses 8 bits to express a character, whereas the standard BCD code uses only 6 bits.

The character code used in the System/360 is known as the __________ BCD Interchange Code. This code uses ___ bits to express a character.

---

extended
8

The use of 8 bits may seem inefficient. However the extended code has some definite advantages not contained in the standard BCD code:

1. 256 different bit configurations are possible.

2. Both upper and lower case alphabetic information can be coded.

3. All possible 256 bit combinations can be punched into an IBM card. This allows pure binary information to be coded on an IBM card, with each column representing 8 bits of binary information.

There are _____ possible bit combinations in the extended BCD code.
All 256 possible bit combinations can be punched into an IBM ______.

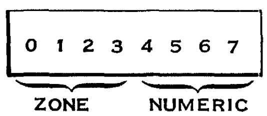
256
card

Basically, most character codes are divided into zone and numeric parts. The extended BCD interchange code is no exception. Let's take a look at the System/360 byte and see how it is divided.

The bit positions of a byte are numbered ___ through ___ from left to right.

0, 7

The EBCDIC (Extended BCD Interchange Code) divides the eight bits of a byte as shown below:

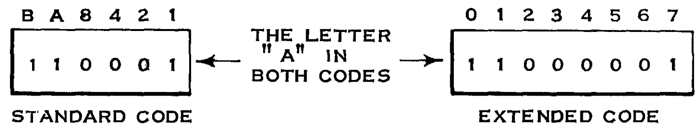


Bit positions 0 - 3 are used to express the zone portion of a character while bits 4 - 7 are used to express its numeric portion.

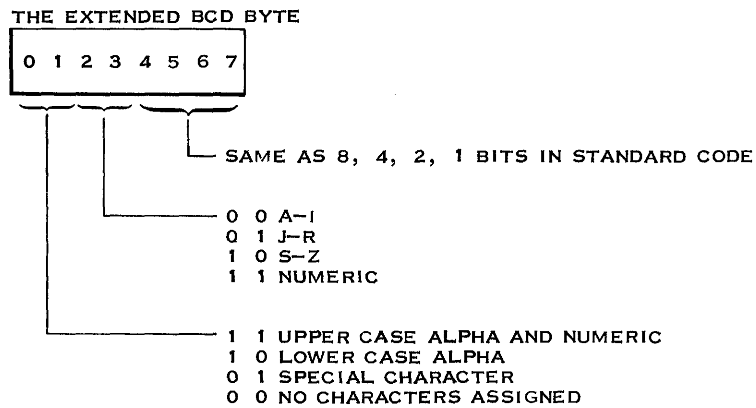
The numeric portion of a character uses bits ___ through ___ of a byte.
The zone portion of a character uses bits ___ through ___ of a byte.

4, 7
0, 3

Let's see how alphameric characters are expressed in the Extended BCD code as compared with the Standard BCD code.



Notice that bits 4 - 7 of the extended BCD code are used just like bits 8, 4, 2, and 1 of the standard code.



Go to the IBM Systems/360 Principles of Operation manual and study the EBCDIC chart in the Arithmetic and Logical Unit, Logical Operations area of the Systems Structure section. Use the chart as an aid while doing the following frames.

Bits 4-7 of the extended BCD code are used just as bits ___, ___, ___, and ___ of the standard code.

8, 4, 2, 1

Bits 2 and 3 of the extended code are used like the B and A bits of the standard code but in reverse order. In the standard code, the presence of B and A bits indicates the letters A-I and the absence of them indicates the numbers 0-9. In the extended code the absence of bits 2 and 3 indicates the letters A-I while the presence of them indicates the numbers 0-9.

Bits 2 and 3 of the extended code are the reverse counterpart of the _________ bits of the standard BCD code.

B and A

Bits 0 and 1 of the extended code are used to group the characters. Both bits indicate numeric as well as upper case letters. Bit 0 by itself indicates lower case letters while bit 1 by itself indicates special characters.

Examples:

Character	Bit Combination
A	11 00 0001
a	10 00 0001
1	11 11 0001
/	01 10 0001

What is the extended BCD code for the character "B"? ________

11000010

What is the extended BCD code for the character "b"? ________

10000010

What is the extended code for the character "2"? ________

DECIMAL DATA FORMATS

11110010

Decimal data consists of numeric fields which are coded to represent decimal numbers. For instance, the decimal number "17" can be represented in two columns of an IBM card by a 1-hole punch and a 7-hole punch. In the extended BCD code, this same number can be represented in two bytes like this:

1 1 1 1 0 0 0 1	1 1 1 1 0 1 1 1
1	7

Decimal data consists of ________ fields which are coded to represent (in your own words) ________________.

---

numeric
decimal numbers

In the preceding illustration, the number 17 did not have a sign and was considered plus.

How is the sign of a numeric field conventionally indicated in an IBM card?
____.

By a zone punch over the low-order or units position of the field.

A minus field is indicated by an _____ hole punch while a plus fi a _____ hole punch. The absence of zone punches in a card can a used to indicate a _______ field.

11
12
plus

A 12 and 1 punch in the low-order of a field would indicate a ______ sign and a low-order digit of ___.

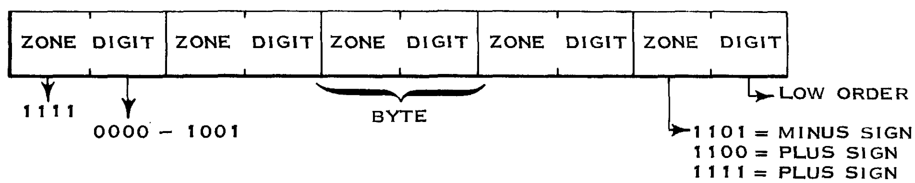
---

plus
1

When not dealing with decimal data fields a 12 and 1 punch can also be used to represent the character ___, and in the extended BCD code would have this bit configuration: ____________.

A
1100 0001

Decimal numeric fields in the extended BCD code are said to be in the zoned or unpacked format. The zoned or unpacked decimal format looks like this:



Decimal data in the extended BCD code are said to be in the __________ or zoned format.

---

unpacked

If the input to the System/360 is in card form, the sign of the unpacked format is indicated by a ______ punch over the low-order digit.

---

zone

It is a waste of storage space and processing speed to use the unpacked or zoned format for decimal arithmetic operation. The decimal feature of the System/360 uses a more efficient format for decimal arithmetic. It is called the packed or unzoned format. Since only four binary bits are needed to express a decimal digit, why not pack two digits into each byte of a decimal field? This is the packed format as used by the System/360.

The decimal feature of the System/360 uses the ________ format. The packed format has _____ decimal digits in a byte.

---

packed
two

What about the sign of a packed field? It is contained in the low-order bits of the low-order byte. A comparison of the unpacked and packed low-order byte is shown below.

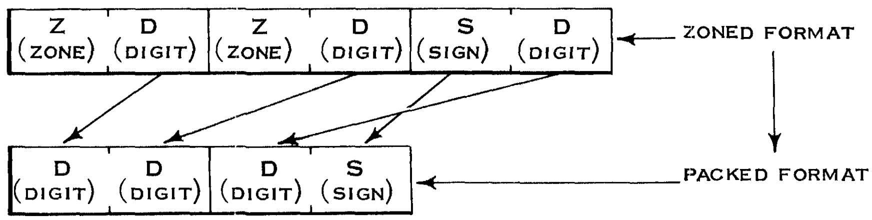
The packed format has two digits in each byte except for the low-order byte which has the ________________ in bits 0 to 3 and the _____ in bits 4 to 7.

In the unpacked format, the sign is in bits _____ of the low-order byte.

---

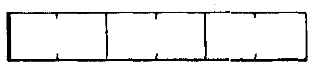
low-order digit
sign
0-3

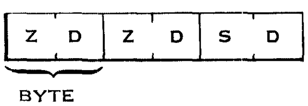
The next question is: If the System/360 will only process decimal data when it is in the packed format, how do you pack it? The System/360 has an instruction called "pack" which will take a decimal field in the zoned format and change it to the packed format as follows:



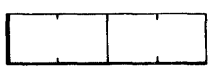
You are not expected to know this instruction at this time. You should be aware, however, that zoned decimal fields can be changed to the packed format by a machine instruction.

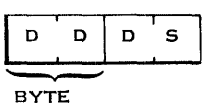
Show a three-digit field in the zoned format.





Show a three-digit field in the packed format.





A zoned decimal field can be changed to the packed format (in your own words) ____.

---

by a machine instruction called "pack."

To be able to use the instructions of the decimal feature, decimal fields must be in the ________ format.

---

packed

Decimal fields are variable in length and, as such, are processed using the storage-to-storage concept as previously discussed.

Variable length fields can start at _____ byte location in main storage.

---

any

The length of a variable length field is specified by (in your own words) ________________________________.

---

length code in the instruction

Decimal fields are __________ in length, and are processed using the _________-to-_________ concept.

---

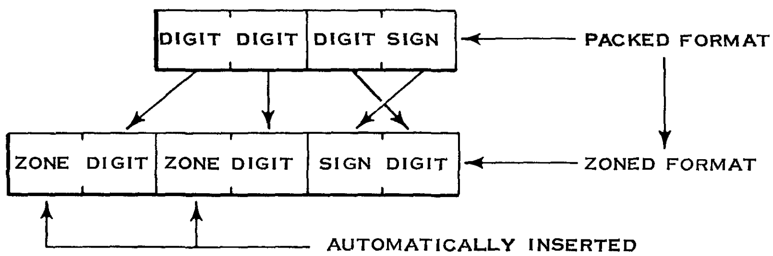
variable
storage-to-storage

To process fields using the decimal feature instructions, the fields must be in the ________ format.

---

packed

After decimal fields have been processed, they may be left in the packed format if the output medium is code insensitive. A magnetic tape unit would be a case of an output device that is code insensitive. If the output device is code sensitive, the packed fields must be changed to the zoned format. An example of a code sensitive output device is a card punch unit. Just as there was an instruction called "pack," there is an instruction called "unpack." The "unpack" instruction will change a packed format field to the zoned format as shown below.



After packed fields have been processed, they may be changed to the zoned format by (in your own words) ____

---

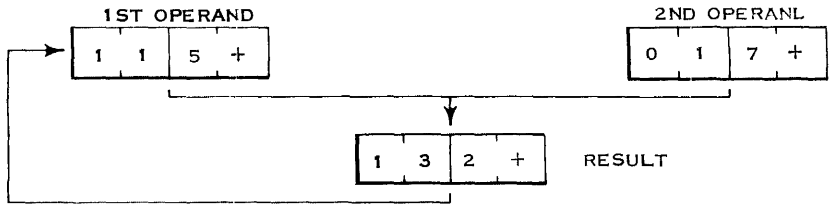
an instruction called "unpack"

Now that you know how to complement decimal numbers and the format of decimal fields, let's take a few examples of decimal arithmetic. Decimal fields, of course, will be in the packed format. These arithmetic operations will involve the storage-to-storage concept. This means that both operands are located in main storage and that the result will go back into main storage. What you may not realize about the storage-to-storage concept is that the result will replace one of the operands. In the System/360 the operands are referred to as the 1st and 2nd operands. In most System/360 operations involving two operands, the result of the operation replaces operand 1.

In the System/360, the results of decimal arithmetic operations replace the _____ operand.

1st

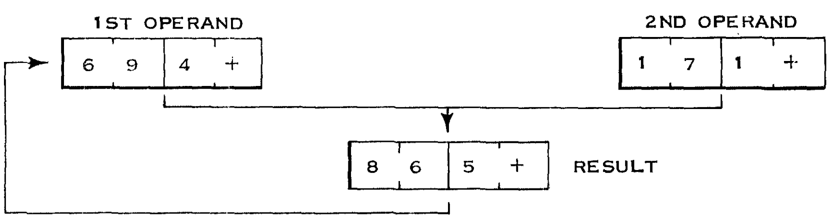
Supposing you wanted to add +17 to +115.



Notice that in the above example:

1. The result will replace the 1st operand.
2. The 2nd operand has a high-order zero digit. Packed decimal fields are variable by byte length, not by digit length.

In similar fashion as the preceding example, show the addition of +171 to +694.



The first step in a decimal arithmetic operation is sign analysis. In the above add operation the signs were the same, so you added the two operands. If the signs were different, the rules of addition would call for subtracting the smaller from the larger. Of course subtraction is handled by complement addition.

The first step in a decimal arithmetic operation is to analyze the ________.

---

signs

If the signs were different on an add operation, the two operands would be (in your own words) ______.

---

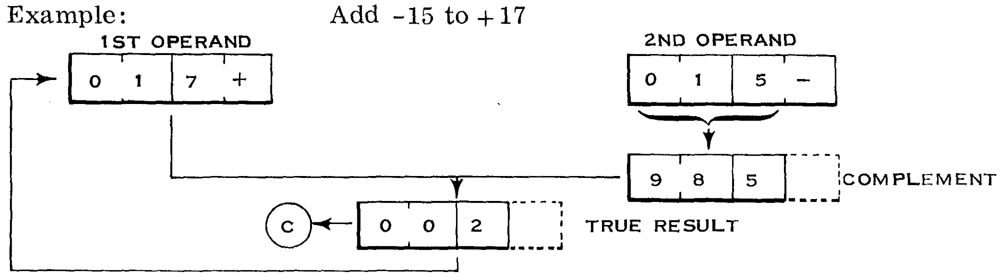
complement added

If the signs were alike on an add operation, the two operands would be (in your own words) ______.

---

added or more correctly <u>true</u> added.

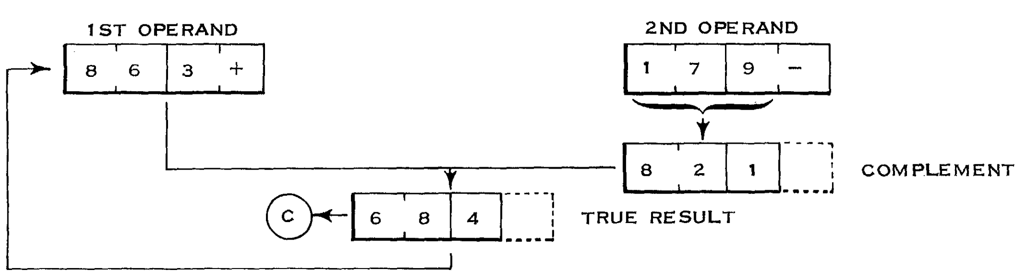
Example: Add -15 to +17



Notice that in the above example:

1. Because of sign analysis, the 2nd operand is complement added to the 1st operand.
2. The signs are analyzed but do not otherwise take part in the addition.
3. The carry out of the high order indicates that the answer is in true form and does not need to be re-complemented.

In a manner similar to the preceding example, show the addition of -179 to +863.



Supposing the operation says to subtract -17 from +115. According to the sign analysis, the two operands would be ______ added.

---

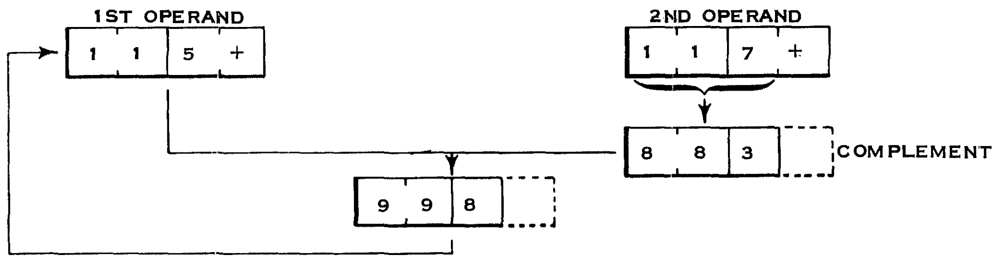
true

If the operation says to subtract +17 from +115, the two operands would be ____________ added.

---

complement

Example: Subtract +117 from +115



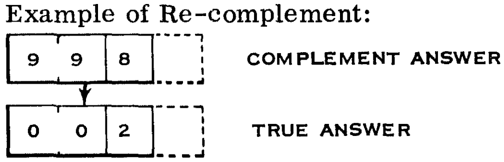
The absence of a carry in the above example indicates (in your own words) ____.

---

the answer is in complement form.

Complement answers will be automatically re-complemented so that the result will be in a true form.

Example of Re-complement:



Since re-complementation also involves a sign change, the 1st operand will contain ______________________ at the end of the operation.

---

a new sign (minus)

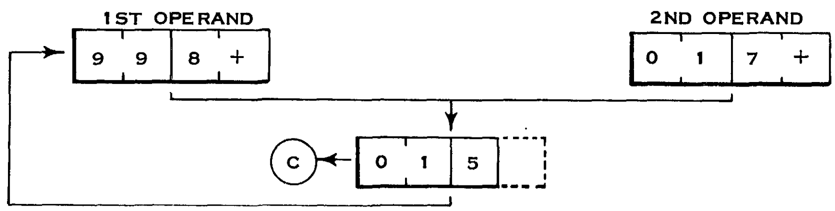
The absence of a high-order carry during complement addition indicates that the result is in ____________ form and needs to be _______________.

The presence of a high-order carry during complement addition indicates that the answer is in ______ form.

---

complement
re-complemented
true

In similar manner to previous examples, show the addition of +17 to +998.



In the preceding problem, because of sign analysis, the two operands were ______ added.

---

true

Notice that, in the preceding problem, there was a carry out of the high-order position. This carry is lost. Whenever a carry occurs out of the high-order during true addition of decimal fields, it is known as a "decimal overflow."

The presence of a high-order carry during true addition indicates a ________ __________.

---

decimal overflow

A carry out of the high-order during true addition is called a ________ __________ and occurs because the result is too _______ (large/small) to be contained in the 1st operand location.

---

decimal overflow
large

A decimal overflow will occur anytime all the significant digits of true addition cannot be contained in the length of the 1st operand.

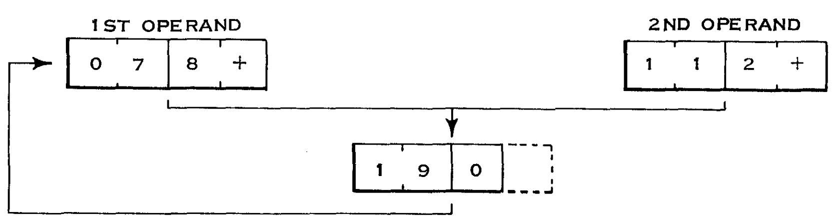
One way for a decimal overflow to occur during true addition is to have (in your own words) ______________________.

---

a carry out of the high-order

A decimal overflow will also occur if the 2nd operand contains more significant digits than the 1st operand has room for.

Example: Add +112 to +78

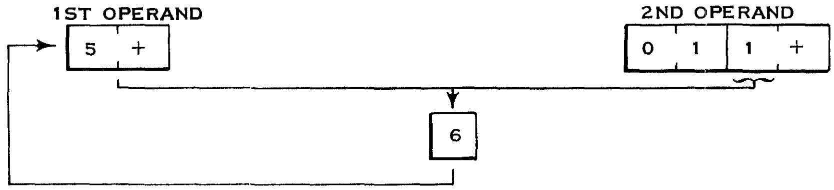


In the above example a decimal overflow __________ occur because the 1st operand has room for all significant digits.

---

will not

Example: Add +11 to +5

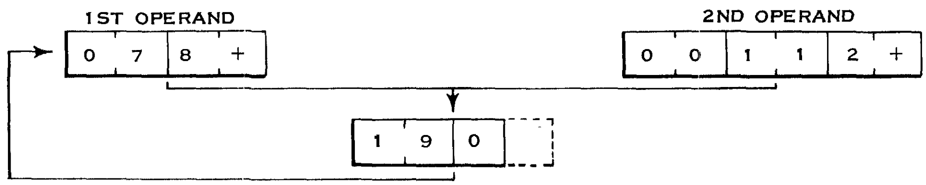


In the above example a _________ _________ would occur because all significant digits of the result cannot be contained in the 1st operand.

---

decimal overflow

Example Add +00112 to +078



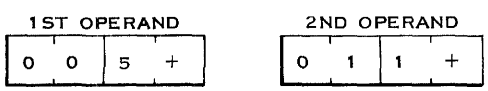
In the above example a decimal overflow _________ occur because the 1st operand location can contain all significant digits of the result.

---

will not

The next question is: How can a programmer make sure a decimal overflow will not occur during decimal arithmetic operations? A good method would be to make the 1st operand long enough (by having high-order zeros) to accommodate 1) any possible high-order carry as well as 2) all significant digits from the 2nd operand.

You wish to add +11 to +5. Show the operands necessary to avoid a decimal overflow.



Remember, a partial byte cannot be used as an operand.

Go to the IBM System/360 Principles of Operation manual and briefly study the Decimal Arithmetic area of the System Structure section.

REVIEW QUESTIONS ON DECIMAL FORMATS
AND EXTENDED BCD CODE

- Try to answer the questions without referring to the material. However, if you do require aid, refer to this book and/or the System/360 Principles of Operation manual and consider reviewing the area where aid is required.

1. The extended BCD code uses _______ bits to express a character.

2. The numeric portion of a character uses bits _____ of a byte.

3. Bits 2 and 3 of the extended BCD code are the reverse counterpart of the ___________ bits of the standard BCD code for alphameric information.

4. What is the extended BCD code for the character "2"? ________

5. Decimal data in the extended BCD code is said to be in the ______ ______________ format.

6. The packed format has _____ digits in each byte except the low-order byte which has the sign in bits ______.

7. Show a three-digit field in the zoned format. [ ]

8. Show a three-digit field in the packed format. [ ]

9. To use the instructions of the decimal feature, decimal fields must be in the ______ format.

10. The length of a decimal field is specified by (in your own words) ____.

11. Decimal fields are processed using the _________-to-__________ concept.

12. Results of decimal arithmetic operations replace the _____ operand.

13. The first step in a decimal arithmetic operation is to analyze the _______.

14. If the signs were different on an add operation or alike on a subtract operation, the two operands would be ____________________.

15. A high-order carry during complement addition indicates (in your own words) ______________________________.

16. The absence of a carry during complement addition indicates (in your own words) ____.

17. Besides complementing the result, re-complementation also involves a ______ ________.

18. Name two items that can cause a decimal overflow.

a. ____

b. ____

ANSWERS TO REVIEW QUESTIONS

1. eight

2. 4-7

3. B and A

4. 1 1 1 1 0 0 1 0

5. zoned or unpacked

6. two, 4-7

7.

ZONE	DIGIT	ZONE	DIGIT	SIGN	DIGIT

8.

DIGIT	DIGIT	DIGIT	SIGN

9. packed

10. a length code in the instruction

11. storage-to-storage

12. 1st

13. signs

14. complement added

15. the answer is in true form

16. the answer is in complement form and needs to be re-complemented. NOTE: Re-complementation is done automatically by the computer.

17. sign change

18. a. High-order carry during true addition.
    b. 2nd operand has more significant digits than the 1st operand has room for.

BINARY DATA FORMATS

You have just learned the decimal data formats. Decimal data is variable in length and is processed with the storage-to-storage concept. Binary data is fixed in length and is processed with both the storage-to-accumulator and accumulator-to-accumulator concepts. Let's see what you remember about fixed length operations on the System/360.

Each main storage address refers to a unique ______ location.

Data fields are addressed by their ______ (high/low) order byte location.

---

byte
high

A halfword is _____ bytes long.

A word is ______ bytes long.

A doubleword is _______ bytes long.

---

two
four
eight

The storage address of fixed length data fields must be divisible by the number of _______ in the field or a s_____________ exception will occur.

---

bytes
specification

Fixed length operations use the storage-to-accumulator concept. For use as accumulators, the programmer has available (in your own words) ______________________________.

---

16 general registers

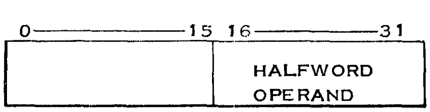
How many bytes may be contained in a general register? ________

---

four (4)

Number the bit positions of the general register below. Also show where a halfword operand would be placed.

GENERAL REGISTER

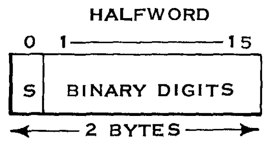


GENERAL REGISTER

Whereas the length of decimal data was specified by a length code in the instruction, the length of binary data is implied by the Op code of the instruction. Binary operands may be either a halfword or a word in length, depending on the INSTRUCTION. Let's discuss halfword operands first.

HALFWORD BINARY OPERANDS

A halfword binary operand is two bytes in length and can be used to express numbers which do not exceed a value of $2^{15} - 1$ (32,767).



As can be seen above, the high order of a halfword is used to represent the sign.

A halfword binary operand is _____ bytes in length. The sign of a halfword operand is represented by (in your own words) ____________ ____________.

---

two
the high-order bit

Halfword operands use only the storage-to-accumulator concept. The 1st operand is located in the low-order (bits 16-31) of a general register and the 2nd operand is located in main storage. As with decimal arithmetic operations, the results of binary arithmetic operations replace the 1st operand.

The address of the 2nd operand in halfword operation must be divisible by _____ or a _____________ exception will occur.

---

two
specification

The 1st operand is located in (in your own words) ____________ ____________.

---

bits 16-31 of a general register

The result of a binary arithmetic operation replaces the _____ operand.

1st

The System/360 does its binary calculations in a rather unique way. As you have already seen, decimal numbers were represented in their true form (absolute value) with a + or - sign. The System/360 does not represent binary numbers in this manner.

Positive binary numbers are represented in their true form while negative numbers are represented in their complement form. The sign or high-order bit is 0 for positive numbers and is 1 for negative or complement numbers.

In the System/360, decimal numbers are represented in their ______ form with a __________.

---

true
+ or - sign

Positive binary numbers are represented in their ______ form with a ___ in the high-order bit position.

Negative binary numbers are represented in their ____________ form with a ___ in the high-order bit position.

---

true
0
complement
1

Represent the decimal value +17 as a halfword binary operand. (The small vertical lines within the operand box have no significance; they simply break up the operand into groups of 4 bits so that the number is easier to read)

---

0 0 0 0	0 0 0 0	0 0 0 1	0 0 0 1

The sign bit position in the preceding answer is 0. This indicates that the binary number is positive and is in its true form.

Represent the decimal value -17 as a halfword binary operand.

---

1 1 1 1	1 1 1 1	1 1 1 0	1 1 1 1

The preceding answer is the complement of 17. The high-order 1 bit tells us that the operand is negative and therefore is represented in its complement form.

Show the largest positive binary number that can be in a halfword operand.

---

0 1 1 1	1 1 1 1	1 1 1 1	1 1 1 1

Any positive binary number larger than the preceding answer would need a 1 in the high-order bit position. The high-order bit is reserved as a sign bit. A sign bit of 1 would indicate that the number is negative and is in its "twos" complement form. It is very important to remember that negative numbers are always represented in their "twos" complement form.

Show the value of -1 in a halfword binary format.

---

1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1

To verify the preceding answer, express the absolute value of the number in true form and then complement it.

Example: 0000000000000001 ⟵ absolute value of 1

To complement a binary number, invert each bit and add 1.

```
1111111111111110
              +1
1111111111111111 <----- "Twos" complement of 1
                        or a -1
```

As you recall from our earlier discussion on numbering systems, the value of zero cannot be complemented. Since negative numbers are represented in their complement form in the System/360, there can be no minus zero. This is desirable in arithmetic operations.

Show the value of +1 in a halfword binary format. ______________

0000000000000001

Show the largest positive binary number that can be represented in a halfword. ______________

0111111111111111

Show the value of zero in a halfword binary format. ______________

0000000000000000

Show the value of -1 in a halfword binary format. ______________

1111111111111111	Show the value of -2 in a halfword binary format. ________________
1111111111111110	Show the largest negative binary number that can be represented in a half-word binary format. ________________
1000000000000000	In the System/360 binary numbers are contained in fixed length operands. At this point we are discussing halfword formats. A halfword consists of 16 bits (2 bytes). Of these sixteen bits, bits 1 to 15 represent the number, while bit 0 represents the sign. However, bit 0 does not actually represent a plus or minus sign. Instead it indicates whether bits 1 to 15 contain a true number or a complement number. The total range of the sixteen bits of a halfword operand would look like this:

Complement or Negative Numbers	1111111111111111 ⟶	(Minus one)
	↓	↓
	1000000000000000 ⟶	(Largest negative value)
True or Positive Numbers	0111111111111111 ⟶	(Largest positive value)
	↑	↑
	0000000000000001 ⟶	(Plus one)
	0000000000000000 ⟶	(Zero)

Go to the Principles of Operations manual and briefly study Appendix B.

If bit position 0 contains a 0 bit, it indicates that the halfword is the _____ form of a __________ number.

If bit position 0 contains a 1 bit, it indicates that the halfword is the ____________ form of a __________ number.

---

true
positive
complement
negative

A halfword binary number is placed into bit positions ________ of a general register.

---

16 to 31

When a halfword is placed or loaded into a general register, the halfword is expanded to a fullword by propogating the sign bit to the left. In other words, bits 0 to 16 will be the same. Bit position 16, of course, will contain the sign of the halfword operand.

HALFWORD OPERAND IN A GENERAL REGISTER

0 ——— 15	16	17 ——— 31
SAME AS SIGN BIT	S	BINARY NUMBER

When a halfword is loaded into a general register, it is expanded to a __________________________.

---

word or fullword

A halfword is expanded to a fullword by (in your own words) ________________________ .

---

propagating the sign bit to the left.

Show the contents of a general register after it has been loaded with the following halfword operand from main storage:

1 0 1 1 1 1 1 1	1 1 1 1 1 1 1 1			HALFWORD IN MAIN STORAGE
				GENERAL REGISTER

---

1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1	1 0 1 1 1 1 1 1	1 1 1 1 1 1 1 1

Show the contents of the preceding general register as 8 hex digits.

____

FFFFBFFF

From the preceding discussion you should now realize that halfword operands exist only in main storage. When a halfword is loaded into a general register, it is expanded to a fullword.

FULLWORD BINARY OPERANDS

For use with the binary arithmetic instructions, binary operands in the System/360 may be either a halfword or a word in length. Halfwords are processed using the storage-to-accumulator concept. Binary operands in the general registers are always a word in length. Halfwords are expanded to fullwords whenever they are placed into a general register.

Halfword binary operands are processed using the _______ to _______ concept. Binary operands in a general register are a _______ in length.

---

storage
accumulator
word

Binary operands which are a word in length may reside either in main storage or in a general register. Since there are sixteen general registers, word operands may be processed using either the storage-to-accumulator concept or the accumulator-to-accumulator concept.

Word operands may reside in either _______ _______ or in the __________. When in main storage they must have an address divisible by ______ or a specification exception will occur.

---

main storage
general registers
four

A specification exception is a type of p________ check.

---

program

Besides the storage-to-accumulator concept, word operands may be processed using the __________-to-__________ concept.

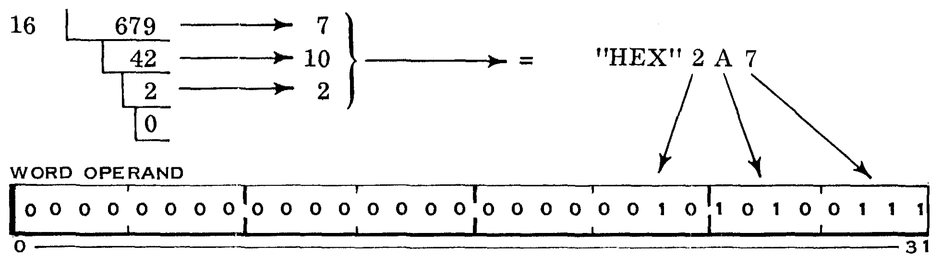
---

accumulator-to-accumulator or register-to-register

As with halfword operands, bit position 0 of a word operand is the sign bit and indicates whether the word is a positive number in true form or a negative number in complement form.

Let's show the value (+679) as a word operand. We'll convert the decimal value to hexadecimal first and then to binary.

Example:



Show the value (-679) as a word operand. Remember this is a negative number and will appear in complement form.

"DEC" 679 = "HEX" 2A7

2A7 = 0000 0000 0000 0000 0000 0010 1010 0111

-2A7 = 1111 1111 1111 1111 1111 1101 0101 1001

From the two preceding problems, you can see that binary operands basically are unsigned numbers in either true or complement form.

The System/360 carries its decimal operand in the packed format. The operands are in their true form with a plus or minus sign in the low-order position. To add or subtract decimal operands, System/360 first has to analyze the signs. Then the two operands would be true or complement added according to the rules of addition. If complement addition did not result in a high-order carry, it meant the answer had to be re-complemented.

Decimal operands are in the true form with a (in your own words) ____________________ in the low-order position.

---

plus or minus sign

The first step in a decimal arithmetic operation is to (in your own words) ____________________.

---

analyze the signs

If the answer to a decimal operation is in complement form, the System/360 has to ______________ it.

---

re-complement

Because binary operands are basically unsigned numbers in either true or complement form, there is no need for sign analysis. If the instruction says ADD, the binary fields are added. If the instruction says SUBTRACT, the binary fields are complement added. Since negative numbers are carried in complement form, there is no need for re-complementing.

Negative binary numbers are carried in their ____________ form.

---

complement

In binary operations there _______ (is/isn't) a need for sign analysis.

---

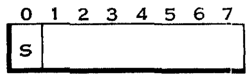
isn't

If the result of a binary operation is in complement form, it ________ (does/does not) have to be re-complemented. This is because a complement answer indicates a __________ result.

---

does not negative

Let's take a look at System/360 binary operations. For simplicity, binary operands will be shown as 8 bits in length.



Show +15 as an 8 bit binary operand.

---

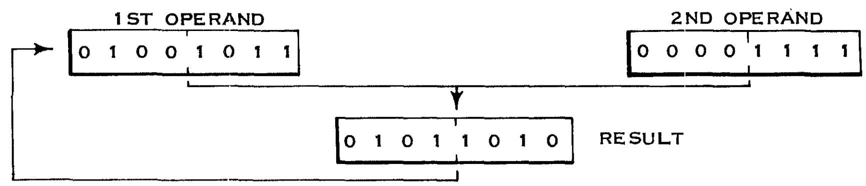
0 0 0 0 1 1 1 1

Show +75 as an 8 bit binary operand.

---

0 1 0 0 1 0 1 1

The System/360 would add the two preceding operands like this:
+15 added to +75 =



Now supposing the operation is to add -15 to +75.

Show the -15 as an 8 bit binary operand.

---

1 1 1 1 0 0 0 1

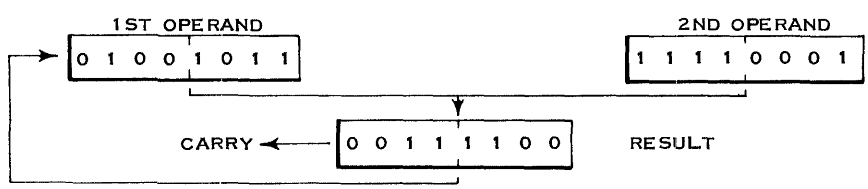
Remember now that negative numbers are represented in complement form.

The preceding operands (-15 and +75) would be ______ added by the System/360.

---

true; The operation says to add and there is no sign analysis to be done.

Example of adding -15 to +75



If you were to convert the result to decimal, you would come up with a +60. As you can see, this is certainly the correct answer.

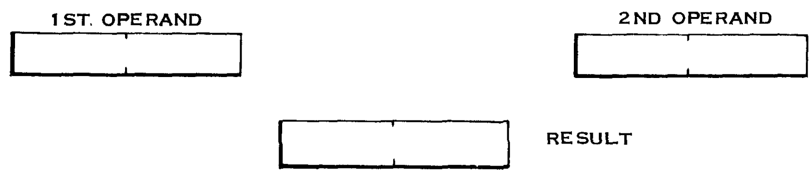
Suppose now that the operation is to add -15 to -75.

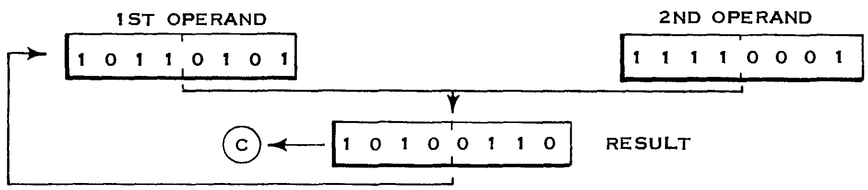
Show the first operand (-75) as an 8 bit binary operand. [   ]

---

1 0 1 1 0 1 0 1 - Since negative numbers are represented in complement form, all you should have done was to complement the 1st operand (+75) from the previous problem.

Show the addition of these two operands: (-15 to -75) Also show the data flow lines.





The result of the preceding problem is in ____________ form and represents a __________ number.

---

complement
negative

10100110 is the complement of __________.

01011010

01011010 converted to decimal is ____.

90; This agrees with our original problem: Add -15 to -75.

Supposing you want to subtract +15 from +90.

Show the 1st operand (+90) as an 8 bit binary operand. [   ]

---

0 1 0 1 1 0 1 0

Show the 2nd operand (+15) as an 8 bit binary operand. [   ]

---

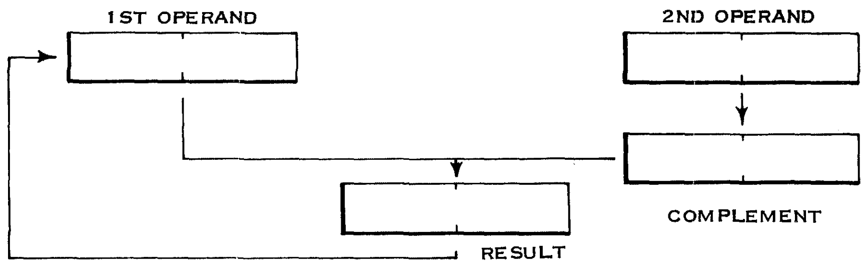
0 0 0 0 1 1 1 1

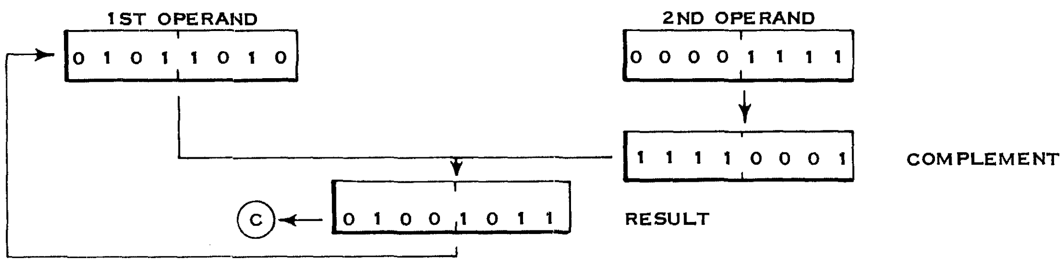
To subtract +15 from +90 the System/360 will ____________ add the 2nd operand to the 1st operand.

---

complement

Show the subtraction (complement addition) of the two operands (+15 from +90).





The result in the preceding problem is in ______ form and has a decimal value of _____.

---

true
+75

In these problems, you have been working with 8 bit binary operands. Of course, the System/360 uses halfword and word binary operands. The arithmetic principles involved, however, are the same regardless of length.

Binary operands in the System/360 are a ________ or ______ in length.

---

halfword
word

The length of a binary operand is implied by the I____________.

Instruction; More specifically the length is implied by the Op code portion of the instruction.

Positive numbers are in ______ form while negative numbers are in ____________ form.

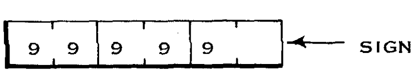
---

true
complement

Bit position ______ of a binary operand indicates whether the operand is in true or __________ form. If bit position 0 is a 1, the operand is in ____________ form and represents a __________ number.

---

zero complement complement negative	Prior to adding or subtracting binary operands, the signs ________ (are/are not) analyzed.
are not	If the instruction is ADD, the two operands are ______ added. If the instruction is subtract, one of the operands is ______________ and then added to the other operand.
true complemented	If the answer to a binary operation is in complement form, it represents a __________ number and ______ (is/isn't) re-complemented.
negative isn't	There is one final principle of System/360 binary operations to be learned. This is the Fixed Point Overflow. Earlier you learned about the decimal overflow and what caused it. Read the following review frames on decimal overflow.
	The absence of a high-order carry during complement addition of decimal fields indicates that the result is in c____________ form and needs to be ________________.
complement re-complemented	The presence of a high-order carry during complement addition of decimal fields indicates that the result is in ______ form.
true	The presence of a high-order carry during true addition of decimal fields indicates a __________ ____________.
decimal overflow	Anytime all the significant digits of a decimal operation cannot be contained in the resulting field a ___________ ____________ will occur.
decimal overflow	A decimal overflow indicates that the decimal result is not correct. The result has exceeded the maximum value that could be contained in the result field (1st operand). The maximum decimal value that can be contained by a three byte 1st operand is ____________________________.



3 BYTE PACKED OPERAND

A fixed point overflow indicates that the result of a binary operation is not correct. The result has exceeded the maximum value that could be contained in the result field (1st operand). Of course binary operands are fixed in length and may be either a halfword or word in length.

Show the largest positive binary number that can be represented in a halfword. ____________

0111111111111111

Show the largest negative binary number that can be contained in a halfword. ____________

1000000000000000

Show the largest positive binary number that can be contained in a word.

____

01111111111111111111111111111111

If the value of +1 were added to the largest positive binary number, a ______ ______ overflow would occur.

---

fixed point

When the result of a binary operation exceeds the maximum value that can be contained in the result field (1st operand), a ______ ______ ________ will occur.

---

fixed point overflow

If the value of -1 were added to the largest negative binary number, a ______ ______ ________ would occur.

---

fixed point overflow

For the sake of simplicity in the following examples, let's again consider the binary operands as being only 8 bits (one byte) in length.

Show the value of +1 as a one byte operand. [ ]

[0 0 0 0 0 0 0 1]

Show the largest positive binary number that could be contained in one byte ________.

01111111

Show the value of -1 as a one byte operand. [ ]

[1 1 1 1 1 1 1 1]

Show the largest negative binary number that could be contained in one byte ________.

10000000

Show the result of adding +1 to the largest positive binary number.

```
0 000 0001     (+1)
0 111 1111     (largest positive number)
```

10000000; Notice that this is the largest <u>negative</u> number.

As a result of the preceding addition, a ________ ________ ________ would occur.

---

fixed point overflow

Show the result of adding -1 to the largest negative binary number.

```
1 111 1111     (-1)
1 000 0000     (largest negative number)
```

---

0 111 1111; Notice that this is the largest <u>positive</u> number.

As a result of the preceding addition, a ________ ________ ________ would occur.

---

fixed point overflow

Whenever the largest negative or positive binary number is exceeded in a binary operation, a ________ ________ ________ will occur.

---

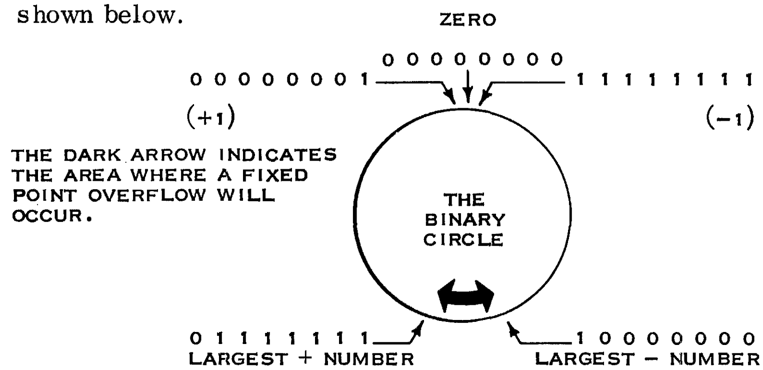
fixed point overflow

Let's review some of the principles of the binary numbering system as used in the System/360. Again for the sake of simplicity, the binary numbers are shown as one byte in length. Actually they are halfwords or words when used in System/360. The principles are the same, however, regardless of length.

```
             ( 1111 1111 ------> (Minus one)
Complement   |                        ↕
or negative  |
numbers      ( 1000 0000 ------> (Largest negative value)

             ( 0111 1111 ------> (Largest positive value)
True or      |                        ↕
positive     | 0000 0001 ------> (Plus one)
numbers      ( 0000 0000 ------> (Zero)
```

The range of numbers from 00000000→01111111 represent positive numbers in true form. The range of numbers from 10000000→11111111 represent negative numbers in complement form. The binary numbering system as used in the System/360 can best be illustrated by a circle as shown below.



Whenever the largest negative or positive binary number is exceeded in value a ________ ________ ________ will occur. When it occurs the sign bit of the result will be ________ from the original number.

---

fixed point overflow
changed or different

Although the sign bit is changed when a fixed point overflow occurs, a sign change is not the cause of the overflow. Instead it is the result of it. The question then is "How does the System/360 know when a fixed point overflow has occurred?" Let's take a look at the addition of +1 to the largest binary number.

SIGN	INTEGER
0	111 1111
0	000 0001
1	000 0000

There was a carry out of the high-order integer bit but there ________ (was/was not) a carry out of the sign bit.

---

was not

When the largest positive binary number is exceeded in value, there is a ________ out of the integer, but there ________ (is/is not) a carry out of the sign bit.

---

carry
is not

Let's take a look at the addition of -1 to the largest negative number.

```
  1 000 0000   (largest negative number)
  1 111 1111   (minus one)
C 0 111 1111
```

There was no carry out of the high-order integer bit. However there was a ________ out of the sign bit.

---

carry	When the largest negative value is exceeded in value, there ________ (is/is not) a carry out of the integer but there is a carry out of the ______ bit.
is not sign	In both of the previous examples, the carry out of the sign bit position and the carry out of the integer did not agree. This is the way System/360 detects a _________ _________ overflow.
fixed point	System/360 detects a fixed point overflow whenever the carry out of the _________ does not agree with the carry out of the _______.
integer sign (in either order)	Indicate which of the following additions will result in a fixed point overflow.

a.
```
  0 111 0000
  0 010 0111
```
(overflow/ no overflow)

b.
```
  1 100 0000
  1 000 1000
```
(overflow/ no overflow)

c.
```
  0 011 0000
  0 100 0100
```
(overflow/ no overflow)

d.
```
  1 111 1111
  1 111 1111
```
(overflow/ no overflow)

---

a.
```
  0 111 0000
  0 010 0111
  1 001 0111
```
overflow (integer carry, no sign carry)

b.
```
  1 100 0000
  1 000 1000
c 0 100 1000
```
overflow (no integer carry, sign carry)

c.
```
  0 011 0000
  0 100 0100
  0 111 0100
```
no overflow (no integer carry, no sign carry)

d.
```
  1 111 1111
  1 111 1111
c 1 111 1110
```
no overflow (integer carry, sign carry)

Go to the IBM System/360 Principles of Operation manual and briefly study the Fixed Point Arithmetic area of the System Structure section.

REVIEW QUESTIONS ON BINARY FORMATS

- Try to answer the questions without referring to the material. However, if you do require aid, refer to this book and/or the System/360 Principles of Operation manual and consider reviewing the area where aid is required.

1. A halfword is ______ bytes long while a word is ________ bytes long.

2. A halfword in main storage is addressable by its ________ (high/low) order byte location.

3. The main storage address of a halfword must be divisible by ______ or a __________________ exception will occur.

4. The results of binary arithmetic operations replace the _____ (1st/2nd) operand.

5. Positive binary numbers are represented in their ________ form with a ___ (1/0) in the high-order bit position.

6. Negative binary numbers are represented in their _____________ form with a ___ bit in the high-order bit position.

7. Represent the decimal value +26 as a halfword binary operand.

8. Represent the decimal value -1 as a halfword binary operand.

9. Show the largest negative binary number that can be represented in a halfword binary operand.

10. Show the above halfword after it has been placed in a general register.

11. Halfword operands are processed using the _________-to-________________ concept.

12. Prior to adding or subtracting binary operands, the signs ___________ (are/are not) analyzed.

13. Whenever the largest negative or positive number is exceeded in a binary operation, a ______ ________ ____________ will occur.

14. A fixed point overflow is detected when the carry out of the __________ does not agree with the carry out of the ______.

15. Do the following additions and indicate whether or not a fixed point overflow will occur.

a. 0 111 0001
0 000 1010 (overflow/ no overflow)

b. 0 101 0001
0 011 0010 (overflow/ no overflow)

c. 0 001 0111
1 000 0100 (overflow/ no overflow)

d. 1 001 0000
1 011 0001 (overflow/ no overflow)

ANSWERS TO REVIEW QUESTIONS

1. two, four

2. high

3. two, specification

4. 1st

5. true, 0

6. complement, 1

7. | 0 0 0 0 0 0 0 0 | 0 0 0 1 1 0 1 0 |

8. | 1 1 1 1 1 1 1 1 | 1 1 1 1 1 1 1 1 |

9. | 1 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 |

10. | 1 1 1 1 1 1 1 1 | 1 1 1 1 1 1 1 1 | 1 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 |

11. Storage-to-Accumulator (or Register)

12. are not

13. fixed point overflow

14. integer, sign (in either order)

15. a. 01111011 no overflow (no sign or integer carry)

    b. 10000011 overflow (integer carry, no sign carry)

    c. 10011011 no overflow (no sign or integer carry)

    d. 01000001 overflow (no integer carry, sign carry)

Do you need a review? If you think that you may require a review of areas of this book, do the following:

Read the learning objectives at the beginning of each section.

You should review only those areas where you think that you cannot do what the objective indicates.

Before proceeding to the next book of this System/360 Introductory Programming Course, fill out the Course Evaluation Sheet (located in the back of this book).

Alphabetical Index

Student Course Evaluation

You can make this course and all future courses more useful by answering the questions on both sides of this sheet and giving us your comments.

Do you feel that you have an adequate understanding of the learning objectives that are listed at the beginning of the following sections?

Section I:	Numbering Systems	Yes ☐	No ☐
Section II:	Organization	Yes ☐	No ☐
Section III:	Data Formats	Yes ☐	No ☐

List any technical errors you found in this book.

Comments

Please complete the information block on the opposite side. Thank you for your cooperation.

For form R23-2933-1

Field Engineering Education - Student Course Evaluation **IBM**

Student Name	Man Number	B/O Number	Area Number

Student: Please review this evaluation with the person administering the course; then remove it from the book and send to the FE Education Center via IBM mail.

- Were you given a copy of this text to write in and keep? . . . . Yes ☐ No ☐
- How many hours per day were scheduled for this course? ________
- Were you interrupted during this time? . . . . Yes ☐ No ☐
- How many hours were needed to complete this course? ________
- Did you require assistance during this course? . . . . Yes ☐ No ☐
  (If your answer is yes, explain in the comments section)

- Indicate your understanding of the total course. Excellent ☐ Good ☐ Fair ☐ Poor ☐

To be completed by course administrator	Date
Reviewed by:	

To be completed by FE Education Planning	Date
Reviewed by:	

FOLD

IBM Corporation
FE Education Planning
Department 911
South Road
Poughkeepsie, N. Y. 12602

FOLD

TEAR HERE

TEAR HERE

R23-2933-1

IBM
International Business Machines Corporation
Field Engineering Division
112 East Post Road, White Plains, N.Y. 10601

System/360 Intro-Book 1 Student Self-Study Course Printed in U.S.A. R23-2933-1