Machine language

(Assembly language, High level language, mid-level Selection, Compilers, Interpreters, Programming language)

Lecturer: Ghosh K.P.

Hardware

- Hardware is the machine itself and its various individual equipment.
- It includes all mechanical, electronic and magnetic devices such as monitor, printer, electronic circuit, floppy and hard disk.

Software

Software refers to the set of computer programs, which are used in applications and operating systems.

- It is the collection of programs, which increase the capabilities of the hardware.
- Software guides the computer at every step where to start and stop during a particular job.
- The process of software development is called *programming*.

SOFTWARE TYPES

❑ 1- Application Software :

- Application Software is a set of programs for a specific application.
- Application software is useful for word processing, accounting, producing statistical report, Graphics, Excel and Data Base.
- programming languages COBOL, FORTRAN, C++, VB, VC, Java

❑ 2- System Software:

- When you switch on the computer the programs written in ROM is executed which activates different units of your computer and makes it ready for you to work.
- This set of programs can be called system software.
- System software are general programs designed for performing tasks such as controlling all operations required to move data into and out of the computer
- System Software allows application packages to be run on the computer.

Programming languages

Programming languages specially developed so that you could pass your data and instructions to the computer to do specific job.

There are two major types of programming languages: Low Level Languages and High Level Languages

- Low Level languages are further divided in to *Machine language* and *Assembly language*
- High Level Languages are, for scientific application FORTRAN and C languages are used. On the other hand COBOL is used for business applications.

Machine Language

Machine Language is the only language that is directly understood by the computer. It does not needs any translator program

- The only advantage is that program of machine language run very fast

Assembly Language

It is the first step to improve the programming structure. You should know that computer can handle numbers and letter.

- The set of symbols and letters forms the Assembly Language and a translator program is required to translate the Assembly Language to machine language
- This translator program is called `*Assembler*

About the Assembly Language

- Advantages:
  - Assembly Language is easier to understand and saves a lot of time and effort.
  - It is easier to correct errors and modify program instructions.
  - Assembly Language has the same efficiency of execution as the machine level language
- Disadvantages:
  - Assembly language is machine dependent. A program written for one computer might not run in other computers with different hardware configuration.

HIGH LEVEL LANGUAGES

- Assembly and machine level languages require deep knowledge of computer hardware where as in higher language you have to know only the *instructions in English words and logic of the problem.*
- Higher level languages are simple languages that use English and mathematical symbols like +, -, %, / etc. for its program construction
- Any higher level language has to be converted to machine language for the computer to understand
- For example COBOL (Common Business Oriented Language), FORTRAN (Formula Translation) and BASIC (Beginners All-purpose Symbolic Instruction Code) are high level languages

HIGH LEVEL LANGUAGES

❑ Advantages of High Level Languages

- Higher level languages have a major advantage over machine and assembly languages that higher level languages are easy to learn and use (similar to the languages used by us in our day to day life.

The Evolution of Programming Languages

- Higher-level languages are more powerful than assembly language and allow the programmer to work in a more English-like environment.
- Higher-level programming languages are divided into three "generations," each more powerful than the last:

a) Third-generation languages
b) Fourth-generation languages
c) Fifth-generation languages

Higher-Level Languages (3G, 4G)

Third-generation languages (3GLs) are the first to use true English-like phrasing, making them easier to use than previous languages.

3GLs are portable, meaning the object code created for one type of system can be translated for use on a different type of system.

The following languages are 3GLs:

FORTAN, C, COBOL, C++, BASIC, Java, Pascal, ActiveX

Fourth-generation languages (4GLs) are even easier to use than 3GLs.

4GLs may use a text-based environment (like a 3GL) or may allow the programmer to work in a visual environment, using graphical tools.

The following languages are 4GLs: Visual Basic (VB), VisualAge Authoring environments

Higher-Level Languages (5G)

Fifth-generation languages (5GLs) are an issue of debate in the programming community – some programmers cannot agree that they even exist.

These high-level languages would use artificial intelligence to create software, making 5GLs extremely difficult to develop.

Solve problems using constraints rather than algorithms, used in Artificial Intelligence. (Prolog)

Compiler

*Compiler :*It is a program translator that translates the instruction of a higher level language to machine language.

- It is called compiler because it compiles machine language instructions for every program instructions of higher level language.
- Thus compiler is a program translator like assembler but more sophisticated. It scans the entire program first and then translates it into machine code.

The programs written by the programmer in higher level language is called *source program*. After this program is converted to machine languages by the compiler it is called *object program.*

A compiler can translate only those source programs, which have been written, in that language.

Interpreter

- An interpreter is another type of program translator used for translating higher level language into machine language.
- It takes one statement of higher level languages, translate it into machine language and immediately execute it.
- Translation and execution are carried out for each statement.
- It differs from compiler, which translate the entire source program into machine code.

Advantage & Disadvantage Interpreter

- The advantage of interpreter compared to compiler is its fast response to changes in source program
- do not require large memory in computer.
- The disadvantage of interpreter is that it is time consuming method because each time a statement in a program is executed then it is first translated.
- Thus compiled machine language program runs much faster than an interpreted program.

DATA PROCESSING SYSTEMS

Presentation Outline

I. Paper-Based Input Systems
II. Paperless Input Systems
III. Paper-Based Processing Systems
IV. Paperless Processing Systems
V. The Output System

Paper-Based Input Systems

- ✓ Preparation and Completion of the Source Document.
- ✓ Transfer of Source Documents to Data Processing
- ✓ Data Entry
- ✓ Program Data Editing

Preparation and Completion of the Source Document

Source documents such as sales orders are filled in manually. Errors are minimized if source documents are well designed and easy to understand.

Transfer of Source Documents to Data Processing

1. Input Document Control Form Provides batch control totals for batches of input data transmitted between user and data processing departments. Monitors completeness of processing.
2. Data Transfer Log Provides control over the disposition and use of transferred data.

Data Entry

Key verification takes place to ensure accurate data entry. This involves typing in the data twice. As the data is typed in a second time, key verification software verifies that the second keying matches the previous input already on the disk file. A process of visual verification is less effective.

Program Data Editing

Program data editing is a software technique used to screen data for errors prior to processing. Examples include:

1. Table Lookup – Entry must match values in a table.
2. Limit Test – Entry must not exceed an extreme value.
3. Continuous operations auditing – Unacceptable entries are separated and held in suspense until verified or corrected.
4. Check digit – Redundant digit determined by a mathematical calculation.

Paperless Input Systems

- Loss of Internal Controls
- Paperless Input Systems Requiring Human Intervention
- Paperless Input Systems Requiring No Human Intervention

Loss of Internal Controls

In on-line input systems, the need for keying in source documents may be eliminated. The loss of paper-based internal controls can be compensated for by:

1. Transaction logs (registers) – Logs all inputs to a special file that automatically contains tags to identify transactions.
2. Tagging – Provides an audit trail by including additional audit-type information with transaction data. Audit-type information might include dates and user authorization codes.

Paperless Input Systems Requiring Human Intervention

1. On-line manual data-entry systems

   Users manually type transactions into the computer system.

2. Automatic Identification System

   Merchandise and other items such as credit cards are tagged with machine-readable codes.

Paperless Input Systems Requiring No Human Intervention

In some systems, transactions are processed from beginning to end without any human intervention.

An example would be a system that monitors inventory levels and places an order directly with a supplier's system that automatically initiates shipment of the needed goods.

Paper-Based Processing Systems

Virtually all paper-based systems for processing transactions are batch oriented.

- Batch Processing with Sequential File Updating
- Son-Father-Grandfather Master Files
- Batch Processing with Random-Access File Updating

Batch Processing with Sequential File Updating

Four steps include:

1. Preparing the transaction file – Data is edited, validated, and sorted in the same sequence as the master file.
2. Updating the master file (subsidiary records) – Transaction file records are matched with master file records, and a new master file is created with updated data.
3. Updating the general ledger – Master file changes are updated in the general ledger. Documented with journal vouchers.
4. Preparing general ledger reports – Trial balances and other reports are produced.

Batch-Processing with Random-Access File Updating

Individual records are read one by one from the transaction file and used to update the related records in the master file. It is not necessary to:

1. Sort the transaction file in the same order as the master file.
2. Generate a new master file.

Paperless Processing Systems

- Characteristics of Paperless Processing Systems
- Batch Processing in Paperless Processing Systems
- Methods of Real-time Processing in Paperless Processing Systems
- Components of Real-Time Sales Systems

Characteristics of Paperless Processing Systems

Two primary characteristics include:

1. Either batch or real-time processing is possible.
2. With most types of on-line real-time processing, individual transactions are processed as they are input into the system. Under such a system master files are always up to date.

Batch Processing in Paperless Processing Systems

Batch processing in paperless processing systems is similar to batch processing in paper-based systems except that:

1. Journal vouchers are replaced by electronic equivalents.
2. Periodic batch runs automatically update the general ledger.

Methods of On-Line, Real-time Processing in Paperless Processing Systems

1. Inquiry/response systems – Users request information but do not input data for processing.
2. Data entry systems – Users interactively input data that is processed periodically in batches.
3. File processing systems – Users interactively input data that is immediately processed against relevant master files.
4. Full or transaction processing systems – Similar to file processing systems except that the entire transaction is completed when input. For example, an invoice is generated in addition to updating accounts receivable.

Components of Real-Time Sales Systems

- Point-of-Sale (POS) System – to allow data collection at the point where the sale is completed, the traditional cash register is enhanced to allow it to function as the source data-entry device for sales transactions.
- Bar-Coding Technology – scanner technology and machine-readable bar codes such as UPC.
- EDI Ordering System – allows direct computer-to-computer exchange of business information and documents for placing orders and facilitating quick shipment.

The Output System

The output system can be paper-based, paperless, or something in between. Most paper-based batch-oriented systems with sequential file processing produce very large volumes of output. In contrast, on-line, real-time paperless systems tend to produce very little output.

A. The Purpose of Output Controls

B. Examples of Output Controls

The Purpose of Output Controls

Output controls are designed to ensure that processing results in valid output and that outputs are properly distributed.

Examples of Output Controls

1. Control totals should be balanced to control totals generated independently of data processing.
2. Internal audit often sets up an EDP control group that makes sure that errors are corrected, and that controls are instituted to avoid repeat situations.
3. An output distribution register is maintained to control the distribution of reports.

Summary

- Paper-based input systems maintain the traditional audit trail.
- Paperless input systems lose some internal controls that can be accomplished through transaction logs and tagging.
- Paper-based processing systems are usually batch oriented.
- Paperless processing systems may be batch or real time.
- Output systems should ensure valid output and proper distribution of information.