CSC 222: Assembly Language Programming (2 Units) - Compulsory

Binary Number Systems and Other Systems. Types of encoding, modes of representations of data e.g integer, floating, package decimal, character e.t.c Basic Structure of the Computer. Instruction Set and Corresponding Machine Language modes of addressing. Instruction execution and flow of marcos, Linkages, interfacing, Assembling a Language Program with programs in the Other Languages, Necessary Aspect of Job control languages.

Assembly Programming Language. 25-01-2022

1) Binary Number Systems and Other System.

Radix is Also known as Base.

The Number of digits used in Binary system is based on Radix

$^{5}1\,^{4}1\,^{3}0\,^{2}1\,^{1}1\,^{0}0_2 = (1\times2^5)+(1\times2^4)+(0\times2^3)+(1\times2^2)+(1\times2^1)+(0\times2^0)$

$= (1\times32)+(1\times16)+(0\times8)+(1\times4)+(1\times2)+(0\times1)$

$= 32+16+0+4+2+0$

$= 54_{10}$

$^{3}1\,^{2}0\,^{1}0\,^{0}1\,^{-1}\cdot\,^{-2}0\,^{-3}1\,^{-4}0_2 = (1\times2^3)+(0\times2^2)+(0\times2^1)+(1\times2^0)\cdot(0\times2^{-1})+(1\times2^{-2})+(0\times2^{-3})+(1\times2^{-4})$

$= (1\times8)+(0\times4)+(0\times2)+(1\times1)\cdot(0\times0.5)+(1\times0.25)+(0\times0.125)+(1\times0.0625)$

$= 8+0+0+1.0+0.25+0+0.0625$

$= 9.3125_{10}$

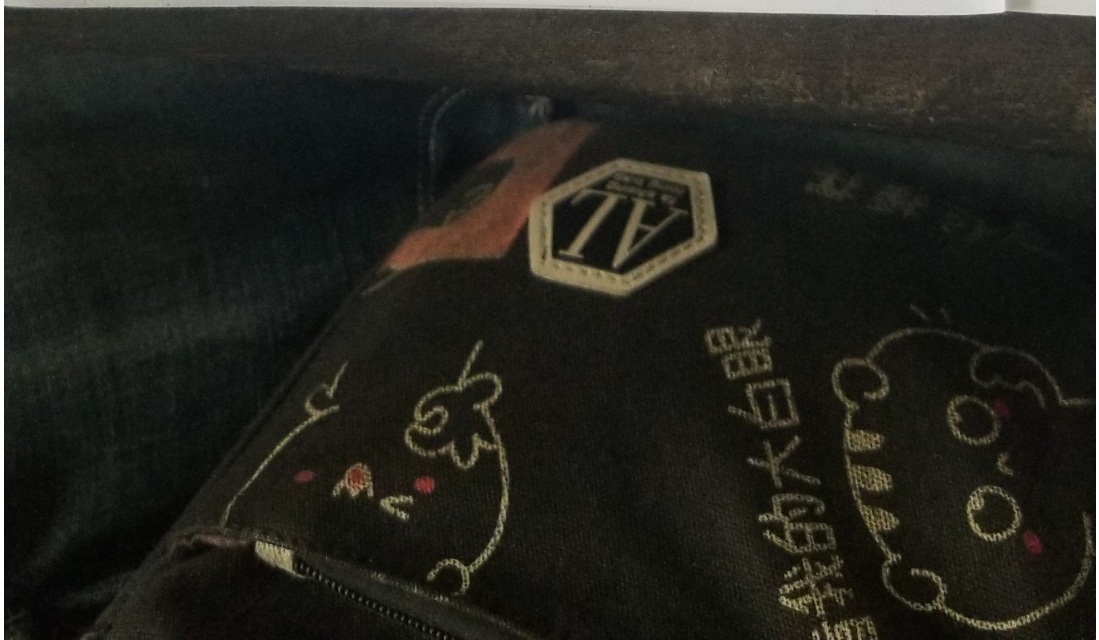
Convert from Octal to decimal

$(172)_8 = (1\times8^2)+(7\times8^1)+(2\times8^0)$

$= (1\times64)+(7\times8)+(2\times1)$

$= 64+56+2$

$= 122_{10}$



$(\overset{2}{1}\overset{1}{3}\overset{0}{7}.\overset{-1}{2}\overset{-2}{1})_8 = (1\times8^2)+(3\times8^1)+(7\times8^0)\cdot(2\times8^{-1})+(1\times8^{-2})$

$= (1\times64)+(3\times8)+(7\times1)\cdot(2\times0.125)+(1\times0.015625)$

$= 64+24+7\cdot0.25+0.015625$

$= 95\cdot0.265625_{10}$

Convert from Hexadecimal to Decimal.

$(1AF)_{16}$

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A(10), B(11), C(12), D(13), E(14), F(15)

$\overset{2}{1}\overset{1}{A}\overset{0}{F}_{16} = (1\times16^2)+(10\times16^1)+(15\times16^0)$

$= (1\times256)+(10\times16)+(15\times1)$

$= 256+160+15$

$= 431_{10}$

$\overset{2}{1}\overset{1}{E}\overset{0}{0}.\overset{-1}{2}\overset{-2}{A}_{16}$

$= (1\times16^2)+(14\times16^1)+(0\times16^0)+(2\times16^{-1})+(10\times16^{-2})$

$= (1\times256)+(14\times16)+(0\times1)+(2\times0.0625)+(10\times3.90625\times10^{-3})$

$= (256)+(224)+(0)+(0.125)+(0.0390625)$

$= 480\cdot0.1640625_{10}$

0.375_{10} to binary

	fractional part	integer part
$0.375 \times 2 =$	0.75	0
$0.75 \times 2 =$	0.5	1
$0.5 \times 2 =$	0.0	1

count downwards ↓

0.011_2

0.011_2

Decimal to Octal 266_{10}

8 |	266
8 |	33 r 2
8 |	4 r 1
	0 r 4

count upwards ↑

$= 412_8$.

Decimal to Hexadecimal 423_{10}

= 16 |	423
16 |	26 r 7
	1 r 10 = A
	10 r 1

$1A7_{16}$

172_8 to binary

001 001 010

Encoding of Machine Instructions 01-02-2022

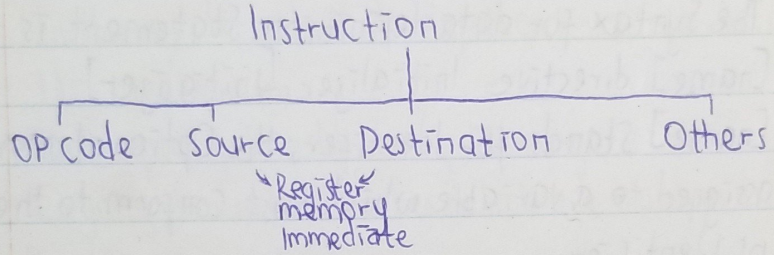
Instructions specifies the actions that must be performed by the processor circuitory to carry out desired task.

Example: Add R1, R2

↓

op code.

Instructions are Majorly divided into



Data Definition in Assembly Language

Assembly Language defines intrisic data type each of which describes a set of values that can be assigned Variables and expression of given types.

The Essential Characteristics of each data type is its size in bit which ranges from :- 8 bits, 16, 32, 48, 64, 80.

Other Characteristic of data type :- - Signed
- Unsigned

D - double S - SIGN

Eg: a Variable declared are D WORD holds Unsigned 32 bits.

Data Defrnition Statement.

A data Definition Statement sets aside Storage in Memory for a variable with an Optional Name.

Data definition Statement Creates variable based on Intrisic data type Such as BYTE, SBYTE, WORD, SWORD, DWORD.

The Syntax for data definition Statement is

[name] directive Initializer [Initializer]

[name] Stands for Identifer, the Optional name assigned to a variable which must conform to the rules of Identifier.

Directive in a data definition Statement can be BYTE, SBYTE WORD, SWORD, OR DWORD.

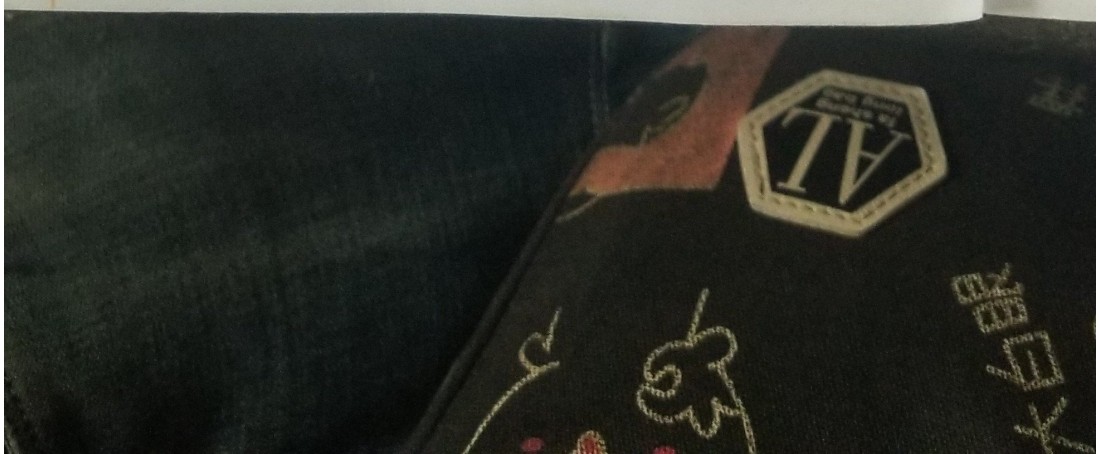
Initializer: at Least One Initializer is required in a data definition and we can have more than One Initializer if required.

Eg Byte : 41 h

SByte : -128

SByte : +127

DWORD : 4000H.



Basic Structure of the Computer.

Computer Organization refers to Operational units and their Interconnection that realize the Architectural Specification.

Examples of Architectural Attribute.

- Instruction Set.
- Number of bits Used to Represent various data type
- I/O Mechanism.
- Techniques for addressing Memory.

Organizational Attributes which are transparent to Programmers are

- Control Signal
- Interfaces between Computer peripherals and Memory tech

Structure And function of Computer.

- Structure is the way in which Components are Interelated (i.e Relationship between Components)

Function is the Operation of individual Component in the Structure.

They can be divided into four (4) function

- Data Processing.

function and Instruction types is the same Way.

Data Storage: We can store our data in the Main memory.
Data Movement
Data Control :- control the way data are processed.

Considering the Structural Components four main Structural Part of the System Are:

STRUCTURE

→ The Central Processing → ALU, → control Units, ↓ Registers
- Main Memory.
- I/O devices: We get our data through I/O devices whether through a devices byte.
- System Interconnection.

Assembly Programming Language. 08/02/2022

Assembly Language is a type of Low-Level computer programming Language consisting mostly of Symbolic equivalents of a particular computers machine language.

Assembler

An Assembler is a mnemonics (i.e SUB, ADD). Assembler Do allow the Use of Symbolic names for constant and Re

Labels (Variable) to Create Instruction and Memory addresses
When a program in assembly language is fed into a program called an assembler, the assembler Convert the program into a binary Program Suitable for actual execution.

PROCESSOR CYCLE.

The execution of a Single Instruction on an assembler can be divided into the following Steps

- Fetch instruction from the memory i.e Code Segment Using Program Counter.
- Increment the Program Counter.
- Decode fetched Instruction.
- Fetch the Necessary Data from Memory or processor register
- Perform the Instruction
- Store the result of the Instruction in Memory or Register
- Go back to next Instruction from the Code Segment in the program Counter.

TYPES OF REGISTER AND DEFINITION OF REGISTERS.

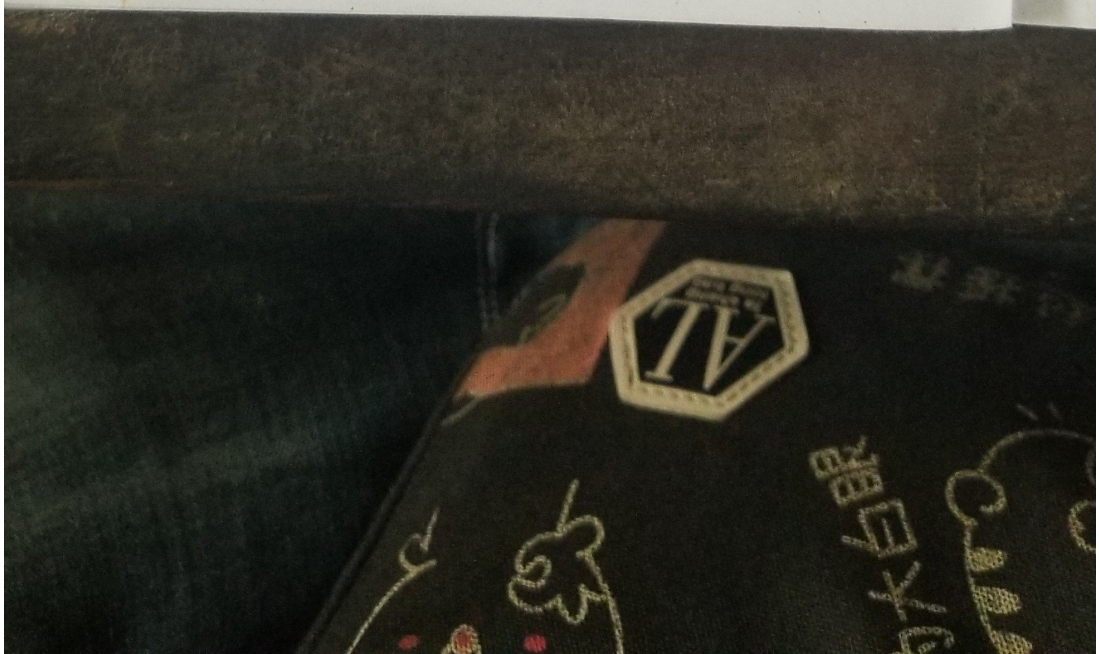
General Registers: It is made up of Ax, Bx, Cx, Dx.

Data Registers.

Ax - Accumulator Register: It is used to collect the result of Computation and it is the target of many instruction.

- Bx - Base Register : It is used for most of the things that Ax can do but it has One power Ax does not have which is the Ability to store memory Address and then execute an Instruction whose Operand Comes from memory address Contained Bx. Therefore Bx can hold a pointer to Memory while Ax cannot
- Cx - Counter Register : It is used for Counters for loops It documents
- Dx - Data Register : It is used together with Ax with Dword (Double word) Length instruction. When Dx contains the Higher Order 16 bits and Ax contains Lower Order 16 bits.

2. Pointer Register
   - Stack pointer : It holds context Information about a running Program. It is denoted by SP.
   - Base Pointer : It contains an address in the stack
   - SP Point to the top of the stack. BP Point to any Location ~~with~~ within the Stack.
   - Index Register : Combination of Source Index and destination. They are used together with base Index. Pointer to address data in the stack.



Program Counter: It holds the Next Instruction to be executed.

Flag Register / condition code Register: It is a Single bit Register Set by Arithmetic instruction that relate to the following Result. It hold flag for Instance Zero flag denoted by Z if the Sign 0 is negative it will be 1 if positive it will be 0. If there's a borrow from bit 7 to 6 then there is an Overflow and it will be 1

Carry flag is represented by C.

Auxillary flag: If there is no borrow between the bits it gives 1

P Parity flag: If there is even number of 1 then it gives 1 Otherwise 0.

Format of Writing Instruction.

Many Instruction Contain two Operand i.e Source Operand and Destinaton Operand

TYPES OF ADDRESSING:

- Data Segment Addressing: It contain address moves for the data Segment. Addresses of this type Always contain a pair of parenthesis to Indicate that the content of the address Instead of the value is meant.
- Direct Addressing: It is a data addressing in which data

address of the Operand is in the Instruction Itself.
Register Indirect Addressing:- The address of the Operand is stored in One of the Register.

- Immediate Addressing:- In which the constant byte or word value in the Instruction Itself.
- Implied Addressing:- In this case the Operand are Implicit in the Instruction itself.

Push Ax i.e Add the Content of Ax.

Instruction Set:

It is the heart of every computer. To Understand a Computer it is neccessary to have a good Understanding of its Instruction set.

MOV

It is an Instruction used to copy and Move data.

XCHG

It exchange data or the Content of Register.

LEA

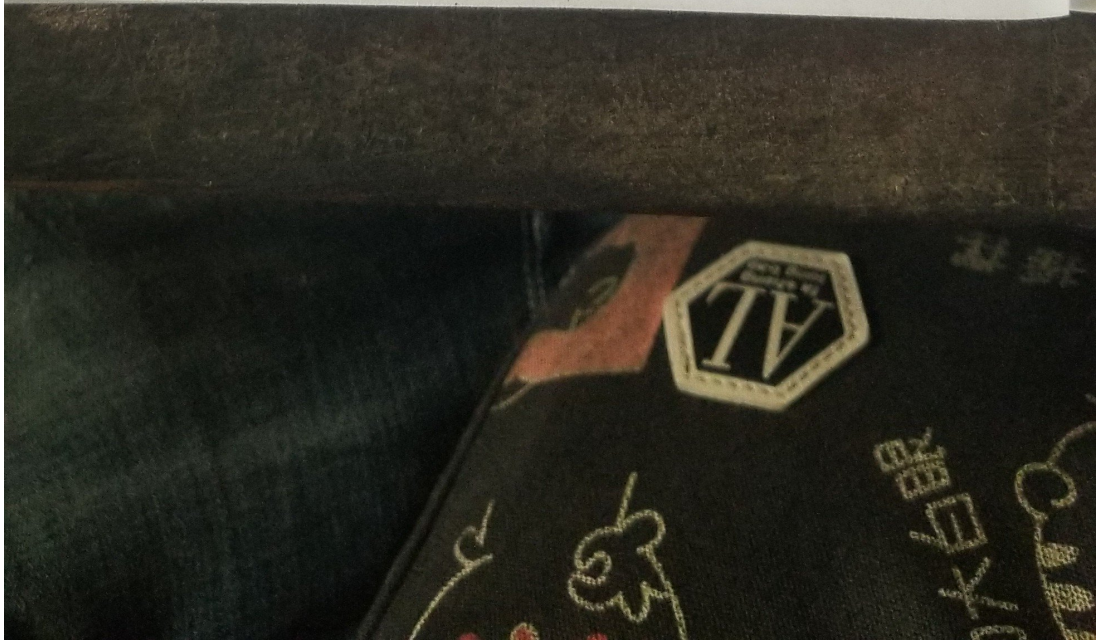
It is an Instruction used to Lead Effective Address.

PUSH

POP

PUSH F:- A PUSH Instruction with flag Register

POP F:- A POP Instruction with flag Register



- Arithmetic Instruction Sets: ADD SUB, MUL, DIV
  Others: JUMP
- Logical Instructron Sets: AND, OR, XOR.

Define Addressing in Assembly Programming Language

PUSHFD: Instruction pushes the 32-bit EFLAGS Register on the stack
POPFD: POP the stack into EFLAGS.

In the early days there are various processor

Assembler: It is a Utility program that convert Source Code Assembly Language to Low-Level Language.

page3 * Differentiate Between a Linker and Debugger.
page4 * How Does Assembly Language Relate to Machine Language.
to6 * Intel processor are x86 architecture.
* How Does C# and Java Relate to Assembly Language.
* is Assembly Language Portable?
* Why Learn Assembly Language?
* Are there Rules In Assembly Language?
* Assembly Language Application

Comparison of Assembly Language to High Level Language.
Virtual Machine Concepts.
Data Representation.
Binary Integers.
How many bytes are Contained in each of the following data types

a) word
b) Doubleword
c) Quadword
d) Double Quad word.

Page 7 for the Assignment.
Basic Language Elements.

An integer literal(also known as an integer constant) is made up of an Optional Leading Sign, one or more digits and an Optional radix character that indicates the number's base

Precedence
(), *, /, MOD, +, -
Operator precedence refers to the Implied order of Operations when an expressions contains two or more Operators.