

# Lab #2 CoSc 20203 Fall 2018

## M68000 Machine Instructions

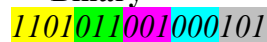
### Due: Tuesday Oct 2<sup>nd</sup> Midnight

#### DESCRIPTION

The Motorola 68000 family of CPUs is a Complex Instruction Set Computer (CISC); i.e., there are many formats, lengths, etc. We will consider only the four main instructions with a single addressing mode. An instruction in assembly language is written like

**ADD.W D5,D3 ;add what's in D5 to D3**

That instruction tells the CPU to add a “word” (16-bit) value from data register 3 to the value in data register 5. In machine language, the instruction looks like

Binary	Hex
	D645

The format of the instructions that we will consider is

Assembly	Language	Machine Language
ADD.s	Dx,Dy	1101yyysss000xxx
SUB.s	Dx,Dy	1001yyysss000xxx
MULS.s	Dx,Dy	1100yyy111000xxx
DIVS.s	Dx,Dy	1000yyy111000xxx

where “s” is the size of the data

000 B Byte 8 bits

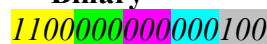
001 W Word 16 bits

010 L Long 32 bits (note the difference in terminology)

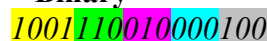
and “x” and “y” are register numbers (0–7).

Other examples are

**MULS.B D4, D0 ;multiply in D4 by content in D0**

Binary	Hex
	C004

**SUB.L D4, D6 ;subtract D4 from D6**

Binary	Hex
	9C84

*Sources:*

<https://simpledevcode.wordpress.com/2016/12/15/mini-guide-to-68000-assembly-programming/>

[https://en.wikipedia.org/wiki/Motorola\\_68000](https://en.wikipedia.org/wiki/Motorola_68000)

## ASSIGNMENT

Write a Java application with a graphical user interface that allows a user to encode and decode **M68000** instructions. The program must provide the following functionalities

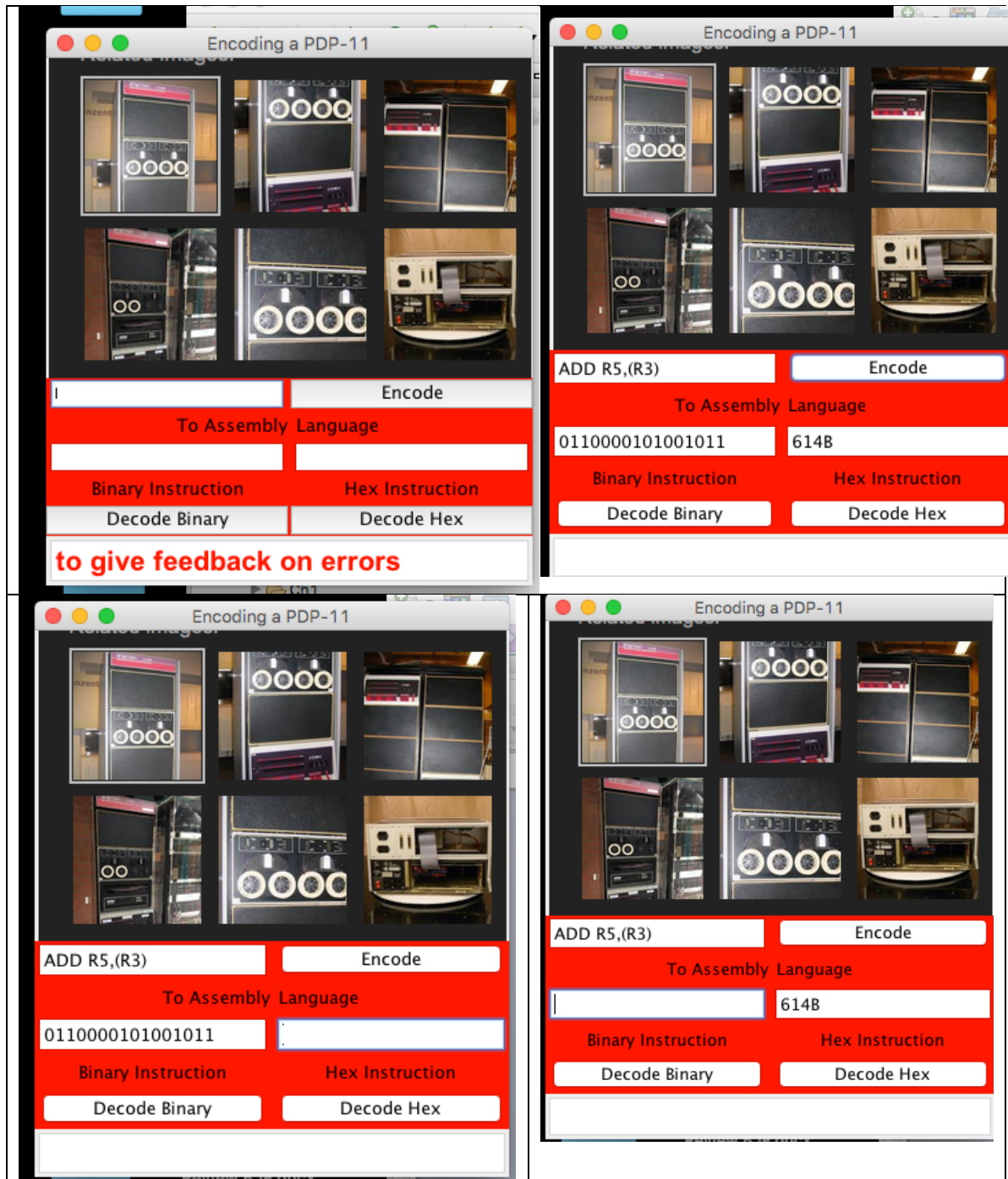
- Conversions should be done from Assembler Mnemonics to Assembler Code and vice versa.
- Upper and lowercase are accepted and represent the same instruction
- Generate the message of errors if the input cannot be translated (i.e. encoded/decoded)
- Display JTextFields with the instructions encoded in both binary and hexadecimal during session should be provided
- Exception handling for checked and unchecked exceptions must be provided
- The facility to read/save text files using either FileDialog or JFileChooser should be provided
- A menu bar option as well as a help option may added as an extra feature

### (BONUS)


The design of the interface is left to you. A Zip file with this requirements (i.e. screenshots, manual, jar, javadoc, src) should be submitted using D2L. You should use appropriate widgets to allow the user to enter a **four digit hex** number and request that it be decoded into assembly language or enter an assembly(in **binary**) language instruction and request that it be encoded into hex representation. Your program needs to encode/decode only the four instructions shown above.

Here is an initial visual guide on the final product. Note that your task is only the software translation (**i.e. encoding/decoding**) and not the hardware execution of the instructions. This is a string processing application, so in order to complete this project you must use the **StringTokenizer** class, substrings and testing characters in a **String** as we have seen in in class. And do remember to **document** each method and generate a **clickable jar** that should be submitted along with the **java sources** and the **javadoc**. **Exceptions** must be caught using **try/catch** in the model or in the control with the corresponding **throws** in the model method.

**IMPORTANT NOTE** The example presented here is for the encoding of another machine the **pdp11**, but the functionality and performance are the similar



Encoding a PDP-11



ADD R3,D4

Encode

To Assembly Language

Binary Instruction

Hex Instruction

Decode Binary

Decode Hex

Illegal register specification