International Journal of Electrical Engineering and Technology.
ISSN 2249-3085 Volume 9, Number 2 (2019), pp. 121-134

http://www.ripublication.com

Different Approaches to Writing Calculation Programmes with the 68K Motorola Microprocessor Assembly Language and with High Level Language

Sajid M. Sheikh[1], Thato Kabo Chimidza[2], Olebogeng Bulawayo[3]

Faculty of Engineering and Technology, University of Botswana, Gaborone, Botswana.

Abstract

It is realized that there are not sufficient calculation programme examples and tutorials available for students and thus students fail to gain confidence in writing their own calculation programmes with the 68K assembly language. This serves as a tutorial paper with examples on writing calculation programmes with both assembly languages and High level languages. Knowledge of both assembly language and High Level Languages is vital when teaching microprocessor based system courses.

Keywords — 68K Microprocessor, Assembly directives, High Level Languages, Registers

I. INTRODUCTION

In a Microprocessor Based System Course in an Electronic or Computer Engineering Degree programme in University, assembly languages such as that for the 68K microprocessor are still taught. Such courses also teach High Level Languages (HLL) such as C++ and python that are used by the Arduino Microcontroller and the Raspberry Pi Microcontroller. Such courses therefore, equip students with the use of both assembly languages and HLL.

It is realized that students do not have sufficient calculation programme examples and thus fail to gain confidence in writing their own calculation programmes with the 68K assembly language. Students always find it a challenge to grasp the art of writing 68k calculation programmes. Writing assembly language code, one has to be well aware of the addressing modes, registers and hardware that make up the microprocessor. The

68000's instruction set reference can be found in [1]–[4]. Other 68k processor technical details can also be found in [3], [4].

In this tutorial paper we give examples of calculation programmes done both in assembly languages and High level languages in order to help students master how to write calculation programmes that require user input with the 68K. Simple programmes for calculating Power and Efficiency are presented in this paper done in both C++ and 68k assembly language. A code quick reference to writing C++ code and 68K Assembly code to get input from a user and present output to the user can be found in [2]. However, there are limited examples with the 68k that prompt users to end values and then output the results for the calculations performed. This tutorial aims a making users conversant with using the TRAP # 15 statements with the 68k to get input from the users and display output to the users. The same programmes are also written in C++ so users can make a comparison.

Even though HLL are more often used now, assembly language still has its advantages over high level language. It allows one to write code instructions directly based on the microprocessor and its hardware. It gives the programmer a clear feel on the execution of such programmes and their performance. It is therefore, very important to understand the internal architecture of a microprocessor to write assembly code, while for HLL, no internal architecture knowledge is required.

In section 2 of this paper we present 2 calculation formulas for which programmes are written in section 3. To prompt and get user input or display output with the 68K microporocessor, the TRAP # 15 statements are used, while with the C++ HLL *cin* and *cout* commands are used as shown in table 1.

Table 1: C++ and 68K input and output statements references [2].

C++	68k Assembly Language
cin >> Integer Allow the user to enter a number	**MOVE.B #4,D0** **TRAP #15** Takes a user-entered number and stores in D1.L
cin >> char Allow the user to enter a character	**MOVE.B #5,D0** **TRAP #15** Takes a user-entered character and stores in D1.B
cin >> string Allow the user to enter a string	**MOVE.B #2,D0** **TRAP #15** Stores a user-entered, null-terminated string at address A1. Length of string is stored in D1.W
cout << * Displays variable or characters/strings	**String:** **MOVE.B #13,D0 (Adds a new line)** **OR** **MOVE.B #14,D0(Doesn't start new line)** **TRAP #15** Displays string at address A1 with or without carriage return, line-feed **Number:** **MOVE.B #3,D0** **TRAP #15** Displays number located in position D1.L **Character:** **MOVE.B #6,D0** **TRAP #15** Displays what's in D1.B as ASCII character

II. METHODOLOGY

In this paper we present programme examples written with different approaches for the below formulas:

1. Electric Power= Voltage x Current
2. Efficiency ($\boldsymbol{\eta}$) = ($P_{out}./P_{in}$) * 100%

III. PROGRAMMES

Two programs for Electric Power and Efficiency are written with both the 68K and C++ to present different approaches. Each of the codes provided are commented with explanations provided to understand the code.

Assembly Language 68K Programme for Electric Power

APPROACH 1

```
*-----------------------------------------------------------
* Program    : Electric Power
* Description: Calculating Electrical power using formula Electric power=Voltage*current.
* The program prompts user to enter voltage and current then calculates the power
*-----------------------------------------------------------

PRTSTRCRLF     EQU    0          ;Trap function used to display characters of string with carriage return and line feedback
PRTSTR  EQU    1                  ;Trap function used to display characters without carriage return and line feedback
NUMIN   EQU    4                   ;Trap function used to read keyboard
NUMOUT  EQU    3                   ;Trap function used to display or output decimal numbers

ORG    $1000    ;allocate space for program start

*     Prompt for  voltage to be entered
      MOVEA.L #PROMPT1,A1                    ;Move prompt1 message declared below as DC.B to address register A1
      MOVE.W  #(Prompt2-Prompt1), D1         ;sequence to display Prompt1 message first then begin next line and display Prompt2
message
      MOVE.B  #PRTSTR,D0                     ;Set up the TRAP function 1 to print string(words)
      TRAP    #15                             ; convert D0.B to D1.B display unsigned number it
```

```
*     Get the Voltage from the keyboard and store it
      MOVE.B  #NUMIN,D0                    ;Set up trap function 4 to get unsigned number from the keyboard and store it in Data
register D0.B
      TRAP    #15                          ; display unsigned number entered
      MOVE.W  D1,voltage                   ;move contents of data register D1.W to memory storage 'voltage' declared below as DS.W

*     Prompt for Current to be entered
      MOVEA.W #PROMPT2,A1                  ;Move prompt2 message declared below as DC.B to address register A1.W
      MOVE.W  #(Prompt3-Prompt2),D1        ;sequence to display prompt2 message first then begin next line and display Prompt3
message
      MOVE.B  #PRTSTR,D0                   ;Set up the TRAP to print string
      TRAP    #15                          ; convert D0.B TO D1.B and display it

*     Get the current from the keyboard and store it
      MOVE.B  #NUMIN,D0                    ;Set up trap to get unsigned number from keyboard and store it in Data register D0.B
      TRAP    #15                          ;read entered unsigned number in D0.B and store it in D1.B
      MOVE.W  D1,current                   ;move contents of data register to memory storage 'current' declared below as DS.W

*     Work out the Electric power
      MOVE.W  voltage,D2                   ;move contents of voltage to data register D2.W
      MULU.W   current,D2                  ;multiply contents of current with contents of data register D2.W
      MOVE.W  D2,Electric_power            ;Move contents of D2.W to Electric power

*     Display the Electric Power
      MOVEA.L #Prompt3,A1                  ;Move prompt3 message declared below as DC.B to address register A1
      MOVE.W #(Prompt3-Prompt2),D1         ;sequence to display Prompt2 message first then begin next line and display Prompt3
message
      MOVE.B  #PRTSTR,D0                   ;Set up output string trap
      TRAP    #15                          ; setup and display unsigned number in data register D1
      MOVE.W  D2,D1              ;move contents of data register D2.W to D1.W
      MOVE.B  #NUMOUT,D0                   ;Set up trap to output unsigned number
      TRAP    #15                          ; convert D0.B contents to D1.B then Print it

      STOP    #$2700             ; halt program

Prompt1 DC.B    'Please enter Voltage: '    ; declaring word constant to ask user to enter voltage
Prompt2 DC.B    'Please enter Current: '    ; declaring word constant to ask user to enter current
Prompt3 DC.B    'Electric power =:'         ; declaring word constant to display after calculating Electric power
```

```
voltage    DS.W   1                    ;declare space,1st byte to store 'voltage'
current    DS.W   1                    ;declare space, 2nd byte to store 'current'
Electric power  DS.W   1               ; declare space, 3rd byte to store ‘electric_power'

     END    $1000                      ;exit program
```

APPROACH 2

```
*The code prompts the user to enter the values of voltage in volts
*and current in amps and from there it calculates the product of the
*two variables giving a parameter known as power.
*      Main Program
START          ORG $1000                          ; Starts program at $1000
               LEA          VOLTAGEPROMPT,A1      ; LOADS VOLTAGEPROMPT INTO ADDRESS REGISTER A1
               MOVE.B       #14,D0                ; Copies the number 14 to D0 for trap task use

*1. Trap task #14                                 ; displays the text loaded into Address register A1
               TRAP         #15

*2. Trap #15 is used to perform input/output operations of the task number loaded *into data register d0
               MOVE.B       #4,D0                 ;Copies the number 4 in D0 for trap task use
*3. Trap task #4 is used to read input number from the keyboard and copy the number *to D1.L
               TRAP         #15                   ; Refers to 2
               MOVE.B       D1,VOLTAGE            ;stores the value of input voltage to location named *VOLTAGE
               MOVE.B       VOLTAGE,D2            ;copies the stored voltage value to D2
               LEA          CURRENTPROMPT,A1      ;LOADS CURRENTPROMPT INTO ADDRESS REGISTER A1
               MOVE.B       #14,D0                ;Copies the number 14 to D0 for trap task use
               TRAP         #15                   ; Refers to 2
               MOVE.B       #4,D0                 ;Copies the  4 number to D0 for trap task use
               TRAP         #15                   ; Refers to 2
               MOVE.B       D1,CURRENT            ; stores the value of current in location named *CURRENT
               MOVE.B       CURRENT,D3            ; copies the stored current value to D3
               MULU         D2,D3                 ; performs multiplication of current and voltage and *stores product in D3
               MOVE.B       D3,POWER              ; copies value of product(the power) to location named *POWER
               MOVE.B  POWER,D1                   ; Copies value of power into D1 for output purposes
               LEA          POWERRESULT,A1        ; LOADS POWERRESULT INTO ADDRESS REGISTER A1
               MOVE.B       #14,D0                ; Copies the number 14 to D0 for trap task use
               TRAP         #15                   ; Refers to 2
               MOVE.B       #3,D0                 ; Copies the number 3 to D0 for trap task use
```

```
*4. Trap task #3 displays the number loaded in D1.L in decimal number type
                TRAP        #15                 ; Refers to 2
                LEA         POWERUNITS,A1       ; LOADS SIGN INTO ADDRESS REGISTER A1
                MOVE.B      #14,D0              ; Copies the number 14 to D0 for trap task use
                TRAP        #15                 ; Refers to 2
                MOVE.B      #4,D0               ; Copies the number 4 to D0 for trap task use
                TRAP        #15                 ; Refers to 2
VOLTAGE         DS.B        10                  ; Reserves a space of 10 bytes for the voltage
CURRENT         DS.B        10                  ; Reserves a space of 10 bytes for the current
POWER           DS.B        10                  ; Reserves a space of 10 bytes for the power
VOLTAGEPROMPT               DC.B                ; 'ENTER VOLTAGE IN VOLTS ',0 TEXT TO BE DISPLAYED FOR *VOLTAGE
CURRENTPROMPT               DC.B                ; 'ENTER CURRENT IN AMPS ',0 TEXT TO BE DISPLAYED FOR *CURRENT
POWERRESULT  DC.B           'POWER = ',0        ; Text to indicate the power
POWERUNITS   DC.B           'W',0               ; Units of power

    END    START                                ; Terminates the program
```

Assembly Language 68K Programme for Efficiency

APPROACH 1

```
*-----------------------------------------------------------
* Program: Efficiency
* Description: Program prompts user to enter power out and power in values then calculates the efficiency using the equation;
Efficiency=Pout/Pin X100%
*-----------------------------------------------------------

PRTSTRCRLF     EQU    0                    ; TRAP function to print string with carriage return and line feed back
PRTSTR  EQU    1                            ; TRAP function to print character string without carriage return and line feedback
NUMIN   EQU    4                            ; TRAP function to read number from keyboard
NUMOUT  EQU    3                            ;TRAP function to output signed (decimal) numbers

ORG    $1000
*      Prompt for power out value
    MOVEA.L #Prompt1,A1                     ;Move prompt1 message declared as DC.B below to address register A1
    MOVE.W  #(Prompt2-Prompt1),D1           ;sequence to display/call Prompt1 message first then begin next line and display Prompt2
    MOVE.B  #PRTSTR,D0                      ;Set up the TRAP 1 to print string
    TRAP    #15                             ; convert D0 contents to D1 and display D1 contents
```

```
*     Get the power out value from the keyboard and store it
      MOVE.W  #NUMIN,D0                ;Set up trap function 4 to input unsigned number from keyboard
      TRAP    #15                      ; convert contents of D0 to D1 then read the D1 contents
      MOVE.W  D1, Pout                 ;move contents of data register D1 to memory space 'Pout'

*     Prompt for power in value
      MOVEA.L #Prompt2,A1              ;move prompt2 message to address register A1
      MOVE.W  #(Prompt3-Prompt2),D1    ;sequence to display/call prompt2 message first then begin next line and display Prompt3
      MOVE.W  #PRTSTR,D0               ;Set up the TRAP 1 to print string
      TRAP    #15                      ; convert D0 contents to D1 and display the D1 contents

*     Get the power in value from the keyboard and store it
      MOVE.W  #NUMIN,D0                ;Set up trap 4 to input unsigned number from keyboard
      TRAP    #15                      ; convert D0 contents to D1 and display it
      MOVE.W  D1,Pin                   ;Move contents of D1 to memory space 'Pin'

*     Work out the Efficiency
      MOVE.W  Pout,D2                  ; move contents of memory space Pout to Data register D2.W
      MULU.W #100,D2                   ; Multiply 100 with contents of D2.W and store in D2.W
      MOVE.W Pin,D3                    ; Move contents of memory space 'Pin' to data register D3.W
      DIVU.W  D3,D2                    ; Divide contents of D2 by contents of D3 and store result in D2.W
      MOVE.W  D2,Efficiency            ;Move contents of D2 to memory space 'Efficiency'

*     Display the Efficiency
      MOVEA.L #Prompt3,A1              ; Move Prompt3 message declared below as DC.B to address register A1
      MOVE.W  #(Prompt3-Prompt2),D1    ; sequence to display Prompt2 message first then begin next line and display Prompt3
                                       message stored in D1
      MOVE.W  #PRTSTR,D0               ; Set up output string trap
      TRAP    #15                      ; Print it
      MOVE.L  D2,D1                    ; move contents of D2.L to D1.L
      MOVE.W  #NUMOUT,D0               ; Set up trap to output unsigned number stored in D0.W
      TRAP    #15                      ; convert D0.W contents to D1.W and Print it

      STOP    #$2700
Prompt1 DC.B    'Please enter power out value: '  ; declaring word constant to ask user to enter 'power out'
Prompt2 DC.B    'Please enter power in value: '   ; declaring word constant to ask user to enter 'power in'
Prompt3 DC.B    'Efficiency= '                    ; declaring word constant to be displayed after calculating efficiency

Pout    DS.L    1                        ; declaring data space to store 'power out' value
Pin     DS.L    1                        ; declaring data space to store 'power in' value
```

```
Efficiency   DS.L   1                    ; declaring data space to store 'Efficiency'

     END    $1000                        ; exit program
```

APPROACH 2

```
*The code prompts the user to enter the values of input power in watts
*and output power in watts and from there it calculates the efficiency .
*      Main Program
START          ORG $1000                          ; Starts program at $1000
               LEA          PINPROMPT,A1          ; LOADS PINPROMPT INTO ADDRESS REGISTER A1
               MOVE.B       #14,D0                ; Copies the number 14 to D0 for trap task use

*1. Trap task #14 diplays the text loaded into Address register A1
               TRAP         #15

*2. Trap #15 is used to perform input/output operations of the task number loaded *into data register D0
               MOVE.B       #4,D0                 ; Copies the number 4 in D0 for trap task use

*3. Trap task #4 is used to read input number from the keyboard and copy the number *to D1.L
               TRAP         #15                   ; Refers to 2
               MOVE.B       D1,PIN                ; stores the value of input power to location named PIN
               MOVE.B       PIN,D2                ; copies the stored PIN value to D2
               LEA          POUTPROMPT,A1         ; LOADS POUTPROMPT INTO ADDRESS REGISTER A1
               MOVE.B       #14,D0                ; Copies the number 14 to D0 for trap task use
               TRAP         #15                   ; Refers to 2
               MOVE.B       #4,D0                 ; Copies the number 4 in D0 for trap task use
               TRAP         #15                   ; Refers to 2
               MOVE.B       D1,POUT               ; stores the value of output power in location named POUT
               MOVE.B       POUT,D3               ; copies the stored POUT value to D3
               MULU         #100,D3               ; MULTIPLIES CONTENTS OF D3(POUT) BY 100
               DIVU         D2,D3                 ; divides Pout times 100 by Pin
               MOVE.B       D3,EFF                ; Copies quotient(efficiency) to location named EFF
               MOVE.B       EFF,D1                ; Copies value of efficiency into D1 for output purposes
               LEA          EFFRESULT,A1          ; LOADS EFF INTO ADDRESS REGISTER A1
               MOVE.B       #14,D0                ; Copies the number 14 to D0 for trap task use
               TRAP         #15                   ; Refer to 2
               MOVE.B       #3,D0                 ; Copies the number 3 to D0 for trap task use
```

```
*4. Trap task #3 displays the number loaded in D1.L in decimal number type
                TRAP            #15                             ; Refers to 2
                LEA             EFFICIENCYSIGN, A1              ; LOADS SIGN INTO ADDRESS REGISTER A1
                MOVE.B          #14,D0                          ; Copies the number 14 to D0 for trap task use
                TRAP            #15                             ; Refers to 2
                MOVE.B          #4,D0                           ; Stores the number 4 in data register D0
                TRAP            #15                             ; Refers to 2
PIN             DS.B            10                              ; Reserves a space of 10 bytes for the POWER INPUT
POUT            DS.B            10                              ; Reserves a space of 10 bytes for the POWER OUTPUT
EFF             DS.B            10                              ; Reserves a space of 10 bytes for the EFFICIENCY
PINPROMPT       DC.B            'Enter input power in watts ',0 ; TEXT TO BE DISPLAYED *FOR Pin PROMPT
POUTPROMPT      DC.B            'Enter output power in watts ',0 ; TEXT TO BE DISPLAYED *FOR Pout PROMPT
EFFRESULT       DC.B            'Efficiency = ',0               ; TEXT TO BE DISPLAYED TO INDICATE *EFFICIENCY VALUE

EFFICIENCYSIGN  DC.B            '%',0                           ;SYMBOL FOR EFFICIENCY
    END    START                                                ;Terminates the program
```

HLL C++ Programme for Electric Power

APPROACH 1

```
//C++ program to calculate Electric-power using formula P=VI

#include <iostream>
using namespace std;

int main()
{
   double Voltage;                                  //declaring the voltage as a decimal value
   double Current;                                  // declaring current as a decimal value
   double Electric_power;                           // declaring Electric power as a decimal value

   cout<<"Enter Voltage and Current\n";             // output message to enter voltage and current value
   cin>>Voltage;                                    // input voltage value
   cin>>Current;                                     // input current value

   Electric_power=Voltage*Current;                  // process calculation of Electric_power
   cout<<"Electric_power="<<Electric_power<<"W";    //output Electric_power in watts
}
```

APPROACH 2

```
/*
The code prompts the user to enter the values of voltage in volts
and current in amps and from there it calculates the product of the
two variables giving a parameter known as power.
 */
 #include <iostream>
#include <string>
using namespace std;
//variable declarations
//output messages
string voltageprompt= "Enter the voltage in volts";
string currentprompt= "Enter the current in amps";
string powerresult= "power =";
string powerunits= "W";

//variables for calculation of power

float voltage, current, power;

int main()                                          // main functions begins
{
cout<< voltageprompt<<endl;                         //displays voltage prompt message
cin>>voltage;                                       //reads value of voltage input from keyboard
cout<<currentprompt<<endl;                          // displays current prompt message
cin>>current;                                       //reads value of current input from keyboard
power= current*voltage;                             //calculating power
cout<<powerresult<< power<< powerunits <<endl;      //outputs value of power in watts
}
```

HLL C++ Programme for Efficiency

APPROACH 1

```
#include <iostream>
using namespace std;
```

```
int main()
{
   double power_in;                          //declaring power in as a decimal value
   double power_out;                         //declaring power out as a decimal value

   double x;                                  //declaring a x as a decimal to use in Efficiency formula
   double Efficiency;

   cout<<"Enter power_out and power_in\n";   //output prompt to request user to enter power out and power in
   cin>>power_out;                            //input power out value
   cin>>power_in;                            //input power in value

   x=(power_out/power_in);                   //calculate power out divided by power in entered and store in x
   Efficiency=x*100;                          // calculate Efficiency by multiplying x times 100
   cout<<Efficiency<<"%";                     // output Efficiency
}
```

APPROACH 2

```
/*
The code prompts the user to enter the values of input power in watts
and output power in watts and from there it calculates the efficiency .
 */

 #include <iostream>
#include <string>

using namespace std;

//variable declarations
//output messages
string pinprompt= "Enter input power in watts";
string poutprompt= "Enter output power in watts";
string effresult= "Efficiency =";
string  efficiencysign = "%";

//variables for calculation of efficiency
int pin , pout, eff;
```

```
int main()                                          // main functions begins
{
cout<< pinprompt<<endl;                             //displays voltage prompt message
cin>>pin;                                           //reads value of voltage input from keyboard
cout<<poutprompt<<endl;                             // displays current prompt message
cin>>pout;                                          //reads value of current input from keyboard
eff= (100*pout)/pin;                                //calculating power
cout<<effresult<< eff<< efficiencysign <<endl;      //outputs value of power in watts
}
```

IV. DISCUSSION

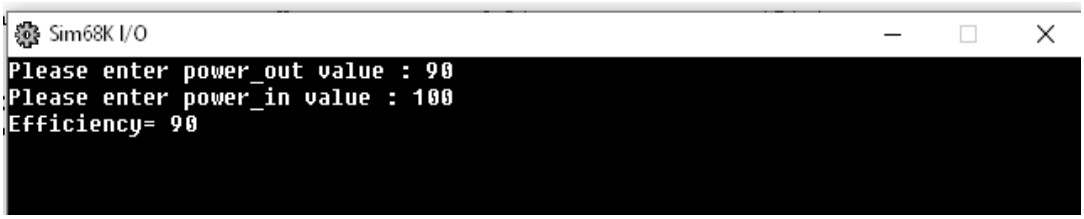
Two methods of computer programming namely MC6800 Assembly language and C++ were used to write a program code for calculating Electric Power and Efficiency were written and explained in this paper. The outputs for the 68k programs are shown in figures 1 for Electric Power and in figure 2 for Efficiency.

then the result is calculated and displayed as shown;

Figures 1: Output for Electric Power

Enter power in;

The result is then as;



Figures 2: Output for Efficiency

The programs written demonstrate the comparison and use of the statements in table 1 for C++ and 68K input and output statements.

V. CONCLUSION

Writing assembly language calculation programmes is more complex and requires knowledge of the microprocessor instruction set, registers, signals and addressing modes, while HLL are more general. There are limited tutorials available to write calculation progammes with the 68k assembly language. The tutorial has demonstrated the use of TRAP # 15 codes with assembly language for engineers to write calculation programmes that require input from the users and display output to the user.

This tutorial will provide students with a guide on writing such programmes. These skills are beneficial especially to final year students who build projects. Knowledge of assembly language is still vital when teaching microprocessor based system courses.

REFERENCES

[1] "The 68000's Instruction Set." [Online]. Available: http://wpage.unina.it/rcanonic/didattica/ce1/docs/68000.pdf.

[2] "Edit68K Editor/Assembler." [Online]. Available: http://www.easy68k.com/features.htm.

[3] T. P. Skinner, *Assembly Language Programming for the 68000*. 1998.

[4] R. Manual, *M68000 FAMILY Programmer's Reference Manual*. Motorola, 1992.