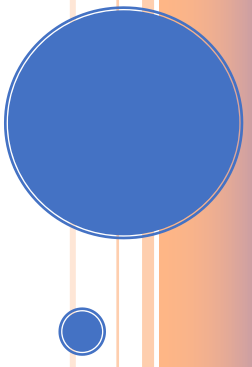
CS1021 Assignment 1

To build a simple calculator

Student Number: 17340382



INTRODUCTION

The objective of this assignment was to make a calculator capable of carrying out simple arithmetic tasks entered by the user using the keyboard. The assignment was broken into three stages

1. Stage one (Console Input)
   This stage involved writing a program that can read unsigned values entered by a user using a keyboard
2. Stage two (Expression Evaluation)
   This stage involved extending the program which was used in stage one to be able to evaluate a simple arithmetic expression and load the result into a register
3. Stage three (Displaying the Result)
   This stage followed on from stage one and two and used the program made in each stage. The key role of this stage was to display the result of the arithmetic expression on the screen.

CONSOLE INPUT

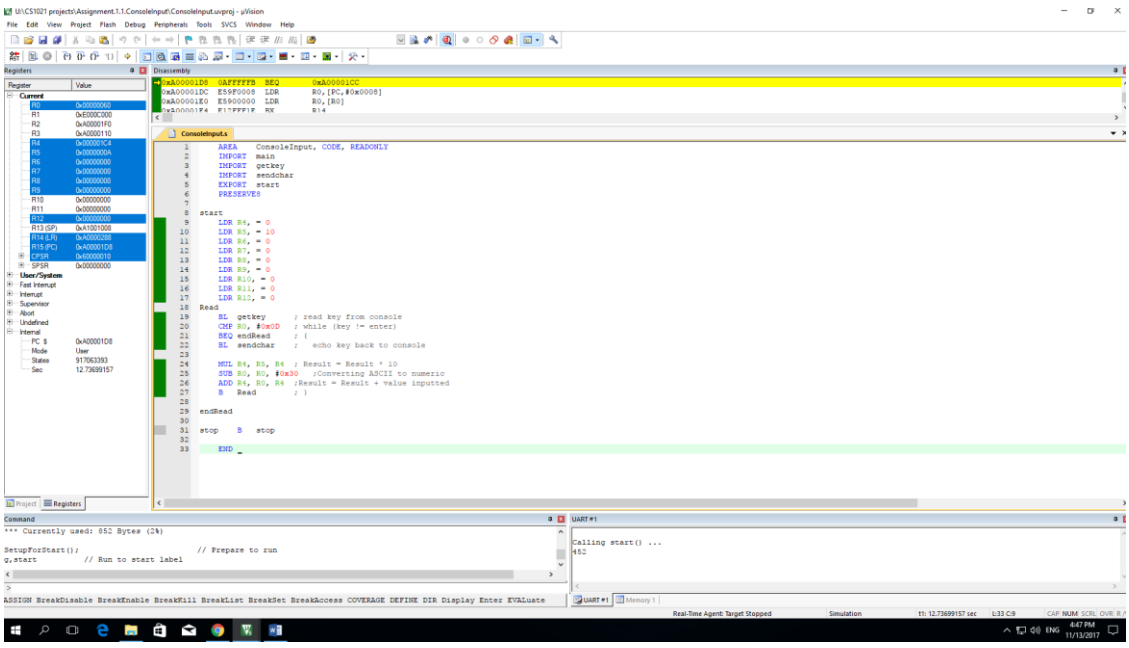
This stage is aimed at designing, writing and testing a program which will read an unsigned value entered by the user and store this value in R4. The numbers will be entered by the user in a string of ASCII characters.

To start off the program, it was necessary to set R4 equal to zero and R5 equal to 10

To read the key inputted off the user using the keyboard it was necessary to use BL getkey. BL stands for branch with link. This reads the key from the console.

The ASCII code 0x0D was then used as it is a code for character return.

The result is then stored as the key value. The result is then multiplied by 10 if there is another character after. The result is then added to the next key value and stored as the result. It is then multiplied by 10 again and this process is continued on as long as the user keeps entering input. This is necessary as it allows the numbers to be stored as the full number that they are. The substitution part is used to convert ASCII to numeric, so it'll print out a numerical value.



Pseudo Code for when 452 is entered

```
Result = 0

ReadKey;

While (key != enter)

{

        PrintKey;

        Result = Result * 10

        Value = key – 0x30

        Result = Result + value

        ReadKey;

}
```

This process was used to convert 4 5 2 from 3 random values into the number 452. It can be used for any values to turn them into numbers.

Expression Evaluation

Stage two is a follow on from stage one. The information is taken from the user in stage one and is used to carry out the arithmetic expression for addition subtraction or multiplication. The first stage is copied into stage two as it is needed to take the information from the user. The arithmetic input (+ - *) is then assigned based on what is entered by the user using the keyboard. The value stored in R0 is then transferred into R6 and then the input is entered by the user. Each number entered into the console is then formed into one whole number. This was explained in the first part. The sign then typed by the user is assigned then to mean either add subtract or multiply. The ASCII code for addition (+) is 0x2B. The ASCII code for subtraction (-) is 0x2D. The ASCII code for multiplication (*) is 0x2A. The next part is to show the answer on the console.

The first part is the same pseudo code from part one

From line 27-32

{

Subtract R0 from 0x2B ignoring the result but updating the carry flags

Branch if the answer is equal to zero

Subtract R0 from 0x2D ignoring the result but updating the carry flags

Branch if the answer is equal to zero

Subtract R0 from 0x2D ignoring the result but updating the carry flags

Branch if the answer is equal to zero

Subtract R0 from 0x2D ignoring the result but updating the carry flags

Branch if the answer is equal to zero

Subtract R0 from 0x2A ignoring the result but updating the carry flags

Branch if the answer is equal to zero

}

This code was used to branch the add subtract or multiply sign depending on what button was pressed on the keyboard by the user

The next part is just the same as part one forming the numbers entered into a number

From line 58 – 71 the calculation is made for weather the numbers are to be either added subtracted or multiplied

{

{

Subtract R0 from 0x2B ignoring the result but updating the carry flags

Branch if the answer is equal to zero

Add the value inputted to the result. This then becomes the new result

}

{

Subtract R0 from 0x2D ignoring the result but updating the carry flags

Branch if the answer is equal to zero

Subtract the value inputted to the result. This then becomes the new result

}

{

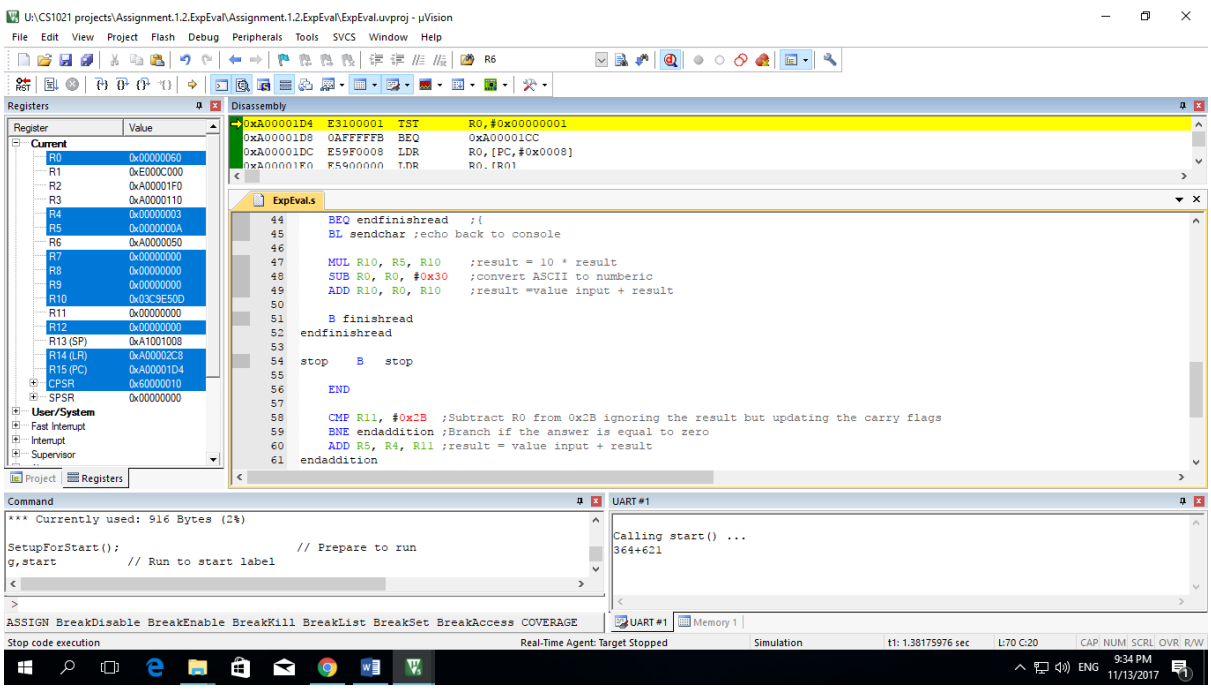
Subtract R0 from 0x2A ignoring the result but updating the carry flags

Branch if the answer is equal to zero

Multiply the value inputted to the result. This then becomes the new result

}

}



Unfortunately the program didn't store the results as I would've hoped so when I went to try carry out a simple test as shown above the program wouldn't store the values correctly

DISPLAYING THE RESULT

Stage three was a continuation on from stage 1 and 2 combined. To start this off I copied and pasted stage 2 as this was used to take the input from the user and also carry out the arithmetic expression. My error in stage 2 meant I was unable to get the program to work exactly as I would've hoped but I continued on anyway.

To start off part three it was required to firstly use the power program which was given to us in the lecture slides to compute powers

While (R0 != 0) {

Result = Result * R0

Result = Result – 1 }

The result is then incremented by one and goes by again

The result was then moved into R11

It is then sent to display the output on the console

The next part was to use the division program which was recently constructed in a lab

Input was firstly taken by a user then went into the division program

Lines 99 – 113 consisted of the division code

Pseudo code is

Move R0 into R4

Allow the quotient to equal to zero

The remainder is then stored in R9

Subtract R1 from 0 and if it's less than then branch to endifzero

{

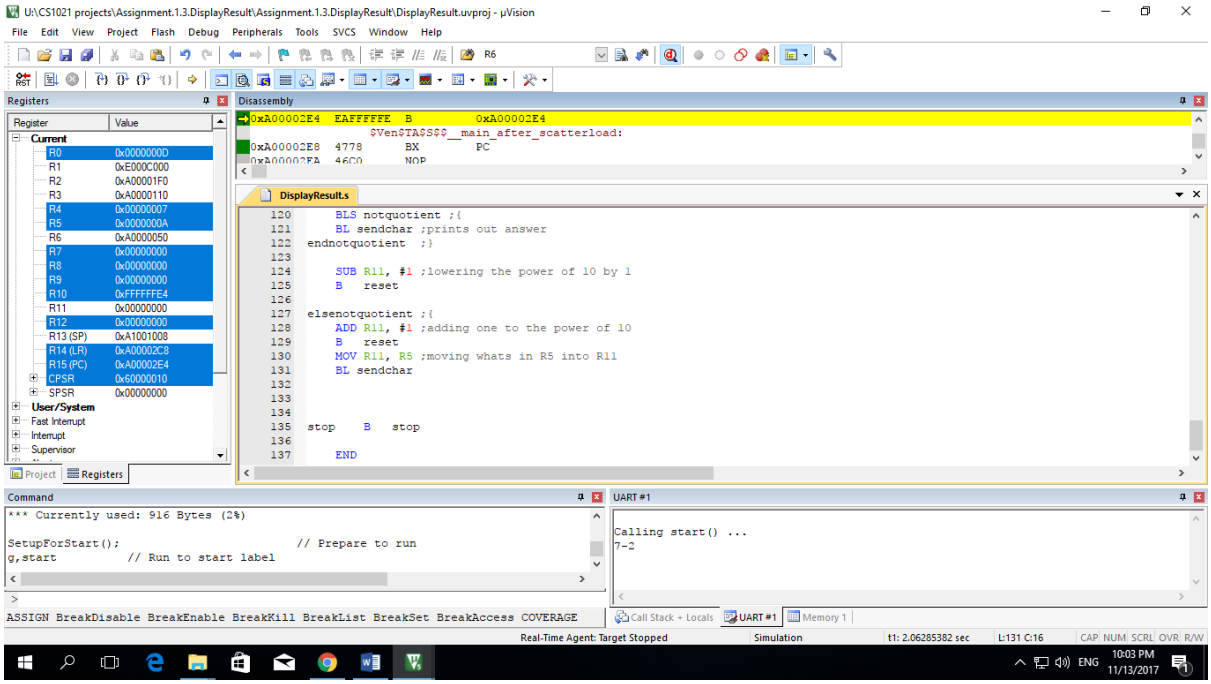
Then subtract R1 from R9 and ignore the result and only update the condition code flags

While the remainder is less then what's in R1 the program runs

1 is then added to the new quotient and R1 is subtracted from the remainder

}

The end of the program is then used to display the results.



RESULTS

I tested my program at the end with many different values but it dint seem to store the values as I had hoped

Test	Result	Comment
2+2	2	The result was incorrect as I have an issue with the registers which im not aware of what is wrong
32*63	3	Once again there is an issue
6-3	6	Error in register
5-13	5	Error