**Microcontroller Calculator Documentation**

CSC202 Final Project

December 20, 2012

For: Professor Fazekas

By: Andy Nguyen

**Table of Contents**

**Project Summary**………………………… p.2

**Program Details:**

Overview……………................ p.3-4

Keypad……………………………… p.5

Flowchart…………………………. p.6

Functions…………………………. p.7-8

**Microcontroller Components:**

Block Diagram………………….. p. 9

Schematics……………………….. p.10-11

**Resources**…………………………………..p.12

**Source Code**……………………………….. \*Available on the provided CD

**Project Summary**

The project consists of the usage of the Dragon 12-Plus microcontroller to perform a specific task. For this project, the microcontroller was programmed to function as a basic infix/postfix calculator. The program was created with C language and uses the LCD, LEDs, and keypad available on the microcontroller. The keypad provides the input and the LCD/LEDs are used for output. This program is an expanded version of the calculator example introduced in the textbook *Programming the Dragon-12* by Richard Haskell (p.64).

To reduce confusion, the calculator in this project will be referred to as an extended calculator.

**Extended Calculator features** (in comparison to the textbook example):

* Perform all standard operations (addition, multiplication, ect.)
* Shift mode to provide additional key functions
* Use infix or postfix notation as input
* Handle multiple digit numbers as well as decimals
* Handle Negative signs
* Handle parenthesis
* Save and recall numbers
* Delete key that removes a char from the keypad buffer
* Replace operators in the operation string
* Use exponential form for input and output
* Handle outputs that exceed the max LCD display length
* Display both operation performed along with input
* Scroll left or right when output is larger than display
* Error prevention measures (ensuring the operation string can be successfully calculated)

**Program Details: Overview**

Summary:

The extended calculator program uses character arrays (strings) to store input from the keypad. The keys pressed are read as chars and are stored in kbuf. The operation created by the input is stored in oper. Additionally, the program uses flags to track the program state (ex. modes) and for error prevention. Some flags will prevent users from performing actions that will result in an illegal operation (ex. Consecutive operators in infix mode).

The program will start when the user enters a key. The input key is read and if it is a number, decimal point, or parenthesis, it is added to kbuf and displayed on the 2nd line of the LCD. Whenever an operator key is read, the program will add kbuf and the operator into oper and display it on the 1st line. Kbuf is then cleared for new input. This process continues until the equal key is pressed. At which oper is then converted to postfix notation (unless in postfix mode) and passed to the calculate function.

The calculate function returns the result as data type double, which must be converted to string before it can be displayed. The result is displayed on the 2nd line. The program prevents further input after the result is obtained and displayed. Only the scroll keys and the clear key will be accepted. The clear key will clear the display and allow new input.

Basic Steps:

- Get key input

- Check/Set flags

- Determine input mode

- Read key

- if number/deci/paren: add to kbuf

- if operator: add kbuf to oper

- if equal: convert oper to postfix, calculate

- Display

**Program Details: Overview**

Display Example:



**Program Details: Keypad**

The program has a choice of two functions to use to interpret the key input. The “Standard” mode function is generally used to read keys as numbers, while the “Shift” mode function reads the keys as operators. Below is a diagram of the keypad and each key’s action. The program begins in standard mode.

Standard Mode Shift Mode

1 2 3 PF + - % (

4 5 6 E 🡺 \* / ^ )

7 8 9 DEL 🡸 < > S CLR

. 0 (-) SHFT MS MR EQ SHFT

PF : Postfix mode on/off < : Scroll left MS: Memory set (save)

E : Exponent > : Scroll right MR : Memory recall

DEL : Delete S: Space EQ : Equals (enter)

SHFT: Shift on CLR: Clear SHFT: Shift off

**Program Details: Functions**

* Function descriptions are available in the source code

|  |
| --- |
| **Main.c** |
| - char kbuf  - int kidx  - char oper  - int oidx  - char mem  - int midx  - char postfix  - Flag flag |
| - main(): void  - createFlag(): Flag  - stdMode(char, Flag): void  - shiftMode(char, Flag): void  - clrLine(int): void  - display(char, int, char): void  - addOper(char, Flag): void  - subDisplay(): void  - getResult(Flag): void  - getPostResult(Flag): void  - scrollLeft(Flag): void  - scrollRight(Flag):void  - formatAns(double):void |

|  |
| --- |
| **Stack.c** |
| - int capacity  - int size  - double \*elements |
| + createStack(int): Stack  + top(Stack): double  + pop(Stack): double  + push(Stack, double): void |

**Program Details: Functions**

|  |
| --- |
| **Operation.c** |
| + operate(Stack, char, Flag): void  + calculate(const char, Flag): double  + op\_preced(char): int  + inToPost(const char, char): bool |

**Microcontroller Components: Block Diagram**

|  |  |  |  |
| --- | --- | --- | --- |
| Port | Port Name | DDR Name | DRAGON12-Plus use |
| Port J | PJ | DDRJ | LED enable |
| Port A | PA | DDRA | Keypad |
| Port B | PB | DDRB | LEDs |
| Port K | PK | DDRK | LCD |

* The hex keypad is connected to Port K. pins PA0-PA3 are configured as outputs while pins PA4-PA7 are configured as inputs.
* The LEDs are connected to Port B and Port J. Both data direction ports are set to outputs when LEDs are enabled.
* The LCD is connect to Port K. Pin PK1 is connected to the enable signal E while the register select signal RS is connected to PK0. PK7 is for the read/write line and is connect through the J5 header which will allow write-only operation of the LCD.



**Resources**

* *Programming the DRAGON12-PLUS-USB in C and Assembly* by Richard Haskell
  + Listing 2.2 Example 2 p.11
  + Listing 3.8 Example 10 p.39-41
  + Listing 4.6 Example 13 p.60-62
* “Shunting-yard Algorithm” – Wikipedia.com
* “Data Structures: Stacks” – thelearningpoint.net
* “Programming in C” – cs.cf.ac.uk/Dave/C