Name & Student ID

Muhammad Armand bin Kushairi, 20180186

Title

Software Project: Developing software for a writing robot

Software Description

The goal of the project is to design a software that can communicate with the writing robot and print out text from a text file of any length with a letter height of 5mm on a sheet of paper. The program is written in C using the CodeBlocks IDE. It will use a virtual rs232 serial port to connect to the robot/Arduino.

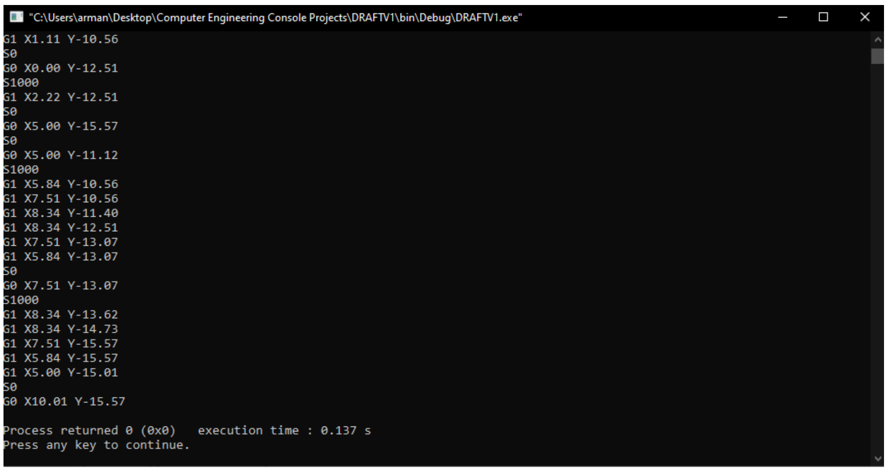
The software is designed to first create a buffer. It will then open the virtual rs232 port using functions from the serial.c and rs232.c file and will wake up the robot by sending specific commands to it and waiting for a $ sign from the robot as a signal its ready.

It will then store font data from “SingleStrokeFont.txt” text file into a structure array (struct StorageArray fontData[]). It will then read the “SampleLines.txt” text file or any file you input into the code by character and sequentially store into variables called int ASCII and int AValue. As a value of ASCII is stored, a custom function called LocateASCIIinFont() will iterate through the structure array to look for the starting point and end point of the specific ASCII character. It will then sequentially offset the coordinates of the font data, scale down the font data by 0.278 of 5/18 to make it 5 units and then convert the coordinates into g-codes to be sent through the RS232 port to the robot using SendCommand(). This code will repeat until all characters in the text file has successfully been converted and sent to robot to be printed and then the file will close. The RS232 virtual port should then close, and the program will end.

Instructions on how to use the program is as such:

1. Replace “SingleStrokeFont.txt” with the name of the font data file you would like to use at line 50
2. Replace “SampleLines.txt” with the name of the text data file you would like to use at line 59
3. Replace value of #define scale if you would like to change the size of the final print data.
4. Make sure #define Serial_Mode in serial.c is uncommented.
5. Check the COM Port of robot that is connected to the computer.
6. Change #define cport_nr to the COM Port of robot connected to computer but minus 1.
7. Compile and run the code.

Several printf statements have been put into the program for diagnostic purposes to check if the expected output is received. Below is an expected sample output of a diagnostic test.



```
"C:\Users\arman\Desktop\Computer Engineering Console Projects\DRAFTV1\bin\Debug\DRAFTV1.exe"
G1 X1.11 Y-10.56
S0
G0 X0.00 Y-12.51
S1000
G1 X2.22 Y-12.51
S0
G0 X5.00 Y-15.57
S0
G0 X5.00 Y-11.12
S1000
G1 X5.84 Y-10.56
G1 X7.51 Y-10.56
G1 X8.34 Y-11.40
G1 X8.34 Y-12.51
G1 X7.51 Y-13.07
G1 X5.84 Y-13.07
S0
G0 X7.51 Y-13.07
S1000
G1 X8.34 Y-13.62
G1 X8.34 Y-14.73
G1 X7.51 Y-15.57
G1 X5.84 Y-15.57
G1 X5.00 Y-15.01
S0
G0 X10.01 Y-15.57

Process returned 0 (0x0)   execution time : 0.137 s
Press any key to continue.
```

Project Files

Main.c – Contains the main code that needs to be run. Changes to text file used, font file used, parameters used for functions and any changes to order or content of sending commands to robots will be here

Rs232.c – contains RS232 library files. Likely no changes needed here. Please leave it as is.

Serial.c – contains serial command, lines and files for communication between the C file and the robot/Arduino. Do not forget to uncomment #define Serial_Mode before use if it has not already been uncommented.

CustomFunctions.c – Contains all custom functions and structure array to be used in the main file. Changed to functions should be done in here.

Rs232.h – Header file containing declarations of functions and items in the rs232.c library. Likely no changes needed. Please leave it as is.

Serial.h – Header file containing declarations of functions and items in serial.c file. Likely no changes needed. Please leave it as is.

CustomFunctions.h – Header file containing declarations of functions and items in CustomFunctions.c . Any changes to definition of functions in CustomFunctions.c will have to updated here.

Functions

*void StoreFontData (char *fileName); // store the font data into a structure array.*

Parameters:

> **fileName – pointer to name of file containing font data.*
>
> *Return value – return 0 if successful.*

float ScaleB4Print(float RawXYToPrint); //to correctly scale the raw coordinates before print.

Parameters:

> *RawXYToPrint - g code coordinates to be scaled by 5/18 or 0.278 before sent to robot*
>
> *Return value - NewXYToPrint – g codes that has been converted.*

*int LocateASCIIinFont(StorageArray fontData[], int ASCII, int *Start, int *Finish);// to search through structure array for the start and end point of specific ascii character inputted.*

Parameter:

> *StorageArray fontData [] – structure array containing font data*
>
> *Int ASCII – required ASCII character to be looked for in array containing font data*
>
> *Int *Start – pointer to store starting point of specific ASCII character in font data*
>
> *Int *Finish- pointer to store ending point of specific ASCII character in font data.*
>
> *Return value – returns 0 if successful, -1 if failed.*

Key Data Items

Name	Data type	Rationale
StorageArray	struct StorageArray { int num1,num2,num3};	The coordinates can only have int values and are stored in a structure to store them together
ASCII	Int ASCII	ASCII of character read from the text file. It only needs to be an integer.
fileNameFont	Char *fileNameFont	Pointer to name of text file containing font data to be read and stored into array. Allows easy switching of files by changing name.
fileToBeRead	Char *fileToBeRead	Pointer to name of text to be read and converted. Allow easy switching of files by changing name
AValue	int AValue	Int to store ASCII value for manipulation as a redundancy measure in case things go wrong.
S	Int S	Starting line of specific ASCII character in StorageArray. It only needs to be an integer
F	Int F	Ending line of specific ASCII character in StorageArray. It only needs to be an integer
XPostScale	Float XPostScale	Store value of x coordinate post scale. It's a float because expected value of x is a decimal.
YPostScale	Float YPostScale	Store value of y coordinate post scale. It's a float because expected value of x is a decimal.
Xmulti, ymulti	Int xmulti, ymulti	Counter for offsets of X and Y coordinates. Makes coding the offset simpler while remaining flexible.
Pvalue	Int Pvalue	To store third number of line of font. It can only have int values
PenCheck	Int PenCheck	To store the third number of previous line of font for determining S0 or S1000. It can only have int values.
XToPrint	Float XToPrint	To store X coordinates after offset. Float because to maintain compatibility with scale factor as it would result in decimals
YToPrint	Float YToPrint	To store Y coordinates after offset. Float because to maintain compatibility with scale factor
*fileToBeRead	FILE *fileToBeRead	File containing text to be read and converted.

Testing Information

Function	Test Case	Test Data	Expected Output
StoreFontData	Success	Char fileNameFont	0.
StoreFontData	Failure Ie. No text file found.	Int 42;	No output
ScaleB4Print	Success	XToPrint ie. 42	XPostScale =11.7
ScaleB4Print	Failure Ie. Wrong Value	Char A	XPostScale = 18.7
LocateASCIIinFont	Success	ASCII = H	Start = 578, Finish = 582 0
LocateASCIIinFont	Failure Ie. Value not found	ASCII = 370882	Start = -1, Finish= -1 -1

Flowchart(s)

Included as separate pdf