



Computer Technology I

Lab. 6 : CyberTech Wall Display



Author: ANAS KWEFATI

Supervisor: ANDERS

HAGGREN

Semester: Autumn 2019

Area: Computer Science

Course code: 1DT301

Contents

1	Task 1	1
2	Task 2	3
3	Task 3	6
4	Task 4	9
5	Task 5	16

1 Task 1

```

1 /*>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
2 ; 1DT301, Computer Technology I
3 ; Date: 2016-09-15
4 ; Author:
5 ;     Anas Kwefati
6 ;
7 ; Lab number: 6
8 ; Title: CyberTech Wall Display
9 ;
10 ; Hardware: STK600, CPU ATmega2560
11 ;
12 ; Function: Program that writes a character on the CyberTech Display.
13 ;
14 ; Input ports: none
15 ;
16 ; Output ports: CyberTech Display.
17 ;
18 ; Subroutines:
19 ; Included files: <avr/io.h>
20 ;
21 ; Other information: Display is connected to the serial port (RS232) on
   the STK600.
22 ; Communication speed is 2400bps.
23 ; Changes in program: (Description and date)
24 <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*/
25
26 #include <avr/io.h>
27 #include <stdio.h>
28 #include <string.h>
29 // #include <util/delay.h>
30 #define FCPU 1000000 // Clock Speed
31 #define BAUD 2400 // Communication Speed Display rate 2400
32 #define MYUBRR (FCPU/16/BAUD-1) // UBRR = 25 -> osc = 1MHz and UBRR =
    47 -> osc = 1,843200MHz
33
34 void uart_int(void);
35 void toPutty(unsigned char data);
36
37 int main(void)
38 {
39     uart_int();
40
41     char* txt = "\rAO0001Hi How are you ? :)";
42     int checksum =0;
43     // We make sure that everything is in it
44     for(int i =0; i<strlen(txt); i++){
45         checksum += txt[i];
46     }
47
48     checksum%=256;
49
50     char toDisplay [ strlen(txt)+3];
51     sprintf(toDisplay , "%s%02X\n", txt , checksum); // \%02x means
print at least 2 digits , prepends it with 0's if there's less.
52     // \%02x is used to convert one character to a hexadecimal string
53
54     for (int i = 0; i<strlen(txt)+3;i++){
```

```

55     toPutty ( toDisplay [ i ] );
56 }
57
58 txt = "\rZD0013C\n";
59 for (int i = 0; i<strlen(txt);i++){
60     toPutty (txt[i]);
61 }
62
63 return 0;
64 }
65
66 //INITIALIZATION OF THE DISPLAY
67
68 void toPutty (unsigned char data){
69     //WAIT FOR DATA TO BE RECEIVED
70     while (!(UCSR1A & (1<<UDRE1)));
71     UDR1 = data;
72 }
73
74 void uart_int(void) {
75     UBRR1L = MYUBRR; //25 because we are setting the board at 1MHz
76     /* Enable receiver and transmitter*/
77     UCSR1B = (1<<RXEN1|1<<TXEN1); // Receive Enable (RXEN) bit //
78     Transmit Enable (TXEN) bit

```



```

17 ; Output ports: CyberTech Display.
18 ;
19 ; Subroutines:
20 ; Included files: <avr/io.h>
21 ;
22 ;Other information: Display is connected to the serial port (RS232) on
    the STK600.
23 ; Communication speed is 2400bps.
24 ;Changes in program: (Description and date)
25 <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*/
26
27 #include <avr/io.h>
28 #include <stdio.h>
29 #include <string.h>
30 // #include <util/delay.h>
31 #define FCPU 1000000 // Clock Speed
32 #define BAUD 2400 // Communication Speed Display rate 2400
33 #define MYUBRR (FCPU/16/BAUD-1) //UBRR = 25 -> osc = 1MHz and UBRR =
    47 -> osc = 1,843200MHz
34
35 void uart_int(void);
36 void toPutty(unsigned char data);
37 void toDisplayOnLCD(char* stringChar);
38
39 int main(void)
40 {
41     uart_int();
42
43     char* txt = "\rAO0001First Line                Second Line";
44
45     toDisplayOnLCD(txt);
46
47
48
49     txt = "\rBO0001Third Line";
50     toDisplayOnLCD(txt);
51
52     txt = "\rZD0013C\n";
53     toDisplayOnLCD(txt);
54
55     return 0;
56 }
57
58
59 //METHOD TO DISPLAY ON THE SCREEN
60 void toDisplayOnLCD(char* stringChar){
61
62     int checksum = 0;
63     //We make sure that everything is in it
64     for(int i =0; i<strlen(stringChar);i++){
65         checksum += stringChar[i];
66     }
67
68     checksum%=256;
69
70     char toDisplay [strlen(stringChar)+3];
71     sprintf(toDisplay, "%s%02X\n", stringChar, checksum); //\%02x
    means print at least 2 digits, prepends it with 0's if there's less
    .

```

```

72 //\\%02x is used to convert one character to a hexadecimal string
73
74 for (int i = 0; i<strlen(stringChar)+3;i++){
75     toPutty(toDisplay[i]);
76 }
77 }
78
79 //INITIALIZATION OF THE DISPLAY
80
81 void toPutty(unsigned char data){
82     //WAIT FOR DATA TO BE RECEIVED
83     while (!(UCSR1A & (1<<UDRE1)));
84     UDR1 = data;
85 }
86
87 void uart_int(void) {
88     UBRR1L = MYUBBRR; //25 because we are setting the board at 1MHz
89     /* Enable receiver and transmitter*/
90     UCSR1B = (1<<RXEN1|1<<TXEN1); // Receive Enable (RXEN) bit //
        Transmit Enable (TXEN) bit
91 }

```



```

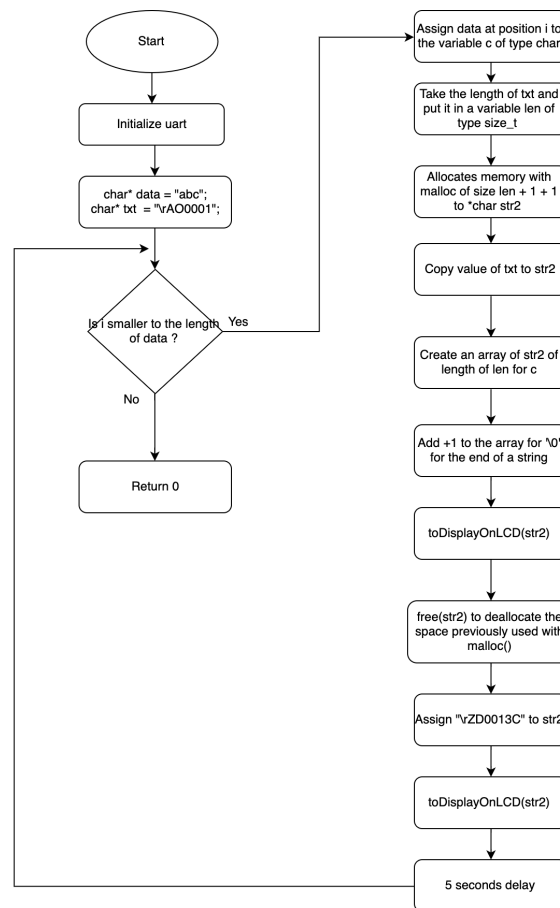
17 ;
18 ; Subroutines:
19 ; Included files: <avr/io.h> and <util/delay.h>
20 ;
21 ;Other information: Display is connected to the serial port (RS232) on
   the STK600.
22 ; Communication speed is 2400bps.
23 ;Changes in program: (Description and date)
24 <====*/
25 #include <avr/io.h>
26 #include <stdio.h>
27 #include <string.h>
28 #include <stdlib.h>
29
30 #define F_CPU 1000000 // Clock Speed
31 #include <util/delay.h>
32 #define BAUD 2400 //Communication Speed Display rate 2400
33 #define MYUBRR (F_CPU/16/BAUD-1) //UBRR = 25 -> osc = 1MHz and UBRR =
   47 -> osc = 1,843200MHz
34
35 void uart_int(void);
36 void toPutty(unsigned char data);
37 void toDisplayOnLCD(char* stringChar);
38
39 int main(void)
40 {
41     uart_int();
42
43
44     char* data = "abc";
45     char *txt = "\rAO0001";
46
47     for(int i =0;i<strlen(data);i++){
48         //The idea is to take char by char and add it one by one to str2
49         char c = data[i];
50         size_t len = strlen(txt); //take the length of txt
51         char *str2 = malloc(len + 1 + 1); //give a length of len and
   allocate a bit more memory with malloc in case
52         strcpy(str2 , txt); // copy txt to str2
53         str2[len] = c; //create an array of str2 with a length of len for
   the char c
54         str2[len + 1] = '\0'; // we add 1 to len and add the end char \0
55         toDisplayOnLCD(str2); //call display
56         free(str2); //free str2 deallocate the space used by malloc()
57
58         str2 = "\rZD0013C";
59         toDisplayOnLCD(str2);
60         _delay_ms(5000); //wait 5s
61     }
62
63
64     return 0;
65 }
66
67
68
69 //METHOD TO DISPLAY ON THE SCREEN
70 void toDisplayOnLCD(char* stringChar){
71

```

```

72  int checksum = 0;
73  //We make sure that everything is in it
74  for(int i =0; i<strlen(stringChar);i++){
75      checksum += stringChar[i];
76  }
77
78  checksum%=256;
79
80  char toDisplay [ strlen(stringChar)+3];
81  sprintf(toDisplay , "%s\\%02X\\n", stringChar , checksum); //\\%02x
    means print at least 2 digits , prepends it with 0's if there's less
82  .
    //\\%02x is used to convert one character to a hexadecimal string
83
84  for (int i = 0; i<strlen(stringChar)+3;i++){
85      toPutty(toDisplay[i]);
86  }
87 }
88
89 //INITIALIZATION OF THE DISPLAY
90
91 void toPutty(unsigned char data){
92     //WAIT FOR DATA TO BE RECEIVED
93     while (!(UCSR1A & (1<<UDRE1)));
94     UDR1 = data;
95 }
96
97 void uart_int(void) {
98     UBRR1L = MYUBRR; //25 because we are setting the board at 1MHz
99     /*Enable receiver and transmitter*/
100    UCSR1B = (1<<RXEN1|1<<TXEN1); // Receive Enable (RXEN) bit //
        Transmit Enable (TXEN) bit
101 }

```



4 Task 4

```

the STK600.
23 ; Communication speed is 2400bps.
24 ; TASK 5 is the same as TASK 4 but with more address to choose (1-9)
25 ;Changes in program: (Description and date)
26 <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<*/
27 #include <avr/io.h>
28 #include <stdio.h>
29 #include <stdlib.h>
30 #include <string.h>
31 #include <stdbool.h>
32
33 #define F_CPU 1000000 // Clock Speed
34 #include <util/delay.h>
35 #define BAUD 2400 //Communication Speed Display rate 2400
36 #define MYUBRR (F_CPU/16/BAUD-1) //UBRR = 25 -> osc = 1MHz and UBRR =
    47 -> osc = 1,843200MHz
37 #define MAX_LINES 3 // Possible lines TASK4
38 #define POSSIBLE_DIGIT_TO_CHOOSE_LINE "123" //TASK4
39
40 // #define POSSIBLE_DIGIT_TO_CHOOSE_LINE "123456789" //TASK 5
41 // #define MAX_LINES 9 //TASK 5 and uncomment the 2 other for TASK 4
42
43 void uart_int(void);
44 void toPutty(unsigned char data);
45 char getChar();
46
47 void improvedToDisplayOnLCD(char address, char* specialCommand, char*
    stringChar);
48 void endToDisplayOnLCD();
49 void setUpToDisplayOnLCD();
50
51 void changeLine(int targetLine);
52 char possibleCharacter(char* possibleLine, char c);
53
54 //SOME GLOBAL VARIABLES
55 int currentLine = 0;
56 bool selectionOfLine = false;
57 char lines[8][24] = { "", "", "", "", "", "", "", "" }; //8ROWS 24
    COLUMNS
58
59
60 int main(void)
61 {
62     uart_int();
63     setUpToDisplayOnLCD();
64
65     while (1) {
66         char character = getChar(); // We get the input from terminal
67
68         //IF SELECT A LINE WE WAIT FOR A VALID DIGIT that is >= 1
69         if (selectionOfLine == true) {
70             if (character < '1') {
71                 continue; // If character <1 do nothing
72             }
73             // We check if the digit that the user has put is in what
    is possible
74             if (possibleCharacter(POSSIBLE_DIGIT_TO_CHOOSE_LINE,
    character)) {
75                 int convertCharToInt = character - '1'; //convert char

```

```

76         to int
77         /*LOGIC behind the conversion :
78         char c = '2';
79         int i = c - '0';
80         it will take the ASCII value of character 2 (50) and
81         0 (48)
82         Then it will subtract them -> int i = 50 - 48 which
83         gives us 2
84         */
85         changeLine(convertCharToInt); //call the method to
86         choose the possible line. Change currentLine by the targetLine (
87         here "convertCharToInt")
88
89         selectionOfLine = false; //False
90     }
91     else {
92         if (character == '>'){
93             selectionOfLine = true; //Make selectionOfLine true
94         } else if (character == 13 ){
95             //Else if character == 'enter'
96             changeLine(-1); // We increment currentLine and we go
97             to the next line
98         } else {
99             // Add character to the end of the selected line
100             char* line = lines[currentLine];
101             sprintf(line , "%s%c", line , character);
102         }
103     }
104     setUpToDisplayOnLCD(); // Update the screen
105 }
106
107 return 0;
108 }
109
110
111
112 //METHOD TO DISPLAY ON THE SCREEN
113 void improvedToDisplayOnLCD(char addressLine , char* specialCommand ,
114 char* stringChar){
115
116     //we get the length of special command which is "O0001"
117     //and we get the length of the specific message in stringChar
118     int specialCommandLen = strlen(specialCommand);
119     int stringCharLen = strlen(stringChar);
120
121     char* toDisplay = malloc(1 + specialCommandLen + stringCharLen + 3)
122     ;//give a length of specialCommand and stringCharLen and allocate a
123     bit more memory with malloc for the end case. We allocate memory
124     for toDisplay
125
126     // ADD EVERYTHING TOGETHER in toDisplay
127     //So we get \rADDRESSLINE_SPECIALCOMMAND_STRINGCHAR
128     //For example \rAO0001a"

```

```

125     sprintf(toDisplay , "\r%c%s%s", addressLine , specialCommand ,
126             stringChar);
127
128     int checksum = 0;
129     // We calculate checksum to make sure that everything is in it
130     for (int i = 0; (toDisplay[i] != 0); i++){
131         checksum += toDisplay[i];
132     }
133
134     checksum %= 256;
135
136     sprintf(toDisplay , "%s%02X\n", toDisplay , checksum); // %02x means
137     // print at least 2 digits , prepends it with 0's if there's less.
138     // %02x is used to convert one character to a hexadecimal string
139
140     for (int i = 0; (toDisplay[i] != 0); i++){
141         toPutty(toDisplay[i]);
142     }
143
144     free(toDisplay); // free toDisplay deallocate the space used by
145     malloc()
146 }
147
148 void endToDisplayOnLCD() {
149     char* txt = "\rZD0013C\n";
150     for(int i = 0; i<strlen(txt);i++){
151         toPutty(txt[i]);
152     }
153 }
154
155
156 //METHOD TO SETUP CHARACTERS FOR EACH LINE
157 void setUpToDisplayOnLCD() {
158
159     //Take currentLine and see if it is <1 if yes then increment
160     lineToDisplay
161     int lineToDisplay = currentLine;
162
163     if (lineToDisplay < 1){
164         lineToDisplay++;
165     }
166
167     //Set up for first and second rows
168     char memory_space_A[48] = "";
169
170
171     //If selectionOfLine is True then add '_' to choosenLine otherwise
172     the currentLine
173
174     char choosenLine;
175     if(selectionOfLine == true){
176         choosenLine = '_';
177     } else {
178         choosenLine = currentLine + '1';
179     }

```

```

179
180 //Add everything in the array of char.
181 sprintf(memory_space_A, "'>' to choose a line:\%c \s",
choosenLine, lines[lineToDisplay-1]);
182
183 // Set up for third row
184 char memory_space_B[48] = " ";
185
186 for (int i = 0; i < 48; i++){
187     //Send data from third row to the memory
188     //memory_space_B[i] = lines[1][i]
189     memory_space_B[i] = lines[lineToDisplay][i];
190 }
191
192 // Send everything to improvedtoDisplayOnLCD to do the calculation
with checksum then it will send it to the screen
193 improvedtoDisplayOnLCD('A', "00001", memory_space_A);
194 improvedtoDisplayOnLCD('B', "00001", memory_space_B);
195 endToDisplayOnLCD();
196 }
197
198
199
200
201 //METHOD TO CHECK IF THE USER INPUT IS ALLOWED TO CHOOSE LINE
202 char possibleCharacter(char* possibleLine, char c){
203     char tempChar;
204
205     //While will continue until the character is \0
206     while (*possibleLine){
207         tempChar = *possibleLine; //We take char by char from
possibleLine and we add it to the char t
208
209         if (tempChar == c) {
210             //In this condition, we check if tempChar is equal to the
user input c
211             return 1; //return 1 if yes
212         }
213         possibleLine++; //increment possibleLine to go to the next
possible char defined by us at the beginning
214     }
215
216     return 0; //0 if not
217 }
218
219
220
221 // METHOD TO CHOOSE LINE BY INCREMENTS OR NOT
222 void changeLine (int targetLine){
223
224     //if targetLine == -1 then we increase currentLine by 1
225     //By default currentLine == 0
226     //So if we press 'enter' it will increase currentLine by 1
227     //Hence currentLine == 1. And it does this as long as we don't
reach the maximum of line possible
228     if (targetLine == -1) {
229         currentLine++;
230         if (currentLine >= MAX_LINES){
231             currentLine = 0; //We reset the currentLine to 0 when it

```

```

exceeds the maximum possible lines.
232     }
233
234     }
235     else {
236         // change to selection
237         currentLine = targetLine;
238     }
239 }
240
241 //INITIALIZATION OF THE DISPLAY
242
243 void toPutty(unsigned char data){
244     //WAIT FOR DATA TO BE RECEIVED and SHOW IT
245     while (!(UCSR1A & (1<<UDRE1)));
246     UDR1 = data;
247 }
248
249 void uart_int(void) {
250     UBRR1L = MYUBRR; //25 because we are setting the board at 1MHz
251     /* Enable receiver and transmitter*/
252     UCSR1B = (1<<RXEN1|1<<TXEN1); // Receive Enable (RXEN) bit //
    Transmit Enable (TXEN) bit
253 }
254
255 char getChar() {
256     //WAIT FOR THE CHARACTER TO BE RECEIVED THEN RETURN IT
257     while (!(UCSR1A & (1<<RXC1)));
258     return UDR1;
259 }

```

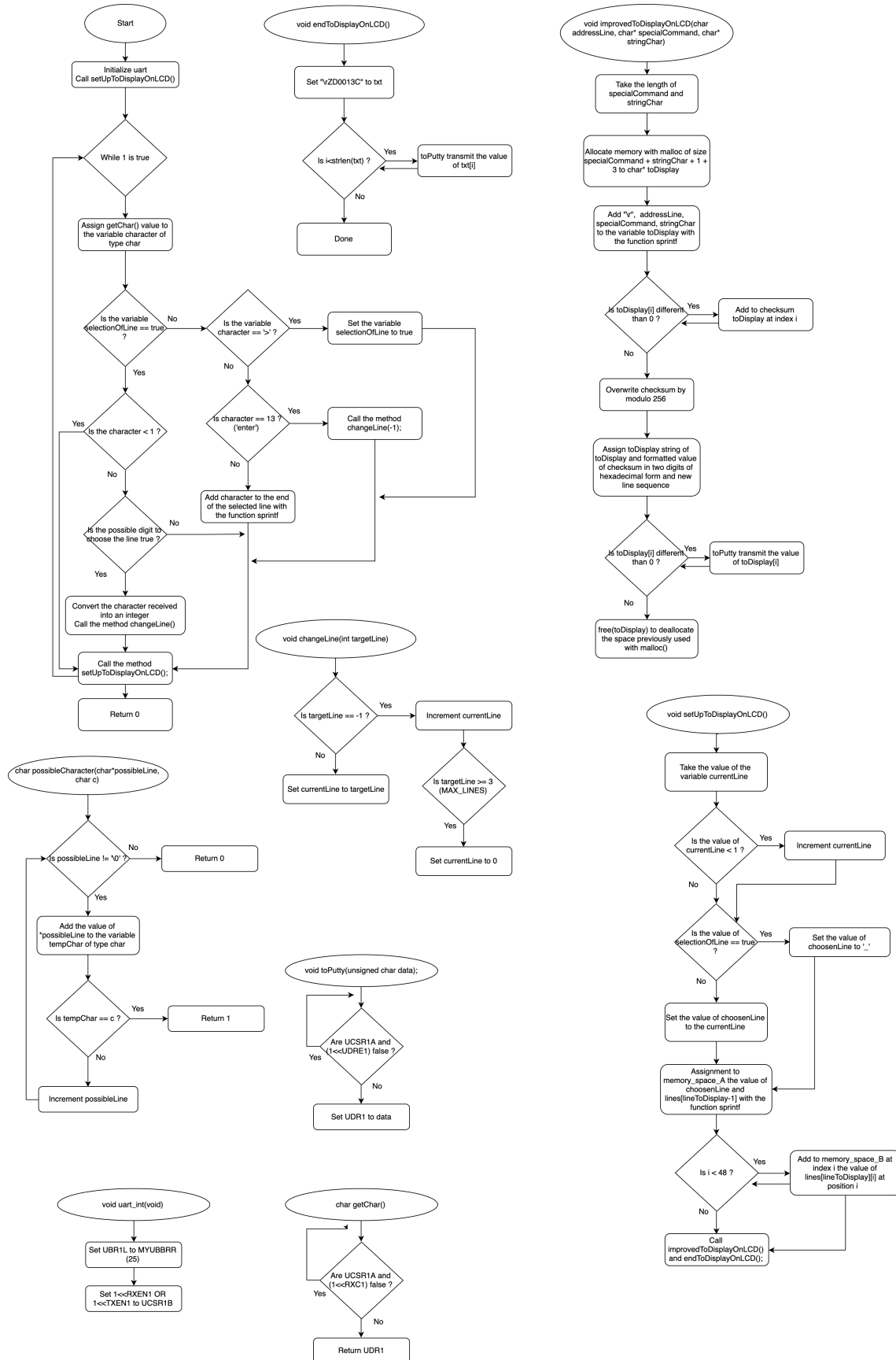



Figure 4: Task 4 flowchart

5 Task 5

Task 5 is the same as Task 4 but with more addresses to choose. (1 to 9). Therefore the flowchart of task 5 is the same as task 4.