Parker Authier & Ben Kepner

Prof. Foster

CE 420-03L Lab 2

Fall Semester 10/24/2019

Parker Authier & Ben Kepner
CE 420-03L Lab 2
Fall Semester 10/24/2019

Table Of Contents

Part 1

Objectives:

The objective of this program was to use an assembly file to move an array of data to a specific memory address. While executing this program, the memory can be monitored to see the changes and experiment with the IDE.

Answers to Questions:

1. 210 bytes = 105 words
2. 0x0
3. GOTO 0x262
4. Address: 0x262 Command: MOV #0x806, W15
5. Address: 0x280 Command: GOTO 0x280

Program Source Code:

Part 1: Assembly Code

```
.include "p24Fxxxx.inc"
.global __reset
.bss

;Declarations
src: .space 2
dst: .space 2
len: .space 2

 ;Code section
        .text
__reset:
        mov #__SP_init, W15 ;stack pointer
        mov #__SPLIM_init,W0
        mov W0, SPLIM ;register for the stack limit

        .equ SOURCE, 0x0900
        .equ DESTINATION, 0x0980
        .equ LENGTH, 10

 ;initalizes values at the start of the program
init:
        mov #SOURCE, W1 ;load source location to register W1
        mov W1, src ;move source location to src
        mov #DESTINATION, W2 ;repeat the same process with destination and register W2
        mov W2, dst
        mov #LENGTH, W0 ;repeat the same process with destination and register W0
        mov WREG, len
        bra z, done ;if the remaining value is 0 it ends the program



loop:
        mov [W1++], [W2++] ;move the values and increment at the same time
        dec W0, W0
        bra z, done ;if the value is 0, end the program
        goto loop

done:
        goto done ;loop forever
        .end
```

Part 2

Objectives:

The objective of this program was to use C code to write a program that parses a string for instances of an indicated character pattern without using the *String* class functions. After the C code was running, the debugger could be used alongside the disassembly window to see the individual instructions that were happening in memory.

Answers to Questions:

6. 00029E → LNK #0x2
7. Text → 0x800, look → 0x82D, count → 0x832, pos → 0x834
8. Input and output parameters are held in different registers at the start of the function. This can be seen in the disassembly file.

10. 576 bytes = 288 words
11. 54 bytes = 27 words

Program Source Code:

Part 2: C Code

```c
/*
 * File:   Lab2CCode.c
 * Author: Parker
 *
 * Created on October 30, 2019, 9:22 PM
 */

#include <stdio.h>
#include <stdlib.h>

#define STRING_END 0x00

char text[] = "The quick brown fox jumps over the lazy dog.";
char look[] = "fox";
int count, pos;
int myStrSearch(char* src, char* look, int* index);

int main(void)
{
	int found;
	count = myStrSearch(text, look, &pos);
	if(count > 0)
	found = 1;
	else
	found = 0;
	return(found);
}

int myStrSearch(char* src, char* look, int* index)
{
	int indexSrc = 0; //index for the character in the string src
	int indexLook = 0; //index for the character in the string look
	int count = 0; //number of matches

	char charSrc, charLook; //current character for both the src and look strings

	//incrementing through src string
	while(src[indexSrc] != STRING_END)
	{
	//compare current character in src to the first character of look
	charSrc = src[indexSrc];
	charLook = look[0];
```

```
        indexLook = 0;
        while(charSrc == charLook)
        {
        //increment through look string until the characters match
        indexLook = indexLook +1;
        charSrc = src[indexSrc + indexLook];
        charLook = look[indexLook];

        //if the the character of the look string is at the end of the string, then it's a match
        if(charLook == STRING_END)
        {
                count = count + 1; //count the match
                if(count == 1)
                {
                *index = indexSrc; //set the index to the start of the matching string
                }
                break;
        }

        //end of src string is the end of the function
        if(charSrc == STRING_END)
        {
                return count;
        }
        }

        indexSrc = indexSrc + 1;

        }

        return count;

}
```

Disassembly Listing File:

Part 2: Disassembly Code

```
Disassembly Listing for Lab2
Generated From:
C:/Users/student/MPLABXProjects/Lab2.X/dist/default/debug/Lab2.X.debug.elf
Oct 31, 2019 8:25:32 AM

---  C:/Users/student/MPLABXProjects/Lab2.X/newmain.c
--------------------------------------------------
1:              /*
2:               * File:   Lab2CCode.c
3:               * Author: Parker
4:               *
5:               * Created on October 30, 2019, 9:22 PM
6:               */
7:
8:              #include <stdio.h>
9:              #include <stdlib.h>
10:
11:              #define STRING_END 0x00
12:
13:              char text[] = "The quick brown fox jumps over the lazy dog.";
14:              char look[] = "fox";
15:              int count, pos;
16:              int myStrSearch(char* src, char* look, int* index);
17:
18:              int main(void)
19:              {
00029E  FA0002     LNK #0x2
20:                int found;
21:                count = myStrSearch(text, look, &pos);
0002A0  208382     MOV #0x838, W2
0002A2  208311     MOV #0x831, W1
0002A4  208000     MOV #0x800, W0
0002A6  07000E     RCALL myStrSearch
0002A8  780200     MOV W0, W4
0002AA  8841B4     MOV W4, count
22:                if(count > 0)
0002AC  8041B4     MOV count, W4
0002AE  520FE0     SUB W4, #0x0, [W15]
0002B0  340003     BRA LE, 0x2B8
23:                    found = 1;
0002B2  200014     MOV #0x1, W4
0002B4  780F04     MOV W4, [W14]
```

```
0002B6  370002     BRA 0x2BC
24:                else
25:                        found = 0;
0002B8  EB0200     CLR W4
0002BA  780F04     MOV W4, [W14]
26:                return(found);
0002BC  78021E     MOV [W14], W4
27:              }
0002BE  780004     MOV W4, W0
0002C0  FA8000     ULNK
0002C2  060000     RETURN
28:
29:              int myStrSearch(char* src, char* look, int* index)
30:              {
0002C4  FA000E     LNK #0xE
0002C6  980740     MOV W0, [W14+8]
0002C8  980751     MOV W1, [W14+10]
0002CA  980762     MOV W2, [W14+12]
31:                int indexSrc = 0; //index for the character in the string src
0002CC  EB0200     CLR W4
0002CE  780F04     MOV W4, [W14]
32:                int indexLook = 0; //index for the character in the string look
0002D0  EB0200     CLR W4
0002D2  980714     MOV W4, [W14+2]
33:                int src_count = 0; //number of matches
0002D4  EB0200     CLR W4
0002D6  980724     MOV W4, [W14+4]
34:
35:                char charSrc, charLook; //current character for both the src and look strings
36:
37:                //incrementing through src string
38:                while(src[indexSrc] != STRING_END)
0002D8  370031     BRA 0x33C
00033C  78021E     MOV [W14], W4
00033E  9002CE     MOV [W14+8], W5
000340  428204     ADD W5, W4, W4
000342  784214     MOV.B [W4], W4
000344  524FE0     SUB.B W4, #0x0, [W15]
000346  3AFFC9     BRA NZ, 0x2DA
39:                {
40:                        //compare current character in src to the first character of look
41:                        charSrc = src[indexSrc];
0002DA  78021E     MOV [W14], W4
0002DC  9002CE     MOV [W14+8], W5
0002DE  428204     ADD W5, W4, W4
0002E0  784294     MOV.B [W4], W5
```

```
0002E2  984765    MOV.B W5, [W14+6]
42:                  charLook = look[0];
0002E4  90025E    MOV [W14+10], W4
0002E6  784294    MOV.B [W4], W5
0002E8  984775    MOV.B W5, [W14+7]
43:                  indexLook = 0;
0002EA  EB0200    CLR W4
0002EC  980714    MOV W4, [W14+2]
44:                  while(charSrc == charLook)
0002EE  37001F    BRA 0x32E
00032E  9042EE    MOV.B [W14+6], W5
000330  90427E    MOV.B [W14+7], W4
000332  52CF84    SUB.B W5, W4, [W15]
000334  32FFDD    BRA Z, 0x2F0
000336  370001    BRA 0x33A
45:                  {
46:                  //increment through look string until the characters match
47:                  indexLook = indexLook +1;
0002F0  90021E    MOV [W14+2], W4
0002F2  E80204    INC W4, W4
0002F4  980714    MOV W4, [W14+2]
48:                  charSrc = src[indexSrc + indexLook];
0002F6  90021E    MOV [W14+2], W4
0002F8  42021E    ADD W4, [W14], W4
0002FA  9002CE    MOV [W14+8], W5
0002FC  428204    ADD W5, W4, W4
0002FE  784294    MOV.B [W4], W5
000300  984765    MOV.B W5, [W14+6]
49:                  charLook = look[indexLook];
000302  90021E    MOV [W14+2], W4
000304  9002DE    MOV [W14+10], W5
000306  428204    ADD W5, W4, W4
000308  784294    MOV.B [W4], W5
00030A  984775    MOV.B W5, [W14+7]
50:
51:                  //if the the character of the look string is at the end of the string, then it's
a match
52:                  if(charLook == STRING_END)
00030C  90427E    MOV.B [W14+7], W4
00030E  524FE0    SUB.B W4, #0x0, [W15]
000310  3A0009    BRA NZ, 0x324
53:                  {
54:                  src_count = src_count + 1; //count the match
000312  90022E    MOV [W14+4], W4
000314  E80204    INC W4, W4
000316  980724    MOV W4, [W14+4]
55:                     if(src_count == 1)
000318  90022E    MOV [W14+4], W4
00031A  520FE1    SUB W4, #0x1, [W15]
00031C  3A000D    BRA NZ, 0x338
56:                     {
57:                             *index = indexSrc; //set the index to the start of the matching
string
00031E  90026E    MOV [W14+12], W4
000320  780A1E    MOV [W14], [W4]
58:                     }
59:                     break;
000322  37000B    BRA 0x33A
000338  000000    NOP
60:                     }
61:
62:                     //end of src string is the end of the function
63:                     if(charSrc == STRING_END)
000324  90426E    MOV.B [W14+6], W4
000326  524FE0    SUB.B W4, #0x0, [W15]
000328  3A0002    BRA NZ, 0x32E
64:                     {
65:                     return src_count;
00032A  90022E    MOV [W14+4], W4
00032C  37000E    BRA 0x34A
66:                     }
67:                     }
68:
69:                     indexSrc = indexSrc + 1;
00033A  E80F1E    INC [W14], [W14]
70:
71:             }
72:
73:             return src_count;
000348  90022E    MOV [W14+4], W4
74:
75:           }
00034A  780004    MOV W4, W0
00034C  FA8000    ULNK
00034E  060000    RETURN
```