CPE 323 Introduction to Embedded Computer Systems
Homework IV

1 (25)	2 (20)	Total
__	__	____

Nolan Anderson

Problem #1. (25 points) C language, Stack Data Placement, Pointers
Consider the following C program. Assume that the register SP at the beginning points to 0x0E00. Answer the following questions. Assume all variables are allocated on the stack, and in the order as they appear in the program.
A. (9 points) Illustrate the content of the stack at the moment before the statement at line 7 is executed.
B. (9 points) Comment the code (lines 7 – 11) indicating the result of each statement. Illustrate the content of the stack at the end of execution of the statement in line 11.
What are the contents of arrays x and carr?
C. (6 points) Show assembly language implementation of the statement at lines 5, 7 (think how would compiler translate these statements; it knows where the variables are placed relatively to the current top of the stack).

FE FC FA

```
1    volatile unsigned int x[3]={1, 32768, 65533};
2    volatile char carr[4]={‘C’, ‘1’, ‘0’, ‘a’};
3    volatile long int z = 65540;
4    volatile char *p_c = carr;       //
5    volatile unsigned int *p_i = x; //
6
7    *(p_c + 2) += ‘6’;       //
8    p_i = p_i + 3;           //
9    *p_i += *(p_i -6);       //
10   p_c = p_c + 3;           //
11   *p_c = ‘A’;              //
```

Address	M[15..0]	Comment
0x0E00	—	TOS
0x0DFE	FFFD	65533
0x 0DFC	8000	32768
0x0DFA	0001	1
0x0 DF8	61 30	"a" "0"
0x0 DF6	31 43	"1" "C"
0x0 DF4	0001	65540 upper
0x0 DF2	0004	65540 lower
0x0 DF0	0x0DF6	Points to bottom (1st) element
0x0DEE	0x0DFA	First element
0x0DEC		
0x0 DEA		

B.

pi ← pi + *pi-6

1	`volatile unsigned int x[3]={1, 32768, 65533};`
2	`volatile char carr[4]={‘C’, ‘1’, ‘0’, ‘a’};`
3	`volatile long int z = 65540;`
4	`volatile char *p_c = carr;       //`
5	`volatile unsigned int *p_i = x; //`
6	
7	`*(p_c + 2) += ‘6’;      //` DF8 ← DF8 + 36 DF8 | 61 66
8	`p_i = p_i + 3;          //` DFA ← DFA + 3·2 = DFA + 6 = 0E00
9	`*p_i += *(p_i -6);      //` 0E00 ← 0E00 + 0001
10	`p_c = p_c + 3;          //` p-c ← p-c + 6 = 0x0DFC
11	`*p_c = ‘A’;             //` 0x0DFC ← ‘A’ ⇒ 0x0DFC ←

Address	M[15..0]	Comment
0x0E00	0E01	`*p_i += *(p_i -6);`
0x 0DFE	FFFD	
0x 0DFC	0041	*p-c = 'A'
0x0DFA	0001	
0x0 DF8	61 66	*(p-c+2) += '6'
0x0 DF6	31 43	
0x0 DF4	0001	
0x0 DF2	0004	
0x0 DF0	0DFC	p-c = p-c + 3
0x0 DEE	0E00	`p_i = p_i + 3;`
0x0 DEC		
0x0 DEA		

C. **C. (6 points)** Show assembly language implementation of the statement at lines 5, 7 (think how would compiler translate these statements; it knows where the variables are placed relatively to the current top of the stack).

```
; volatile unsigned int *p_i = x
```

MOV.W SP, R7

MOV.W R7, 0x0010(SP)

```
*(p_c + 2) += ‘6’;
```

ADD.B #0x0036, 0x0002(R8)

Problem #2. (20 points) C language, Stack Data Placement, Pointers

Consider the following C program. Assume that the register SP at the beginning points to 0x0A00. Answer the following questions. Assume all variables are allocated on the stack, and in the order as they appear in the program. ASCII code for character ‘0’ is 48.

1	`int main( void )   {`
2	4 `volatile long int a = -67;`
3	4 `volatile int c = -6, d = -5;`
4	4 `volatile char mych[4] = {‘0’, ‘2’, ‘4’, ‘8’};`
5	2 `volatile long int *lpa = &a;`
6	2 `volatile int *pi = &d;` 4256
7	`lpa = lpa - 2;` // 09FC - 8 = 09F4
8	`*lpa = *lpa + 1026;` // 3230 + 1026 = 4256
9	`pi += 2;` 2x2=4 // 09F8 + 4 = 09FC
10	`*pi = *pi + c;` // FFBD + FFFA
11	`}`

0A00			
09FE	FFFF		} a
09FC	FFBD		
09FA	FFFA		c
09F8	FFFB		d
09F6	38	34	} mych
09F4	~~32~~	~~30~~	
09F2	09FC →09F4		lpa
09F0	09F8 →09FC		pi
09EE			
09EC			
09FA			

Fill in the following table by determining the values/addresses given below.

#	Question?	Value/Address
1	The number of bytes allocated on the stack for the variable declared in line 2.	4 bytes
2	The number of bytes allocated on the stack for the character array declared in line 4.	4 bytes
3	The number of bytes allocated on the stack for all variables declared in lines 2-6.	16 bytes
4	Value of mych[0] after initialization performed in line 4.	'0' or 0x30
5	Address of variable a (&a).	09FC
6	Value of lpa at the moment after the statement in line 7 is executed.	09F4
7	Value of *lpa at the moment after the statement in line 8 is executed.	4256
8	Value of mych[0] at the moment after the statement in line 8 is executed.	0x56
9	Value of pi at the moment after the statement in line 9 is executed.	09FC
10	Value of *pi at the moment after the statement in line 10 is executed.	FFB7

(Note: The table below is not going to be graded; you can use it to sketch the stack if you want)
