Chris Thomas

11/1/2022

Lab 4 Report

3.)

a.) No resetting the device did not change any of the RAM.

b.) gd = 0x800002B0, 0x55555555

ge = 0x80000280, 0x55555555

gc = 0x800002AC, 0x55555555

a[] = 0x9D0001A0, CST 337 Lab 4

b[] = 0x800002B4, "UUUUUUUUUUUUUUUUUUUU"

c[] = 0x9D0001A0, "Initialized Constant String\0"

d[] = 0x800002CC, "UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUU UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUU"

c.) Global Constants = KSEG0, Program Flash

Global Variables = KSEG0, Data RAM

4.)

a.) The simulator says that the sp = 0x8007FFE0, fp = 0x00000000, gp = 0x80008270

b.) It is in KSEG1, Boot Flash

The manual says this is the address of the Reset, Soft Reset, NMI exception

5.)

a.) sp = 0x8007FFF8, gp = 0x80008270

b.) sp = KSEG0, Data RAM, gp = KSEG0, Data RAM

6.)

a.) Yes sp changed sp = 0x8007FFE0

b.) ge, gd, gc, d, and b all changed

c.) gp was initialized to 0x80008270 even though the code said to set it to 0.

d.) The runtime start-up code will copy all the blocks of initial values from program memory to RAM so the variables will contain the correct values before main is executed.

7.)

a.) All the static variables have their values before being initialized

b.) c and a are in Program Flash and they are constant global variables the other values are either not constant or local variables

c.) The Data RAM is initialized when the line of code is executed

d.) The Program Flash is initialized at startup

9.)

a.) All the non static variables have addresses close to the sp/fp

b.) This would be referred to as the stack.

c.) The keyword const tells the compiler that if we ever try to assign a value to it to generate an error so it is protected by the compiler.

10.)

a.) 0x8007FF90

b.) KSEG0, Data RAM

11.)

a.) 0x9D00_01BC

b.) KSEG0, Program Flash

c.) Initializing a string this way is fine, strcpy has more overhead since it is a function.

12.)

a.) b[] = 0x9D00_0254, d[] = 0x9D00_026C, ge = 0x9D00_02E8

b.) Program Flash

c.) Because it put the two const together and the two non const together.

d.) The non const version is more efficient.

e.) Because the const versions are stored in Flash and we have far more RAM than flash.

13.)

a.) Because the array not big enough to hold it all

b.) The extra went into d[].

c.) The null character

d.) The size of the array.

e.) It would fill the array with what is in b[].

f.) Since it didn't have a null character strcpy doesn't think its done until the array is full so it keeps writing the same thing over and over again.

14.)

a.) 0x9D000648, 0x9D000898

15.)

Data Ram

section	variables	address
.sdata	Ge isf	0x80000280 0x80000284
.sbss	Gc Gd	0x800002ac 0x800002B0
.data	b[] d[]	0x800002b4 0x800002CC
heap		0x80000338
stack		0x80000360

Program Flash

section	variables and initializers	address
.text.main_entry		0x9d000110
.text		0x9d000648
.app_excpt		0x9d000180
.vectors		0x9d000200
.dinit	 Initb[] Initd[] Initge	0x9d000248 0x9D00_0254 0x9D00_026C 0x9D00_02E8
.rodata	A[] c[] initla iscf Lcc	0x9d000190 0x9D0001A0 0x9D00_01BC 0x9D0001D0 0x9D0001D8
.reset		0xbfc00000
.bev_excpt		0xbfc00380

16.)

0x000000009d00089c

17.)

The .resets purpose is to handle the Reset, Soft Reset, and NMI exceptions. The .bev_excpt is for handling general exceptions and interrupts. The .app_excpt handles general exceptions and interrupts in program flash memory.

18.)

The bottom of the stack moved up to a higher address because we allocated more space for using malloc and then freed it.

```
#include <xc.h>
#include <string.h>

const char a[] = "CST 337 Lab 4";
char b[] =       "Initialized Global Var";
const char c[] = "Initialized Constant String";
char d[100] =    "Initialized String Array";
unsigned int gc = 0; //global initialized to 0
unsigned int gd; //global uninitialized
unsigned int ge = 0x45; //global initialized to 0x45

int main(void)
   {
   unsigned int ic = 0x0F1E2D3C;
   unsigned int id = 1;
   unsigned int ie;
   const unsigned int icf = 0x98765432;
   static unsigned int isf = 0x67452301;
   static const unsigned int iscf = 0xABCDEFFE;

   unsigned long long Lc = 0x4B5A69788796A5B4LL;
   static const unsigned long long Lcc = 0xFEDCBA9876543210LL;
   unsigned long long Ld = 1;
   unsigned long long Le;

   char la[] = "Local String Test";

   ie = 0x19283746;
```

```
    ic = icf;
    ic = isf;
    ic = iscf;
    id = gc;

    Le = 0x0123456789ABCDEFLL;
    Le = Lcc;
    Le = Lc;
    char * cpy = malloc(sizeof(char)*18);
    strcpy(cpy, d);
    strcpy(d,b); // Copy the b string into the d string
    strcpy(d,a);
    strcpy(d,c);
    strcpy(b,c);
    free(cpy);
    return 0;
}
```