Unit3 lesson2

Lab1 : Creating a bare-metal Software to send a String

Generating output files :

script.sh part1:

```
#create preprocessed files
arm-none-eabi-gcc.exe -E ./main.c -o ./main.i
arm-none-eabi-gcc.exe -E ./uart.c -o ./uart.i

#define Compiler and Assembler flags
GCC_Flags="-mcpu=arm926ej-s -c -I. -nostdlib"
ASM_Flags="-mcpu=arm926ej-s -M"

#create object files
arm-none-eabi-as.exe $ASM_Flags ./startup.s -o ./startup.o
arm-none-eabi-gcc.exe $GCC_Flags ./main.c -o main.o
arm-none-eabi-gcc.exe $GCC_Flags ./uart.c -o uart.o



#create objdump files
arm-none-eabi-objdump.exe -h ./startup.o > ./start.txt
arm-none-eabi-objdump.exe -h ./main.o > ./main.txt
arm-none-eabi-objdump.exe -h ./uart.o > ./uart.txt
```

script.sh part2:

```
20
21
22  #extract the symbol table of each object
23  arm-none-eabi-nm.exe ./main.o > mainSTable.txt
24  arm-none-eabi-nm.exe ./uart.o > uartSTable.txt
25  arm-none-eabi-nm.exe ./startup.o > startupSTable.txt
26
27  #link and produce the map file
28  arm-none-eabi-ld.exe -T ./linker_script.ld -Map=Map_File.map ./startup.o
29
30
31  #convert elf to bin file
32  arm-none-eabi-objcopy.exe  -O binary ./Azazy.elf ./Azazy.bin
33
34
35  #run the simulation
36  qemu-system-arm.exe -M versatilepb -m 128M -nographic -kernel ./Azazy.elf
37
38
```

sections of main.o :

```
1
2  ./main.o:     file format elf32-littlearm
3
4  Sections:
5  Idx Name          Size      VMA       LMA       File off  Algn
6    0 .text         00000018  00000000  00000000  00000034  2**2
7                    CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
8    1 .data         00000064  00000000  00000000  0000004c  2**2
9                    CONTENTS, ALLOC, LOAD, DATA
10   2 .bss          00000000  00000000  00000000  000000b0  2**0
11                   ALLOC
12   3 .rodata       00000064  00000000  00000000  000000b0  2**2
13                   CONTENTS, ALLOC, LOAD, READONLY, DATA
14   4 .comment      00000012  00000000  00000000  00000114  2**0
15                   CONTENTS, READONLY
16   5 .ARM.attributes 00000032  00000000  00000000  00000126  2**0
17                   CONTENTS, READONLY
18
```

sections of uart.o :

```
./uart.o:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         00000050  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .data         00000000  00000000  00000000  00000084  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  00000084  2**0
                  ALLOC
  3 .comment      00000012  00000000  00000000  00000084  2**0
                  CONTENTS, READONLY
  4 .ARM.attributes 00000032  00000000  00000000  00000096  2**0
                  CONTENTS, READONLY
```

sections of startup.o :

```
./startup.o:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         00000010  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000000  00000000  00000000  00000044  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  00000044  2**0
                  ALLOC
  3 .ARM.attributes 00000022  00000000  00000000  00000044  2**0
                  CONTENTS, READONLY
```

symbols of main.o :

```
00000000 T main
00000000 D str
00000000 R str2
         U Uart_Send_Str
```

symbols of uart.o :

```
00000000 T Uart_Send_Str
```

symbols of startup.o :

```
1           U main
2  00000000 t reset
3           U stack_top
4  00000008 t stop
5
```

symbols of the executable file :

```
1  00010020 T main
2  00010010 t reset
3  00010000 t reset
4  00011150 D stack_top
5  00010008 t stop
6  00010018 t stop
7  000100ec D str
8  00010088 R str2
9  00010038 T Uart_Send_Str
10
```

Simulation output :

```
PC@PC-PC MINGW32 /d/EmbeddedSystems/Online_Diploma/Online_Diploma_Repo/Embedded_
C/unit3_lesson2/codes and output files (main)
$ ./script.sh
./startup.s: Assembler messages:
./startup.s: Warning: end of file in comment; newline inserted
Learn-in-depth: Ahmed Azazy
```