

A dark blue vertical bar runs down the left side of the page. A blue arrow points to the right from the bar, containing the date.

8/20/2023

Assignment Lesson 2

Report of compilation, linking and simulation

Several thin, curved lines in shades of blue and grey sweep upwards from the bottom left corner of the page.

Name: Andrew Adel Hosny Goued
LEARN IN DEPTH DIPLOMA

1) Contents

1) Contents.....	1
2) Files	2
I- App.c	2
II- Uart.c.....	2
III- Uart.h.....	2
IV- Startup.c	3
V- Linker_script.ld	3
3) Compile files	4
I- Commands for compile.....	4
II- Show sections of object_file.....	4
a- App.o	4
b- Uart.o	4
c- Startup.o.....	4
III- Show symbol table for object files	5
4) Use linker_script to get executable_file and mab file	5
I- Command.....	5
II- Show sections for learn-in-depth.elf.....	5
III- Show symbol table of learn-in-depth.elf.....	6
5) Binary file to use in burn	6
I- Command for get binary file.....	6
II- Burn binary file on board using qemu	6
a. Command.....	6
b. Running.....	6

2) Files

I- App.c

```
E:\Courses\Embedded System KS\Units\Unit 3\repo\Master-Embedded-System\EmbeddedC_Course\Lesson2_Assignm
File Edit Selection Find View Goto Tools Project Preferences Help
app.c x uart.h x uart.c x startup.s
1 // @learn in depth
2 // Mastering embedded system course
3 // Written by Andrew Adel
4
5 #include "uart.h"
6 unsigned char* str = "to create rodata section";
7 unsigned char string_buffer[100] = "learn-in-depth:<Andrew Adel>";
8
9 void main(void){
10     Uart_Send_string(string_buffer);
11 }
```

II- Uart.c

```
E:\Courses\Embedded System KS\Units\Unit 3\repo\Master-Embedded-System\EmbeddedC_Course\Lesson2_Assignment\uart.c - St
File Edit Selection Find View Goto Tools Project Preferences Help
app.c x uart.h x uart.c x startup.s x
4
5 #include "uart.h"
6
7 #define UART0DR *((volatile unsigned int* const)((unsigned int*) 0x101f1000))
8
9 void Uart_Send_string(unsigned char* P_tx_string){
10     while(*P_tx_string != '\0'){
11         UART0DR = (unsigned int)(*P_tx_string);
12         P_tx_string++;
13     }
14 }
```

III- Uart.h

```
E:\Courses\Embedded System KS\Units\Unit 3\repo\Master-Embedded-System\EmbeddedC_Course\Lesson2_Assignment\uart.h - St
File Edit Selection Find View Goto Tools Project Preferences Help
app.c x uart.h x uart.c
1 // @learn in depth
2 // Mastering embedded system course
3 // Written by Andrew Adel
4
5 #ifndef _UART_H_
6 #define _UART_H_
7
8 void Uart_Send_string(unsigned char* P_tx_string);
9
10 #endif
```

IV- Startup.c

```
E:\Courses\Embedded System KS\Units\Unit 3\repo\Master-Embedded-System\EmbeddedC_Course\Lesso
File Edit Selection Find View Goto Tools Project Preferences Help
app.c x uart.h x uart.c x start
1 @@Created by Eng. Andrew Adel (for learn-in-depth Diploma)
2 .globl reset
3 reset:
4     ldr sp, =stack_top
5     bl main
6 stop: b stop
7
```

V- Linker_script.ld

```
E:\Courses\Embedded System KS\Units\Unit 3\repo\Master-Embedded-System\EmbeddedC_Course\Lesson2_Assignment\linker_script.ld - Sublime Text
File Edit Selection Find View Goto Tools Project Preferences Help
app.c x uart.h x uart.c x startup.s x commands.sh
1 /* learn-in-depth
2 Unit2_Lesson2_Lab1
3 Mastering Embedded System Diploma
4 Eng. Andrew Adel
5 */
6
7 ENTRY(reset)
8 MEMORY
9 {
10     Mem (rwx) : ORIGIN = 0x00000000, LENGTH = 64M
11 }
12
13 SECTIONS
14 {
15     . = 0x10000;
16     .startup . :
17     {
18         startup.o(.text)
19     }> Mem
20     .text :
21     {
22         *(.text) *(.rodata)
23     }> Mem
24     .data :
25     {
26         *(.data)
27     }> Mem
28     .bss :
29     {
30         *(.bss) *(COMMON)
31     }> Mem
32     . = . + 0x1000; /* 4KB for Stack Memory */
33     stack_top = .;
34 }
```

3) Compile files

I- Commands for compile

```
arm-none-eabi-gcc.exe -c -I . -mcpu=arm926ej-s app.c -o app.o
arm-none-eabi-gcc.exe -c -I . -mcpu=arm926ej-s uart.c -o uart.o
arm-none-eabi-as.exe -mcpu=arm926ej-s -g startup.s -o startup.o
```

```
app.o:      file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text          00000018  00000000  00000000  00000034  2**2
    CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data          00000068  00000000  00000000  0000004c  2**2
    CONTENTS, ALLOC, LOAD, RELOC, DATA
  2 .bss           00000000  00000000  00000000  000000b4  2**0
    ALLOC
  3 .rodata        0000001c  00000000  00000000  000000b4  2**2
    CONTENTS, ALLOC, LOAD, READONLY, DATA
  4 .comment       00000012  00000000  00000000  000000d0  2**0
    CONTENTS, READONLY
  5 .ARM.attributes 00000032  00000000  00000000  000000e2  2**0
    CONTENTS, READONLY
```

b- 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
```

c- 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
  4 .debug_line     0000003a  00000000  00000000  00000066  2**0
    CONTENTS, RELOC, READONLY, DEBUGGING
  5 .debug_info     0000009d  00000000  00000000  000000a0  2**0
    CONTENTS, RELOC, READONLY, DEBUGGING
  6 .debug_abbrev    00000014  00000000  00000000  0000013d  2**0
    CONTENTS, READONLY, DEBUGGING
  7 .debug_aranges  00000020  00000000  00000000  00000158  2**3
    CONTENTS, RELOC, READONLY, DEBUGGING
```

III- Show symbol table for object files

```
show symbol table for all object files
-----
app.o:
00000000 T main
00000000 D str
00000004 D string_buffer
          U Uart_Send_string
uart.o:
00000000 T Uart_Send_string
startup.o:
          U main
00000000 T reset
          U stack_top
00000008 t stop
```

4) Use linker_script to get executable file and mab file

I- Command

```
arm-none-eabi-ld.exe -T linker_script.ld startup.o app.o uart.o -o learn-in-depth.elf -
Map=Map_file.map
```

II- Show sections for learn-in-depth.elf

```
learn-in-depth.elf:      file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .startup       00000010  00010000  00010000  00008000  2**2
    CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .text          00000084  00010010  00010010  00008010  2**2
    CONTENTS, ALLOC, LOAD, READONLY, CODE
  2 .data          00000068  00010094  00010094  00008094  2**2
    CONTENTS, ALLOC, LOAD, DATA
  3 .ARM.attributes 0000002e  00000000  00000000  000080fc  2**0
    CONTENTS, READONLY
  4 .comment        00000011  00000000  00000000  0000812a  2**0
    CONTENTS, READONLY
  5 .debug_line     0000003a  00000000  00000000  0000813b  2**0
    CONTENTS, READONLY, DEBUGGING
  6 .debug_info     0000009d  00000000  00000000  00008175  2**0
    CONTENTS, READONLY, DEBUGGING
  7 .debug_abbrev   00000014  00000000  00000000  00008212  2**0
    CONTENTS, READONLY, DEBUGGING
  8 .debug_aranges  00000020  00000000  00000000  00008228  2**3
    CONTENTS, READONLY, DEBUGGING
```

III- Show symbol table of learn-in-depth.elf

```
analyze the executable file:
00010010 T main
00010000 T reset
000110fc D stack_top
00010008 t stop
00010094 D str
00010098 D string_buffer
00010028 T Uart_Send_string
```

5) Binary file to use in burn

I- Command for get binary file

```
arm-none-eabi-objcopy -O binary learn-in-depth.elf learn-in-depth.bin
```

II- Burn binary file on board using qemu

a. Command

```
qemu-system-arm -M versatilepb -m 128M -nographic -kernel learn-in-depth.bin
```

b. Running

