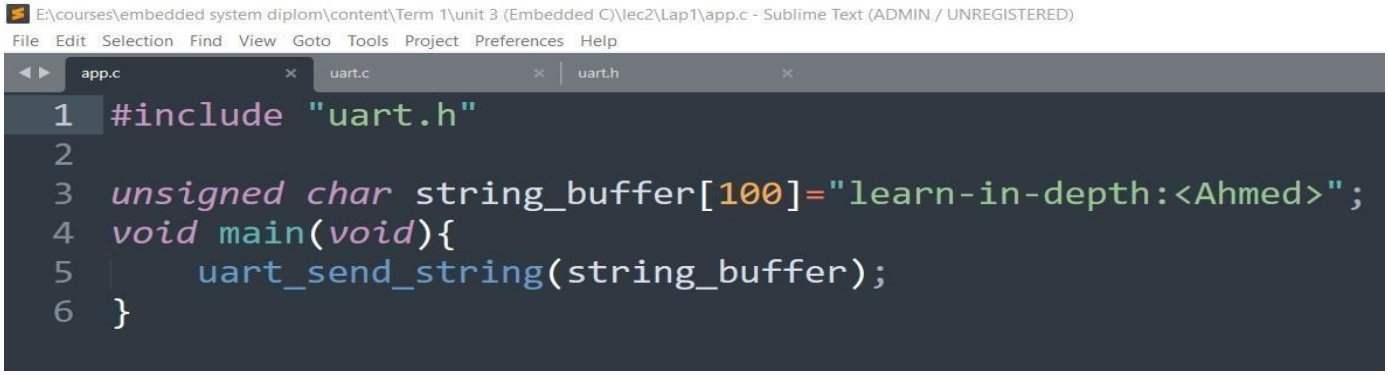
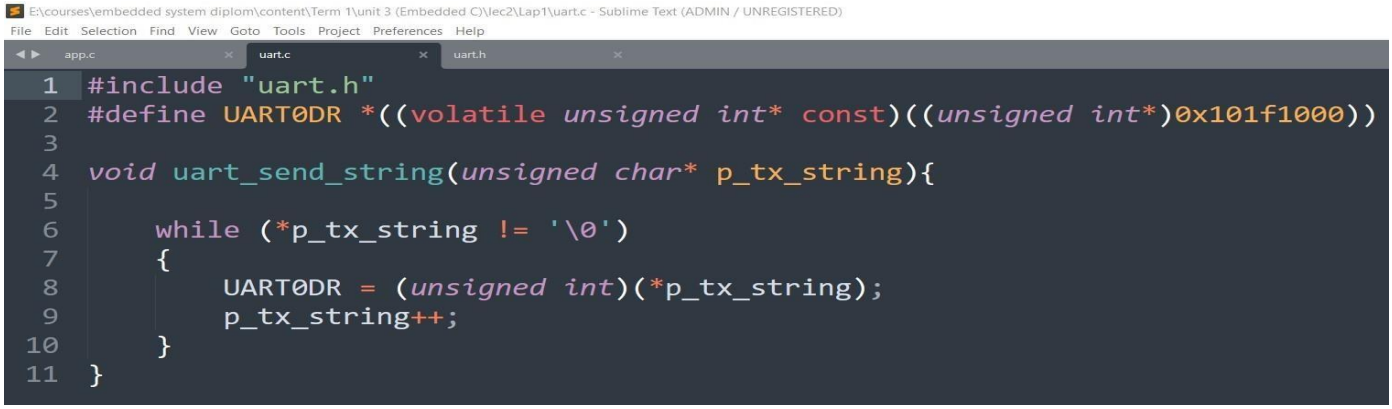
Report about lab1

1-Codes:

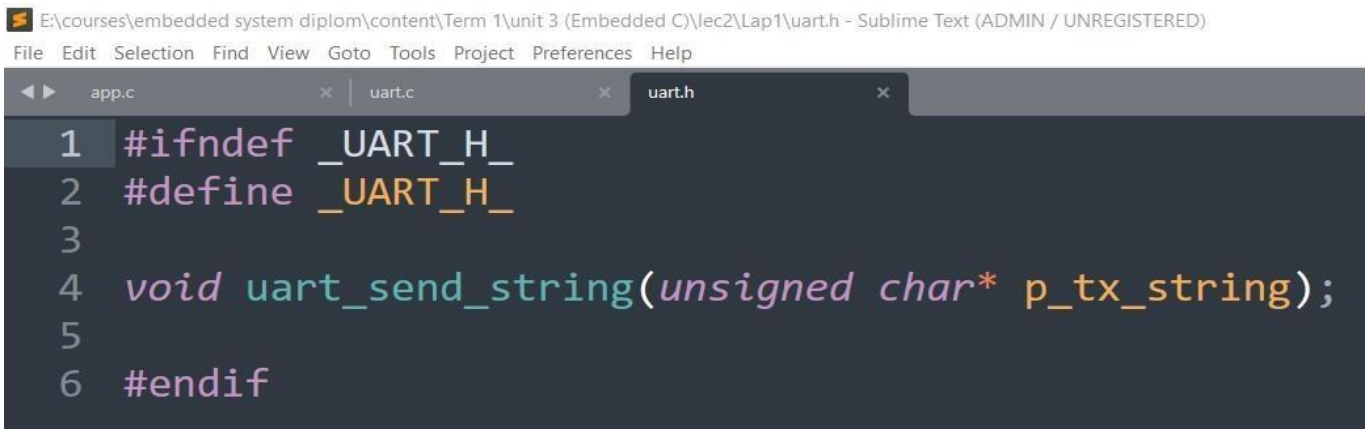
App.c



Uart.c



Uart.h



Startup.s

E:\courses\embedded system diplom\content\Term 1\unit 3 (Embedded C)\lec

File Edit Selection Find View Goto Tools Project Preferences Help

startup.s

```
1  .global reset
2  reset:
3       ldr sp, =stack_top
4       bl main
5  stop: b stop
```

Linker_script.ld

E:\courses\embedded system diplom\content\Term 1\unit 3 (Embedded C)\lec2\Lap1\linker_script.ld - Sublime T

File Edit Selection Find View Goto Tools Project Preferences Help

linker_script.ld

```
1  ENTRY(reset)
2
3  MEMORY
4  {
5      Mem (rwx) : ORIGIN = 0x00000000, LENGTH = 64M
6  }
7  SECTIONS
8  {
9      . = 0x10000;
10     .startup . :
11     {
12         startup.o(.text)
13     }>Mem
14     .text :
15     {
16         *(.text) *(.rodata)
17     }>Mem
18     .data :
19     {
20         *(.data)
21     }>Mem
22     .bss :
23     {
24         *(.bss)  *(.common)
25     }>Mem
26
27     . = . + 0x1000;
28     stack_top = . ;
29 }
```

2-get obj_files from uart.c and app.c

App.o

```
arm-none-eabi-gcc.exe -c -mcpu=arm926ej-s –I app.c –o app.o
```

uart.o

```
arm-none-eabi-gcc.exe -c -mcpu=arm926ej-s –I uart.c -o uart.o
```

startup.o

```
arm-none-eabi-as.exe -mcpu=arm926ej-s startup.s –o startup.o
```

3-sectons for obj_files

App.o

```
$ arm-none-eabi-objdump.exe -h app.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         00000064  00000000  00000000  0000004c  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  000000b0  2**0
                  ALLOC
  3 .comment      00000012  00000000  00000000  000000b0  2**0
                  CONTENTS, READONLY
  4 .ARM.attributes 00000032  00000000  00000000  000000c2  2**0
                  CONTENTS, READONLY
```

Uart.o

```
$ arm-none-eabi-objdump.exe -h 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
```

Startup.o

```
$ arm-none-eabi-objdump.exe -h 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-symbol tables for uart.o , app.o & startup.o

```
$ arm-none-eabi-nm.exe app.o
00000000 T main
00000000 D string_buffer
         U uart_send_string

AMA@DESKTOP-DDNBIPK MINGW64 /e/cou
diplom/content/Term 1/unit 3 (Embe
$ arm-none-eabi-nm.exe uart.o
00000000 T uart_send_string

AMA@DESKTOP-DDNBIPK MINGW64 /e/cou
diplom/content/Term 1/unit 3 (Embe
$ arm-none-eabi-nm.exe startup.o
         U main
00000000 T reset
         U stack_top
00000008 t stop
```

5-get executable_file (App.elf) and map_file

arm-none-eabi-ld.exe -T Linker_Script.ld -Map=prog.map startup.o App.o Uart.o -o prog.elf

6-sections for prog.elf

```
$ arm-none-eabi-objdump.exe -h prog.elf

prog.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         00000068  00010010  00010010  00008010  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  2 .data         00000064  00010078  00010078  00008078  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  3 .ARM.attributes 0000002e  00000000  00000000  000080dc  2**0
                  CONTENTS, READONLY
  4 .comment      00000011  00000000  00000000  0000810a  2**0
                  CONTENTS, READONLY
```

7-sembol table for prog.elf

```
$ arm-none-eabi-nm.exe prog.elf
00010010 T main
00010000 T reset
000110dc D stack_top
00010008 t stop
00010078 D string_buffer
00010028 T uart_send_string
```

8-get binary file

arm-none-eabi-objcopy.exe -O binary prog.elf prog.bin

9-burn binary file on board using qemu

```
$ qemu-system-arm.exe -M versatilepb -m 128M -nographic -kernel prog.bin
learn-in-depth:<Ahmed>
```