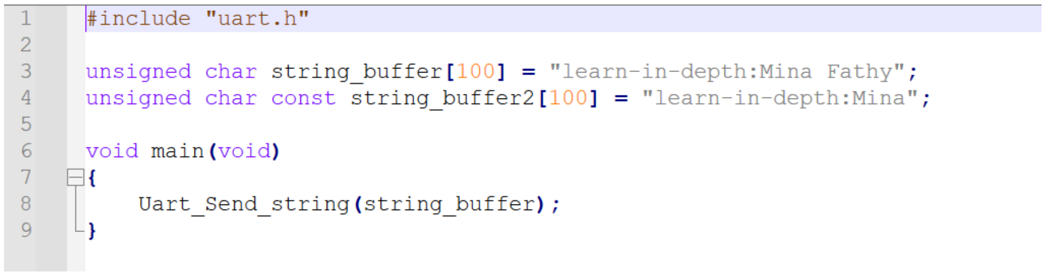
Lab 1 Report

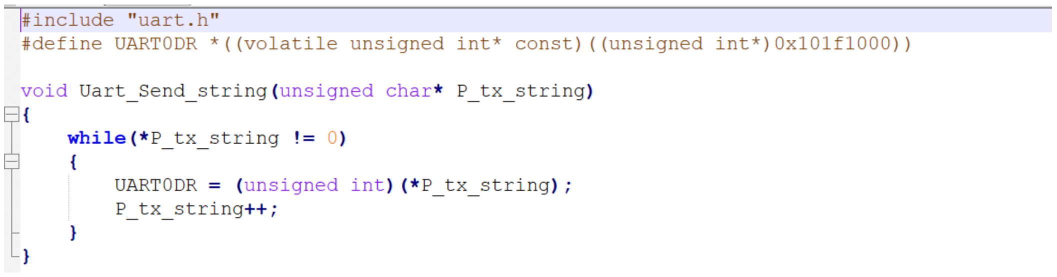
Codes:

- *App.c*



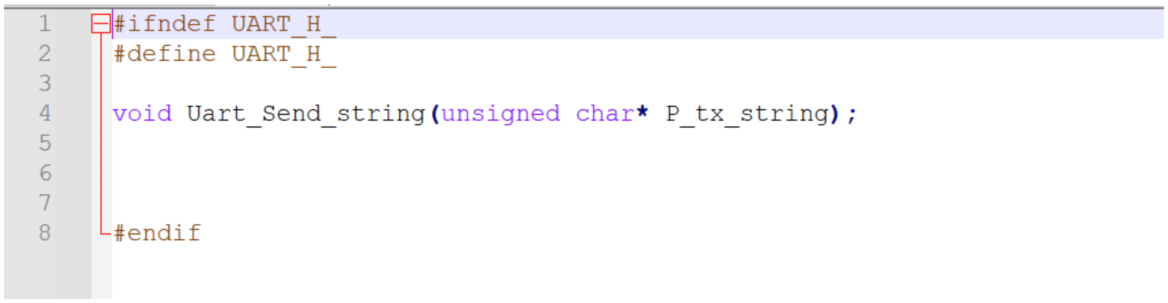
```c
#include "uart.h"

unsigned char string_buffer[100] = "learn-in-depth:Mina Fathy";
unsigned char const string_buffer2[100] = "learn-in-depth:Mina";

void main(void)
{
    Uart_Send_string(string_buffer);
}
```

- *Uart.c*



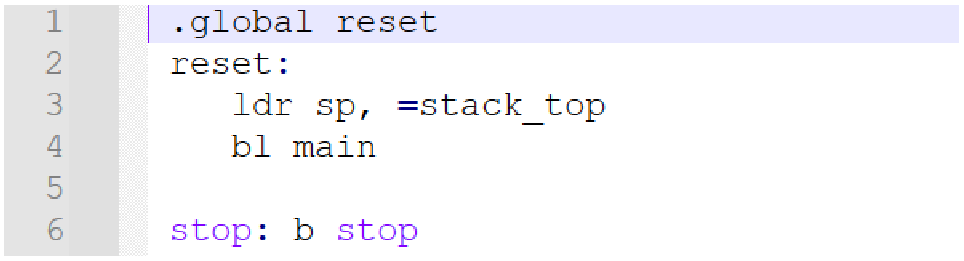
```c
#include "uart.h"
#define UART0DR *((volatile unsigned int* const)((unsigned int*)0x101f1000))

void Uart_Send_string(unsigned char* P_tx_string)
{
    while(*P_tx_string != 0)
    {
        UART0DR = (unsigned int)(*P_tx_string);
        P_tx_string++;
    }
}
```

- *Uart.h*



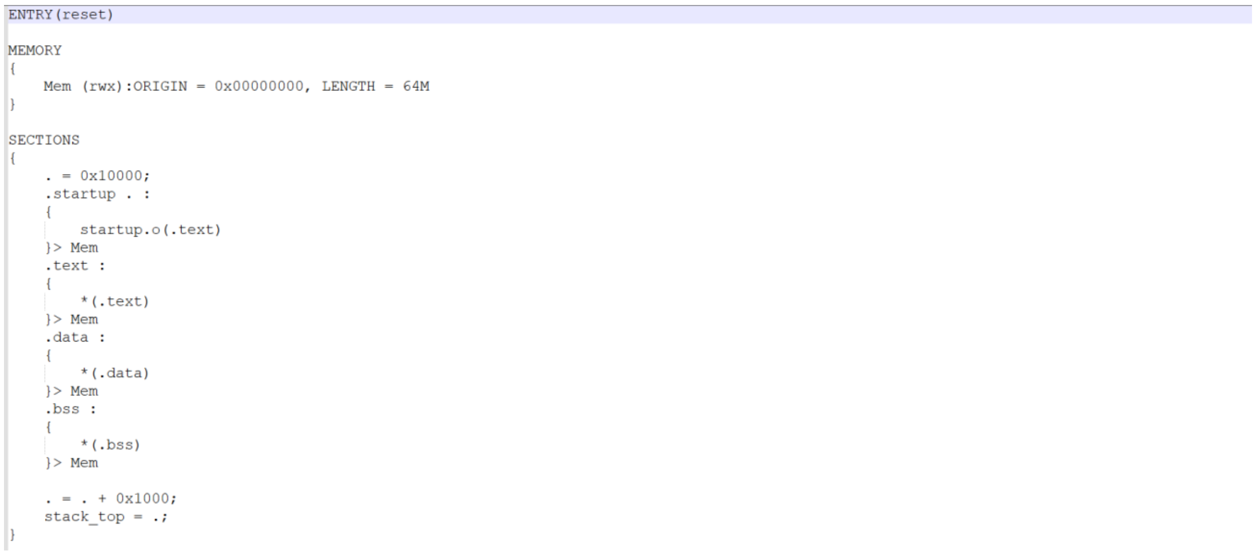
```c
#ifndef UART_H_
#define UART_H_

void Uart_Send_string(unsigned char* P_tx_string);



#endif
```

- *Startup.s*



```asm
.global reset
reset:
    ldr sp, =stack_top
    bl main

stop: b stop
```

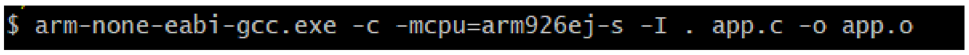
- *Linker_Script*



```
ENTRY(reset)

MEMORY
{
    Mem (rwx):ORIGIN = 0x00000000, LENGTH = 64M
}

SECTIONS
{
    . = 0x10000;
    .startup . :
    {
        startup.o(.text)
    }> Mem
    .text :
    {
        *(.text)
    }> Mem
    .data :
    {
        *(.data)
    }> Mem
    .bss :
    {
        *(.bss)
    }> Mem

    . = . + 0x1000;
    stack_top = .;
}
```

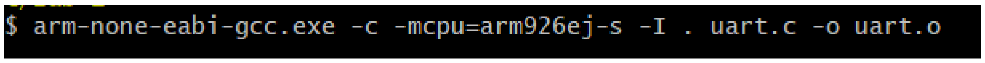
Getting Obj files:

1. App.o



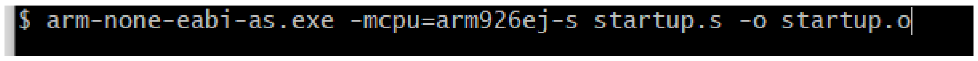
```
$ arm-none-eabi-gcc.exe -c -mcpu=arm926ej-s -I . app.c -o app.o
```

2. Uart.o



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

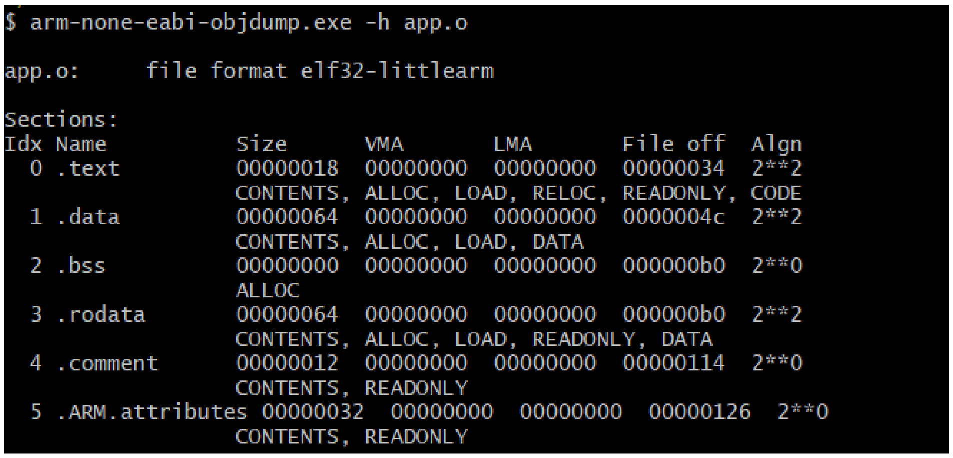
3. Startup.o



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

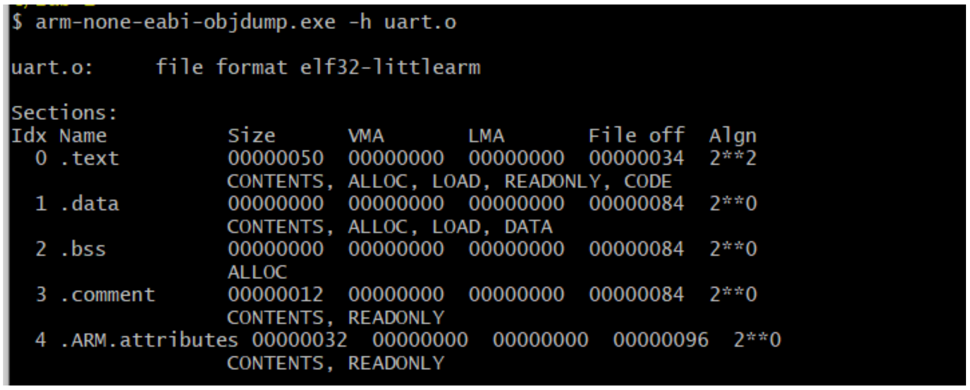
Sections of Object files

- *App.o:*



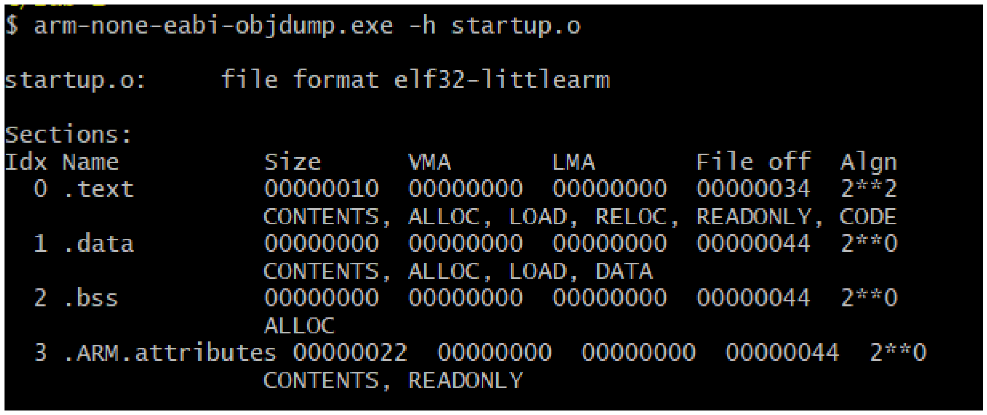
```
$ arm-none-eabi-objdump.exe -h app.o

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 .rodata       00000064  00000000  00000000  000000b0  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, DATA
  4 .comment      00000012  00000000  00000000  00000114  2**0
                  CONTENTS, READONLY
  5 .ARM.attributes 00000032  00000000  00000000  00000126  2**0
                  CONTENTS, READONLY
```

- *Uart.o:*



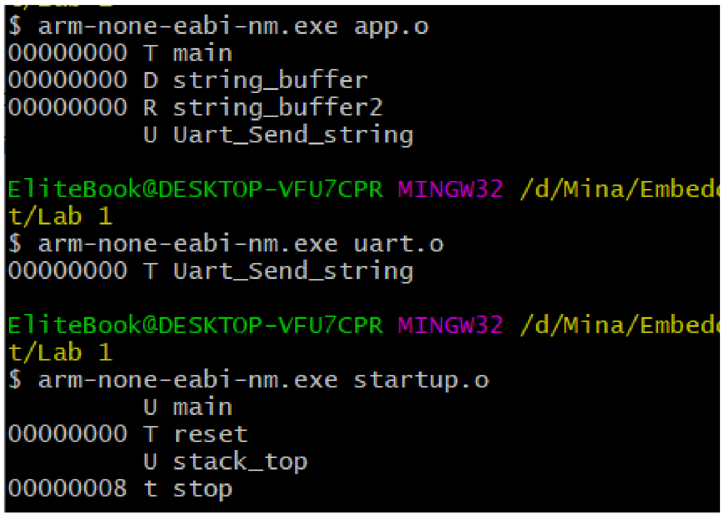
```
$ arm-none-eabi-objdump.exe -h uart.o

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
```

- *Startup.o:*



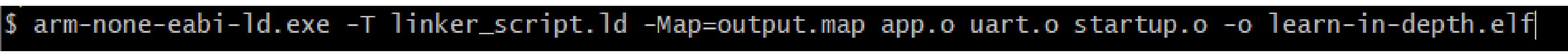
```
$ arm-none-eabi-objdump.exe -h startup.o

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
```

Symbols table of App.o , Uart.o, Startup.o:



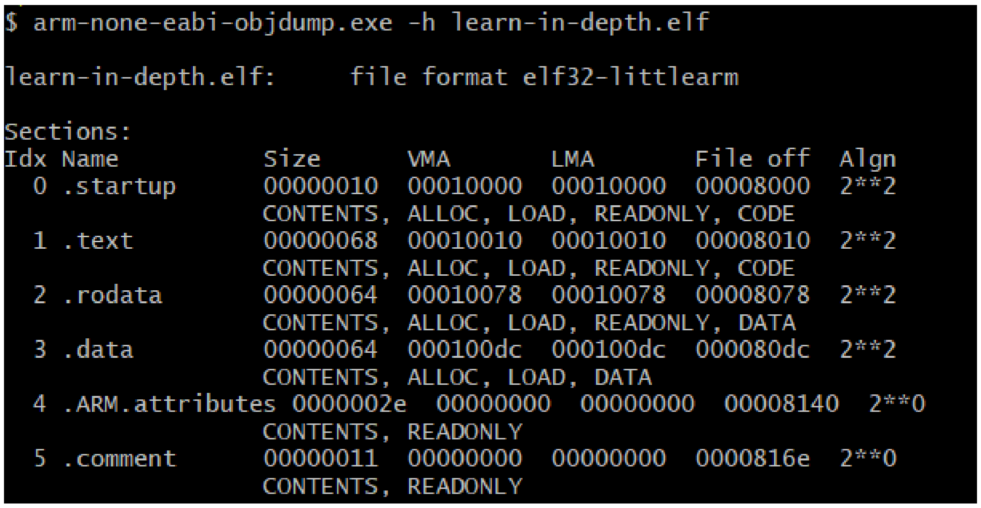
```
$ arm-none-eabi-nm.exe app.o
00000000 T main
00000000 D string_buffer
00000000 R string_buffer2
         U Uart_Send_string

EliteBook@DESKTOP-VFU7CPR MINGW32 /d/Mina/Embede
t/Lab 1
$ arm-none-eabi-nm.exe uart.o
00000000 T Uart_Send_string

EliteBook@DESKTOP-VFU7CPR MINGW32 /d/Mina/Embede
t/Lab 1
$ arm-none-eabi-nm.exe startup.o
         U main
00000000 T reset
         U stack_top
00000008 t stop
```

Getting executable .elf file:



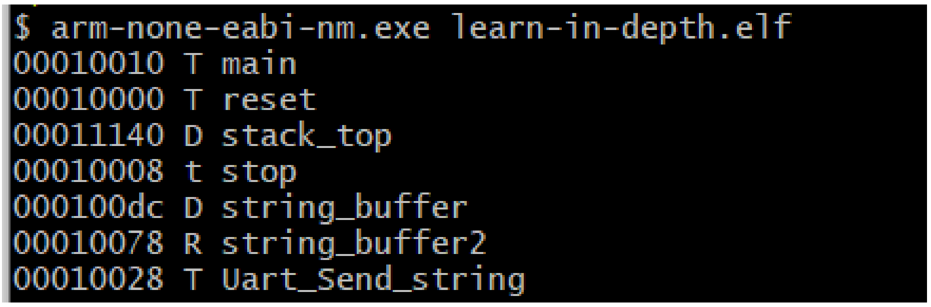
```
$ arm-none-eabi-ld.exe -T linker_script.ld -Map=output.map app.o uart.o startup.o -o learn-in-depth.elf
```

Sections for Learn-in-depth.elf file:



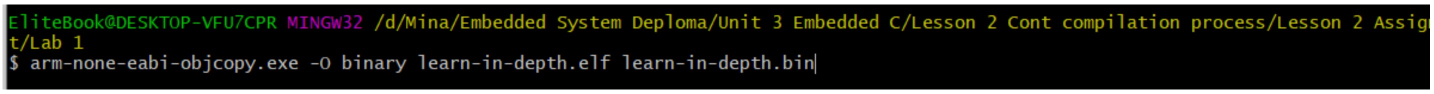
```
$ arm-none-eabi-objdump.exe -h learn-in-depth.elf

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         00000068  00010010  00010010  00008010  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  2 .rodata       00000064  00010078  00010078  00008078  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, DATA
  3 .data         00000064  000100dc  000100dc  000080dc  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  4 .ARM.attributes 0000002e  00000000  00000000  00008140  2**0
                  CONTENTS, READONLY
  5 .comment      00000011  00000000  00000000  0000816e  2**0
                  CONTENTS, READONLY
```

Symbol table of Learn-in-depth.elf file:



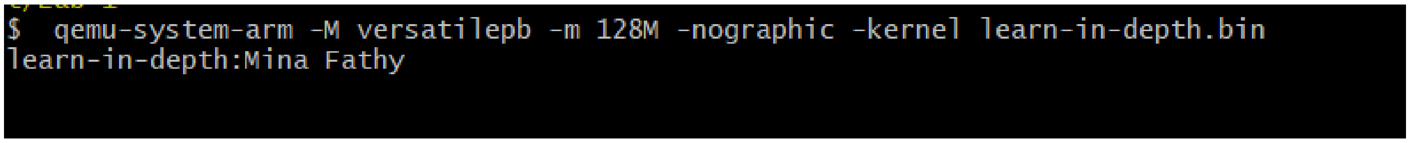
```
$ arm-none-eabi-nm.exe learn-in-depth.elf
00010010 T main
00010000 T reset
00011140 D stack_top
00010008 t stop
000100dc D string_buffer
00010078 R string_buffer2
00010028 T Uart_Send_string
```

Get Binary file to use in burn:



```
EliteBook@DESKTOP-VFU7CPR MINGW32 /d/Mina/Embedded System Deploma/Unit 3 Embedded C/Lesson 2 Cont compilation process/Lesson 2 Assig
t/Lab 1
$ arm-none-eabi-objcopy.exe -O binary learn-in-depth.elf learn-in-depth.bin
```

Burn binary file on board using QEMU:



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