

# ARM VersatilePB Lab Report

Name:	Mohamed belal
Unit:	Embedded C
Assignment:	2
Topic:	create a BareMetal software to send “learn-in-depth:MohamedBelal” using UART.

# 1 Source Code

## 1.1 app.c

```
D:\00_Embedded System learn-in-depth\02_unit 3 Embedded C\Lesson 2 - Embedded C\01_assignment & quiz\LAB_1\app.c - Sublime Text (UNREGISTERED)
app.c x uart.c x uart.h x startup.s x linker-script.ld x
1  /*
2  *   Author : mohamed belal
3  *   Date  : Apr 26, 2023
4  */
5
6  #include "uart.h"
7  unsigned char string_buffer[100] = "learn-in-depth:MohamedBelal";
8  unsigned char const string_buffer2[100] = "learn-in-depth:MohamedBelal";
9  void main()
10 {
11     Uart_Send_String(string_buffer);    /* string_buffer == &string_buffer[0] */
12 }
```

## 1.2 Uart.c

```
D:\00_Embedded System learn-in-depth\02_unit 3 Embedded C\Lesson 2 - Embedded C\01_assignment & quiz\LAB_1\uart.c - Sublime Text (UNREGISTERED)
app.c x uart.c x uart.h x startup.s x linker-script.ld x
4  *   Date  : Apr 26, 2023
5  */
6
7  #include "uart.h"
8  #define UART0DR *((volatile unsigned int* const)((unsigned int*)0x101f1000))
9
10
11 void Uart_Send_String(unsigned char* p_tx_String)
12 {
13     while(*p_tx_String != '\0')    /* loop until end of string*/
14     {
15         UART0DR = (unsigned int)(*p_tx_String); /*transmit char by char from sended string to UART0*/
16         p_tx_String++; /* inc prt to sent another char */
17     }
18 }
```

## 1.Uart.h

```
D:\00_Embedded System learn-in-depth\02_unit 3 Embedded C\Lesson 2 - Embedded C\01_assignment & quiz\LAB_1\uart.h - Sublime Text (UNREGISTERED)
app.c x uart.c x uart.h x startup.s x linker-script.ld x
1
2  #ifndef _UART_H_
3  #define _UART_H_
4
5  void Uart_Send_String(unsigned char* p_tx_String);
6
7  #endif
```

## 1.3 startup.s

```
D:\00_Embedded System learn-in-depth\02_unit 3 Embedded C\lesson 2 - Embedded C\01_assignment & quiz\LAB_1\startup.s - Sublime Text (UNREGISTERED)
app.c x | uart.c x | uart.h x | startup.s x | linker-script.ld x
1 .global reset
2
3 reset:
4     ldr sp, =stack_top
5     bl main
6 stop: b stop
```

## 1.5 linker\_script.ld

```
D:\00_Embedded System learn-in-depth\02_unit 3 Embedded C\lesson 2 - Embedded C\01_assignment & quiz\LAB_1\linker-script.ld - Sublime Text (UNREGISTERED)
app.c x | uart.c x | uart.h x | startup.s x | linker-script.ld x
1
2 ENTRY(reset)
3
4 MEMORY
5 {
6     Mem (rwx) :ORIGIN = 0x00000000 , LENGTH = 64M
7 }
8
9 SECTIONS
10 {
11     . = 0x10000;
12     .startup . :
13     {
14         startup.o(.text)
15     }>Mem
16
17     .text :
18     {
19         *(.text) *(.rodata)
20     }>Mem
21
22     .data :
23     {
24         *(.data)
25     }>Mem
26
27     .bss :
28     {
29         *(.bss)
30     }>Mem
31
32     . = . + 0x1000; /* 1000 == 4KB of Stack Memory */
33     stack_top = . ;
34 }
```

## 2 get obj\_file form App.c Uart .c included Uart.h

### 2.1 App.o

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

### 2.2 Uart.o

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

### 2.3 startup.o

```
Embedded C/01_assignment & quiz/LAB_1  
$ arm-none-eabi-gcc.exe -c -mcpu=arm926ej-s startup.s -o startup.o
```

## 3 To show sections for object\_file

### 3.1 app.o

```
moham@DESKTOP-4ID1J68 MINGW64 /d/00_Embedded System learn-in-depth/02_unit1/02_assignment & quiz/LAB_1  
$ 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
```

### 3.2 uart.o

```
moham@DESKTOP-4ID1J68 MINGW64 /d/00_Embedded System learn-in-depth/02_uni
edded C/01_assignment & quiz/LAB_1
$ 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
```

### 3.2 startup.o

```
moham@DESKTOP-4ID1J68 MINGW64 /d/00_Embedded System learn-in-depth/02_u
edded C/01_assignment & quiz/LAB_1
$ 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 To show symbol table

### 4.1 app.o

```
moham@DESKTOP-4ID1J68 MINGW64 /d/00_Embedded
edded C/01_assignment & quiz/LAB_1
$ arm-none-eabi-nm.exe app.o
00000000 T main
00000000 D string_buffer
00000000 R string_buffer2
                U Uart_Send_String
```

### 4.2 uart.o

```
moham@DESKTOP-4ID1J68 MINGW64 /d/00_Er
edded C/01_assignment & quiz/LAB_1
$ arm-none-eabi-nm.exe uart.o
00000000 T Uart_Send_String
```

### 4.3 startup.o

```
moham@DESKTOP-4ID1J68 MINGW64 /d/00_Embedded S
edded C/01_assignment & quiz/LAB_1
$ arm-none-eabi-nm.exe startup.o
                U main
00000000 T reset
                U stack_top
00000008 t stop
```

## 5 use linker\_script to get executable\_file (App.elf) and map\_file

```
arm-none-eabi-ld.exe -T linker-script.ld -Map=Map_file.map startup.o app.o uart.o
-o learn_in_depth.elf
```

## 6 To show sections for App.elf

```
moham@DESKTOP-4ID1J68 MINGW64 /d/00_Embedded System learn-in-depth/02_unit 3 Embedded C/lesson 2 - Embedded C/01_assignment & quiz/LAB_1
$ 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 .text          000000dc  00000000  00000000  00008000  2**2
   CONTENTS, ALLOC, LOAD, READONLY, CODE
 1 .data          00000064  000000dc  000000dc  000080dc  2**2
   CONTENTS, ALLOC, LOAD, DATA
 2 .ARM.attributes 0000002e  00000000  00000000  00008140  2**0
   CONTENTS, READONLY
 3 .comment        00000011  00000000  00000000  0000816e  2**0
   CONTENTS, READONLY
```

## 7 To show symbol table for App.elf

```
moham@DESKTOP-4ID1J68 MINGW64 /d/00_Embedded System learn-in-depth/02_unit 3 Embedded C/lesson 2 - Embedded C/01_assignment & quiz/LAB_1
$ arm-none-eabi-nm.exe learn_in_depth.elf
00000010 T main
00000000 T reset
00001140 D stack_top
00000008 t stop
000000dc D string_buffer
00000078 T string_buffer2
00000028 T Uart_Send_String
```

## 8 get binary file to use in burn

```
arm-none-eabi-objcopy.exe -O binary learn_in_depth.elf learn_in_depth.bin
```

## 9 burn binary file on board using qemu

```
qemu-system-arm.exe -M versatilepb -m 128M -nographic -kernel learn_in_depth.bin
```

```
moham@DESKTOP-4ID1J68 MINGW64 /d/00_Embedded System learn-in-depth/02_unit 3 Embedded C/lesson 2 - Embedded C/01_assignment & quiz/LAB_1
$ qemu-system-arm.exe -M versatilepb -m 128M -nographic -kernel learn_in_depth.bin
learn-in-depth:~#
```