Lesson 2

App.c

```
$ cat app.c
#include "uart.h"

unsigned char string_buffer [100] = " Learn In Depth : << Joseph Nader Sophy >> ";
unsigned char const string_buffer_2[100] = "to create ro data section";

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

Uart.c

```
$ cat uart.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

```
$ cat uart.h
#ifndef _UART_H_
#define _UART_H_

void uart_send_string(unsigned char *p_tx_string);

#endif
```

Sections

```
$ 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 .debug_info   00000083  00000000  00000000  00000114  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  5 .debug_abbrev 00000061  00000000  00000000  00000197  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_loc    0000002c  00000000  00000000  000001f8  2**0
                  CONTENTS, READONLY, DEBUGGING
  7 .debug_aranges 00000020  00000000  00000000  00000224  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  8 .debug_line   00000035  00000000  00000000  00000244  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  9 .debug_str    000000b0  00000000  00000000  00000279  2**0
                  CONTENTS, READONLY, DEBUGGING
 10 .comment      00000012  00000000  00000000  00000329  2**0
                  CONTENTS, READONLY
 11 .ARM.attributes 00000032  00000000  00000000  0000033b  2**0
                  CONTENTS, READONLY
 12 .debug_frame  0000002c  00000000  00000000  00000370  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
```

Diassemply file from bin

```
# arm-none-eabi-objdump.exe -D app.o > app.s
```

app.c | uart.h | uart.c | app.o | app.s

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


Disassembly of section .text:

00000000 <main>:
   0:   e92d4800    push    {fp, lr}
   4:   e28db004    add fp, sp, #4
   8:   e59f0004    ldr r0, [pc, #4]    ; 14 <main+0x14>
   c:   ebfffffe    bl  0 <uart_send_string>
  10:   e8bd8800    pop {fp, pc}
  14:   00000000    andeq   r0, r0, r0

Disassembly of section .data:

00000000 <string_buffer>:
   0:   61654c20    cmnvs   r5, r0, lsr #24
   4:   49206e72    stmdbmi r0!, {r1, r4, r5, r6, r9, sl, fp, sp, lr}
   8:   6544206e    strbvs  r2, [r4, #-110] ; 0x6e
   c:   20687470    rsbcs   r7, r8, r0, ror r4
  10:   3c3c203a    ldccc   0, cr2, [ip], #-232 ; 0xffffff18
  14:   736f4a20    cmnvc   pc, #32, 20 ; 0x20000
  18:   20687065    rsbcs   r7, r8, r5, rrx
  1c:   6564614e    strbvs  r6, [r4, #-334]!    ; 0x14e
  20:   6f532072    svcvs   0x00532072
  24:   20796870    rsbscs  r6, r9, r0, ror r8
  28:   00203e3e    eoreq   r3, r0, lr, lsr lr
    ...

Disassembly of section .rodata:

00000000 <string_buffer_2>:
   0:   63206f74    teqvs   r0, #116, 30    ; 0x1d0
   4:   74616572    strbtvc r6, [r1], #-1394    ; 0x572
   8:   6f722065    svcvs   0x00722065
   c:   74616420    strbtvc r6, [r1], #-1056    ; 0x420
  10:   65732061    ldrbvs  r2, [r3, #-97]! ; 0x61
```

Uart API

Display the full content of all sections

MINGW64:/c/Users/hp/Desktop/Diploma_Repo/Masteering_In_Embeded_System/Unit_3_Embedded_C/Assignment_Lesson_2

```
hp@Joseph MINGW64 ~/Desktop/Diploma_Repo/Masteering_In_Embeded_System/Unit_3_Embedded_C/Assig
nment_Lesson_2 (main)
$ arm-none-eabi-objdump.exe -s app.o

app.o:     file format elf32-littlearm

Contents of section .text:
 0000 00482de9 04b08de2 04009fe5 feffffeb  .H-.............
 0010 0088bde8 00000000                    ........
Contents of section .data:
 0000 204c6561 726e2049 6e204465 70746820   Learn In Depth
 0010 3a203c3c 204a6f73 65706820 4e616465  : << Joseph Nade
 0020 7220536f 70687920 3e3e2000 00000000  r Sophy >> .....
 0030 00000000 00000000 00000000 00000000  ................
 0040 00000000 00000000 00000000 00000000  ................
 0050 00000000 00000000 00000000 00000000  ................
 0060 00000000                             ....
Contents of section .rodata:
 0000 746f2063 72656174 6520726f 20646174  to create ro dat
 0010 61207365 6374696f 6e000000 00000000  a section.......
 0020 00000000 00000000 00000000 00000000  ................
 0030 00000000 00000000 00000000 00000000  ................
 0040 00000000 00000000 00000000 00000000  ................
 0050 00000000 00000000 00000000 00000000  ................
 0060 00000000                             ....
Contents of section .debug_info:
 0000 7f000000 02000000 00000401 17000000  ................
 0010 01970000 00330000 00000000 00180000  .....3..........
 0020 00000000 000201ab 00000001 06010000  ................
 0030 00001800 00000000 00000103 52000000  ............R...
 0040 4b000000 044b0000 00630005 04070e00  K....K...c......
 0050 00000501 08000000 00069d00 00000103  ................
 0060 3b000000 01050300 00000006 23000000  ;...........#...
 0070 01047d00 00000105 03000000 00073b00  ..}...........;.
 0080 000000                               ...
Contents of section .debug_abbrev:
 0000 01110125 0e130b03 0e1b0e11 01120110  ...%............
 0010 06000002 2e003f0c 030e3a0b 3b0b270c  ......?...:.;.'.
 0020 11011201 40069642 0c000003 01014913  ....@..B......I.
 0030 01130000 04210049 132f0b00 00052400  .....!.I./....$.
 0040 0b0b3e0b 030e0000 06340003 0e3a0b3b  ..>......4...:.;
 0050 0b49133f 0c020a00 00072600 49130000  .I.?......&.I...
 0060 00                                   .
Contents of section .debug_loc:
 0000 00000000 04000000 02007d00 04000000  ..........}.....
 0010 08000000 02007d08 08000000 18000000  ......}.........
 0020 02007b04 00000000 00000000           ..{.........
Contents of section .debug_aranges:
 0000 1c000000 02000000 00000400 00000000  ................
 0010 00000000 18000000 00000000 00000000  ................
Contents of section .debug_line:
 0000 31000000 02001c00 00000201 fb0e0d00  1...............
 0010 01010101 00000001 00000100 6170702e  ............app.
 0020 63000000 00000005 02000000 00184b4b  c............KK
 0030 02040001 01                          .....
Contents of section .debug_str:
 0000 756e7369 676e6564 20636861 72007369  unsigned char.si
 0010 7a657479 70650047 4e552043 20342e37  zetype.GNU C 4.7
```

```
 0020 2e320073 7472696e 675f6275 66666572  .2.string_buffer
 0030 5f320043 3a5c5573 6572735c 68705c44  _2.C:\Users\hp\D
 0040 65736b74 6f705c44 69706c6f 6d615f52  esktop\Diploma_R
 0050 65706f5c 4d617374 65657269 6e675f49  epo\Masteering_I
 0060 6e5f456d 62656465 645f5379 7374656d  n_Embeded_System
 0070 5c556e69 745f335f 456d6265 64646564  \Unit_3_Embedded
 0080 5f435c41 73736967 6e6d656e 745f4c65  _C\Assignment_Le
 0090 73736f6e 5f320061 70702e63 00737472  sson_2.app.c.str
 00a0 696e675f 62756666 6572006d 61696e00  ing_buffer.main.
Contents of section .comment:
 0000 00474343 3a202847 4e552920 342e372e  .GCC: (GNU) 4.7.
 0010 3200                                 2.
Contents of section .ARM.attributes:
 0000 41310000 00616561 62690001 27000000  A1...aeabi..'...
 0010 0541524d 39323645 4a2d5300 06050801  .ARM926EJ-S.....
 0020 09011204 14011501 17031801 19011a01  ................
 0030 1e06                                 ..
Contents of section .debug_frame:
 0000 0c000000 ffffffff 0100027c 0e0c0d00  ...........|....
 0010 18000000 00000000 00000000 18000000  ................
 0020 420e088b 028e0142 0c0b0400           B......B....
```

Startup.s :

startup.s | app.c | uart.h | uart.c

```
.global reset

reset:
    ldr sp, =stack_top
    bl main
stop:
    b stop
```

Compile and analyze it

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

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

```
hp@Joseph MINGW64 ~/Desktop/Diploma_Repo/Masteering_In_Embeded_Syste
nment_Lesson_2 (main)
$ 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
```

Linker_Script :

linker_script.ld | startup.s | app.c | uart.h | uart.c | app.o

```
ENTRY(reset)
MEMORY
{
Mem (rwx) : ORIGIN = 0x00000000, LENGTH =64M
}
SECTIONS
{
    . = 0X10000 ;
    .startup_joseph :
    {
    startup.o(.text)
    }> Mem
    .text_Nader :
    {
    *(.text) *(.rodata)
    }> Mem
    .data_Sophy :
    {
    *(.data)
    }> Mem
    .bss_unintialized_global :
    {
    *(.bss) *(COMMON)
    }> Mem
    . = . + 0x1000 ;/*4 kb of stack */
    stack_top = . ;
}
```

To read the symbols you can use nm cross tool chain bin utility

```
nment_Lesson_2 (main)
$ arm-none-eabi-nm.exe app.o
00000000 T main
00000000 D string_buffer
00000000 R string_buffer_2
         U uart_send_string
```

```
nment_Lesson_2 (main)
$ arm-none-eabi-nm.exe uart.o
00000000 T uart_send_string
```

Let us now to linking all the objects :

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

Analyze the executable file

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

```
nment_Lesson_2 (main)
$ 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_buffer_2
00000028 T uart_send_string
```

Sections

```
nment_Lesson_2 (main)
$ 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_joseph 00000010  00000000  00000000  00008000  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .text_Nader   000000cc  00000010  00000010  00008010  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  2 .data_Sophy   00000064  000000dc  000000dc  000080dc  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  3 .ARM.attributes 0000002e  00000000  00000000  00008140  2**0
                  CONTENTS, READONLY
  4 .comment      00000011  00000000  00000000  0000816e  2**0
                  CONTENTS, READONLY
  5 .debug_info   000000df  00000000  00000000  0000817f  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_abbrev 000000b2  00000000  00000000  0000825e  2**0
                  CONTENTS, READONLY, DEBUGGING
  7 .debug_loc    00000058  00000000  00000000  00008310  2**0
                  CONTENTS, READONLY, DEBUGGING
  8 .debug_aranges 00000040  00000000  00000000  00008368  2**0
                  CONTENTS, READONLY, DEBUGGING
  9 .debug_line   00000072  00000000  00000000  000083a8  2**0
                  CONTENTS, READONLY, DEBUGGING
 10 .debug_str    000000d4  00000000  00000000  0000841a  2**0
                  CONTENTS, READONLY, DEBUGGING
 11 .debug_frame  00000054  00000000  00000000  000084f0  2**2
                  CONTENTS, READONLY, DEBUGGING
```

Output.map

```
Memory Configuration

Name             Origin             Length             Attributes
Mem              0x00000000         0x04000000         xrw
*default*        0x00000000         0xffffffff

Linker script and memory map

                0x00010000                 . = 0x10000

.startup_joseph
                0x00000000       0x10
 startup.o(.text)
 .text          0x00000000       0x10 startup.o
                0x00000000                reset

.text_Nader     0x00000010       0xcc
 *(.text)
 .text          0x00000010       0x18 app.o
                0x00000010                main
 .text          0x00000028       0x50 uart.o
                0x00000028                uart_send_string
 *(.rodata)
 .rodata        0x00000078       0x64 app.o
                0x00000078                string_buffer_2

.glue_7         0x000000dc        0x0
 .glue_7        0x00000000        0x0 linker stubs

.glue_7t        0x000000dc        0x0
 .glue_7t       0x00000000        0x0 linker stubs

.vfp11_veneer   0x000000dc        0x0
 .vfp11_veneer  0x00000000        0x0 linker stubs

.v4_bx          0x000000dc        0x0
 .v4_bx         0x00000000        0x0 linker stubs
```

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

```
$ qemu-system-arm -M versatilepb -m 128M -nographic -kernel learnoo.bin
```