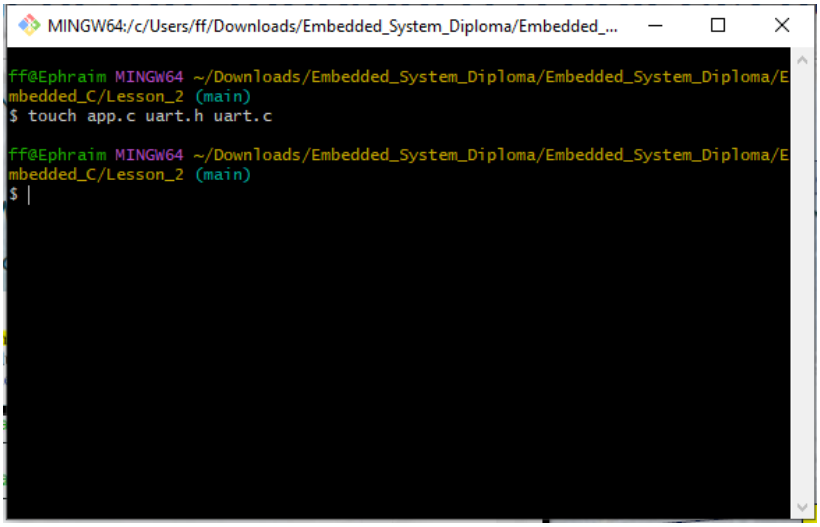
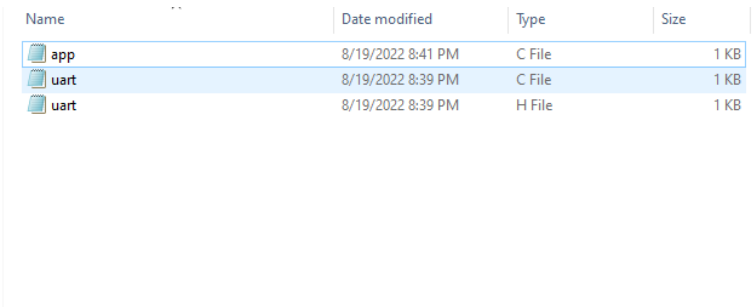
Report for
LAB Of Unit 3 Lesson 2

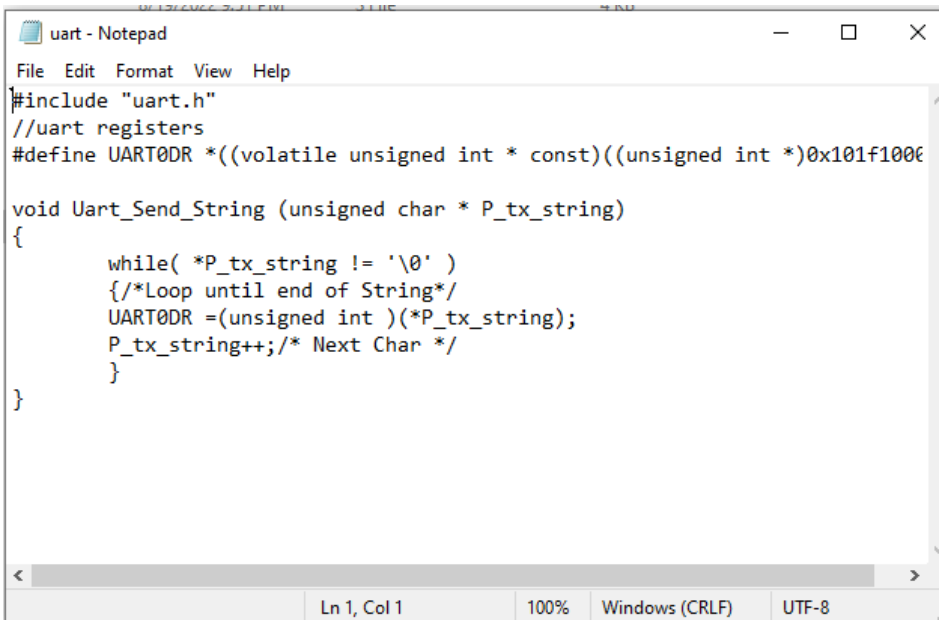
Using command at git Bash to create files :





Name	Date modified	Type	Size
app	8/19/2022 8:41 PM	C File	1 KB
uart	8/19/2022 8:39 PM	C File	1 KB
uart	8/19/2022 8:39 PM	H File	1 KB

Uart.c file :




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

void Uart_Send_String (unsigned char * P_tx_string)
{
        while( *P_tx_string != '\0' )
        {/*Loop until end of String*/
        UART0DR =(unsigned int )(*P_tx_string);
        P_tx_string++;/* Next Char */
        }
}
```

Uart.h file :

uart - Notepad

File Edit Format View Help

```
#ifndef _UART_H_
#define _UART_H_

//UART APIs
void Uart_Send_String (unsigned char * P_tx_string);

#endif
```

Ln 1, Col 1 | 100% | Windows (CRLF) | UTF-8

App.c file :

app - Notepad

File Edit Format View Help

```
#include "uart.h"

unsigned char string_buffer[100]="learn-in-depth:<Ephraim>";
unsigned char const string_buffer_2[100]="Tp Create a rodata Section";
void main (void)
{
        Uart_Send_String (string_buffer);
}
```

Creating object files with debugging Information :

```
MINGW64:/c/Users/ff/Downloads/Embedded_System_Diploma/Embedded_...

ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/E
mbedded_C/Lesson_2 (main)
$ arm-none-eabi-gcc.exe -c -g -I . -mcpu=arm926ej-s app.c -o app.o

ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/E
mbedded_C/Lesson_2 (main)
$ arm-none-eabi-gcc.exe -c -g -I . -mcpu=arm926ej-s uart.c -o uart.o

ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/E
mbedded_C/Lesson_2 (main)
$ ls *.o
app.o  uart.o
```

Name	Date modified	Type	Size
app	8/19/2022 8:41 PM	C File	1 KB
app.o	8/19/2022 8:54 PM	O File	3 KB
uart	8/19/2022 8:39 PM	C File	1 KB
uart	8/19/2022 8:39 PM	H File	1 KB
uart.o	8/19/2022 8:55 PM	O File	3 KB

To watch obj file section :

```
ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/E
mbedded_C/Lesson_2 (main)
$ 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         0000001c  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000064  00000000  00000000  00000050  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  000000b4  2**0
                  ALLOC
  3 .debug_info   00000066  00000000  00000000  000000b4  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  4 .debug_abbrev 0000005a  00000000  00000000  0000011a  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_aranges 00000020  00000000  00000000  00000174  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  6 .debug_line   00000035  00000000  00000000  00000194  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  7 .debug_str    000000eb  00000000  00000000  000001c9  2**0
                  CONTENTS, READONLY, DEBUGGING
  8 .comment      0000007f  00000000  00000000  000002b4  2**0
                  CONTENTS, READONLY
  9 .debug_frame  0000002c  00000000  00000000  00000334  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
 10 .ARM.attributes 00000032  00000000  00000000  00000360  2**0
                  CONTENTS, READONLY

ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/E
mbedded_C/Lesson_2 (main)
$ |
```

Create object file without debugging section And Watch The Output Sections :

```
ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/E
mbedded_C/Lesson_2 (main)
$ arm-none-eabi-gcc.exe -c -I . -mcpu=arm926ej-s app.c -o app.o

ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/E
mbedded_C/Lesson_2 (main)
$ 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         0000001c  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000064  00000000  00000000  00000050  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  000000b4  2**0
                  ALLOC
  3 .comment      0000007f  00000000  00000000  000000b4  2**0
                  CONTENTS, READONLY
  4 .ARM.attributes 00000032  00000000  00000000  00000133  2**0
                  CONTENTS, READONLY

ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/E
mbedded_C/Lesson_2 (main)
$
```

Create .rodata section :

app - Notepad

File Edit Format View Help

```
#include "uart.h"

unsigned char string_buffer[100]="learn-in-depth:<Ephraim>";
unsigned char const string_buffer_2[100]="Tp Create a rodata Section";
void main (void)
{
        Uart_Send_String (string_buffer);
}
```

Ln 4, Col 36 | 100% | Windows (CRLF) | UTF-8

```
ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/E
mbedded_C/Lesson_2 (main)
$ arm-none-eabi-gcc.exe -c -I . -mcpu=arm926ej-s app.c -o app.o

ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/E
mbedded_C/Lesson_2 (main)
$ 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         0000001c  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000064  00000000  00000000  00000050  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  000000b4  2**0
                  ALLOC
  3 .rodata       00000064  00000000  00000000  000000b4  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, DATA
  4 .comment      0000007f  00000000  00000000  00000118  2**0
                  CONTENTS, READONLY
  5 .ARM.attributes 00000032  00000000  00000000  00000197  2**0
                  CONTENTS, READONLY

ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/E
mbedded_C/Lesson_2 (main)
$ |
```

To generate assembly file from object file :

```
ff@Ephraim MINGW64 ~/Downloads/Embedded_System_
mbedded_C/Lesson_2 (main)
$ arm-none-eabi-objdump.exe -D app.o > app.s
```

app - Notepad

File Edit Format View Help

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


Disassembly of section .text:

00000000 <main>:
   0:   e92d4800        push    {fp, lr}
   4:   e28db004        add     fp, sp, #4
   8:   e59f0008        ldr     r0, [pc, #8]    ; 18 <main+0x18>
   c:   ebfffffe        bl      0 <Uart_Send_String>
  10:   e1a00000        nop                     ; (mov r0, r0)
  14:   e8bd8800        pop     {fp, pc}
  18:   00000000        andeq   r0, r0, r0

Disassembly of section .data:

00000000 <string_buffer>:
   0:   7261656c        rsbvc   r6, r1, #108, 10        ; 0x1b000000
   4:   6e692d6e        cdpvs   13, 6, cr2, cr9, cr14, {3}
   8:   7065642d        rsbvc   r6, r5, sp, lsr #8
   c:   3c3a6874        ldccc   8, cr6, [sl], #-464     ; 0xfffffe30
  10:   72687045        rsbvc   r7, r8, #69     ; 0x45
  14:   3e6d6961        vnmulcc.f16     s13, s26, s3    ; <UNPREDICTABLE>
        ...

Disassembly of section .rodata:
```

Creating startup file :

```
MINGW64:/c/Users/ff/Downloads/Embec

ff@Ephraim MINGW64 ~/Downloads/Embe
$ touch startup.s

ff@Ephraim MINGW64 ~/Downloads/Embe
mbedded_C/Lesson_2 (main)
$ |
```

```
startup - Notepad
File  Edit  Format  View  Help
.globl reset
reset:
        ldr sp, =0x00011000
        bl main
stop:   b stop
```

Creating .o file and watch object file sections of startup file :

```
MINGW64:/c/Users/ff/Downloads/Embedded_System_Diploma/Embedded_...

$ arm-none-eabi-as.exe -mcpu=arm926ej-s startup.s -o startup.o
startup.s: Assembler messages:
startup.s: Warning: end of file not at end of a line; newline inserted

ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diplo
mbedded_C/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         0000000c  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000000  00000000  00000000  00000040  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  00000040  2**0
                  ALLOC
  3 .ARM.attributes 00000022  00000000  00000000  00000040  2**0
                  CONTENTS, READONLY

ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diplo
mbedded_C/Lesson_2 (main)
$ |
```

Creating linker script file :

linker_script - Notepad

File Edit Format View Help

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

SECTIONS
{
        . =0x10000;
        .startup . :
        {
                startup.o(.text)
        }>Mem
        .text :
        {
                *(.text) *(.rodata)
        }>Mem
        .data :
        {
                *(.data)
        }>Mem
        .bss :
        {
                *(.bss) *(COMMON)
        }>Mem
        . = . + 0x1000; /*4KB of Stack Memory*/
        stack_top = . ;
}
```

Watching symbols of app.o and uart.o before linking with relocatable addresses :

MINGW64:/c/Users/ff/Downloads/Embedded_System_Diploma/Embedded_...

```
ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diplom
mbedded_C/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

ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diplom
mbedded_C/Lesson_2 (main)
$ arm-none-eabi-nm.exe uart.o
00000000 T Uart_Send_String

ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diplom
mbedded_C/Lesson_2 (main)
$ |
```

New Startup Code :

startup - Notepad

File Edit Format View Help

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

Ln 1, Col 1 | 100% | Windows (CRLF) | UTF-8

MINGW64:/c/Users/ff/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/E...

```
ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/E
mbedded_C/Lesson_2 (main)
$ arm-none-eabi-as.exe -mcpu=arm926ej-s startup.s -o startup.o

ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/E
mbedded_C/Lesson_2 (main)
$
```

linking file and generate learn in depth.elf file

```
ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/Embedded_C/Lesson_2 (main)
$ arm-none-eabi-ld.exe -T linker_script.ld startup.o app.o uart.o -o learn-in-depth.elf -Map=Map_file.map

ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/Embedded_C/Lesson_2 (main)
$
```

watching symbols in learn-in-depth.elf File :

```
MINGW64:/c/Users/ff/Downloads/Embedded_System_Diploma/Embedded_System_...

ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/Embedde
d_C/Lesson_2 (main)
$ arm-none-eabi-nm.exe learn-in-depth.elf
00010010 T main
00010000 T reset
00011148 D stack_top
00010008 t stop
000100e4 D string_buffer
00010080 T string_buffer_2
0001002c T Uart_Send_String

ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/Embedde
d_C/Lesson_2 (main)
$
```

Watching sections in learn-in-depth.elf file with debugging information :

```
MINGW64:/c/Users/ff/Downloads/Embedded_System_Diploma/Embedded_System_...

ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/Embedde
d_C/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      00000010  00010000  00010000  00010000  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .text         000000d4  00010010  00010010  00010010  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  2 .data         00000064  000100e4  000100e4  000100e4  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  3 .ARM.attributes 0000002e  00000000  00000000  00010148  2**0
                  CONTENTS, READONLY
  4 .comment      0000007e  00000000  00000000  00010176  2**0
                  CONTENTS, READONLY
  5 .debug_info   00000057  00000000  00000000  000101f4  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_abbrev 00000051  00000000  00000000  0001024b  2**0
                  CONTENTS, READONLY, DEBUGGING
  7 .debug_aranges 00000020  00000000  00000000  0001029c  2**0
                  CONTENTS, READONLY, DEBUGGING
  8 .debug_line   00000039  00000000  00000000  000102bc  2**0
                  CONTENTS, READONLY, DEBUGGING
  9 .debug_str    000000e9  00000000  00000000  000102f5  2**0
                  CONTENTS, READONLY, DEBUGGING
 10 .debug_frame  00000030  00000000  00000000  000103e0  2**2
                  CONTENTS, READONLY, DEBUGGING

ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/Embedde
d_C/Lesson_2 (main)
$
```

generating binary file from learn-in-depth.elf File

```
ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/Embedded_C/Le
$ arm-none-eabi-objcopy.exe -O binary learn-in-depth.elf learn-in-depth.bin

ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/Embedded_C/Le
$ |
```

Entry Point of Startup File :

```
  Class:                             ELF32
  Data:                              2's complement, little endian
  Version:                           1 (current)
  OS/ABI:                            UNIX - System V
  ABI Version:                       0
  Type:                              EXEC (Executable file)
  Machine:                           ARM
  Version:                           0x1
  Entry point address:               0x10000
  Start of program headers:          52 (bytes into file)
  Start of section headers:          67312 (bytes into file)
  Flags:                             0x5000200, Version5 EABI, soft-floa
  Size of this header:               52 (bytes)
  Size of program headers:           32 (bytes)
  Number of program headers:         1
  Size of section headers:           40 (bytes)
  Number of section headers:         15
  Section header string table index: 14

Section Headers:
  [Nr] Name              Type            Addr     Off    Size   ES Flg L
  [ 0]                   NULL            00000000 000000 000000 00
  [ 1] .startup          PROGBITS        00010000 010000 000010 00  AX
  [ 2] .text             PROGBITS        00010010 010010 0000d4 00  AX
  [ 3] .data             PROGBITS        000100e4 0100e4 000064 00  WA
```

Output :

```
MINGW64:/c/Users/ff/Downloads/Embedded_System_Diploma/Embedded_...

ff@Ephraim MINGW64 ~/Downloads/Embedded_System_Diploma/Embedded_System_Diploma/E
mbedded_C/Lesson_2 (main)
$ qemu-system-arm -M versatilepb -m 128M -nographic -kernel learn-in-depth.bin
learn-in-depth:<Ephraim>
```