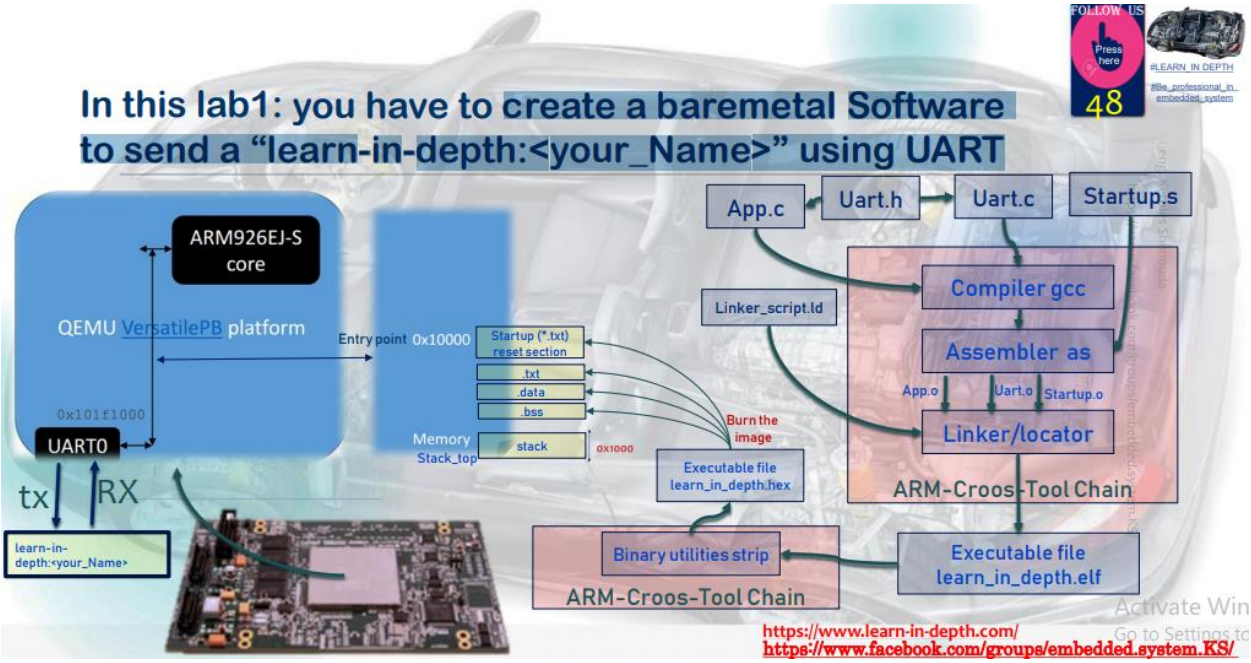
Lab 1

Description:

Create a bare-metal Software to send a “learn-in-depth: <<your name>>” using UART of the ARM Versatile PB board.



Files Created:

- uart.h
- uart.c
- app.c
- startup.s
- linker_script.ld

Analysis Files:

- uart.o
- app.o
- startup.o
- Map_file.map

Executable Files:

- learn-in-depth.elf
- learn-in-depth.bin

Git Commands Used In Compilation Process:

To get the object files:

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

To link the object files together using linker_script and get the .elf and .map files:

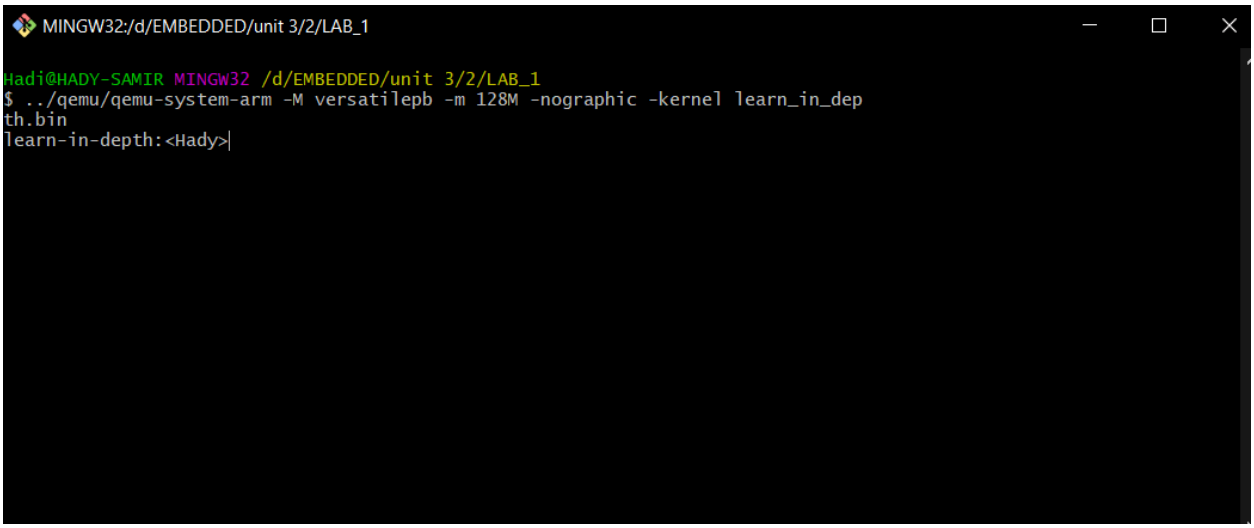
- `$ arm-none-eabi-ld.exe -T linker_script.ld app.o uart.o startup.o -o learn-in-depth.elf -Map=Map_file.map`

To get the bin file:

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

To run the program in the QEMU Simulator (“VersatilePB physical Board”):

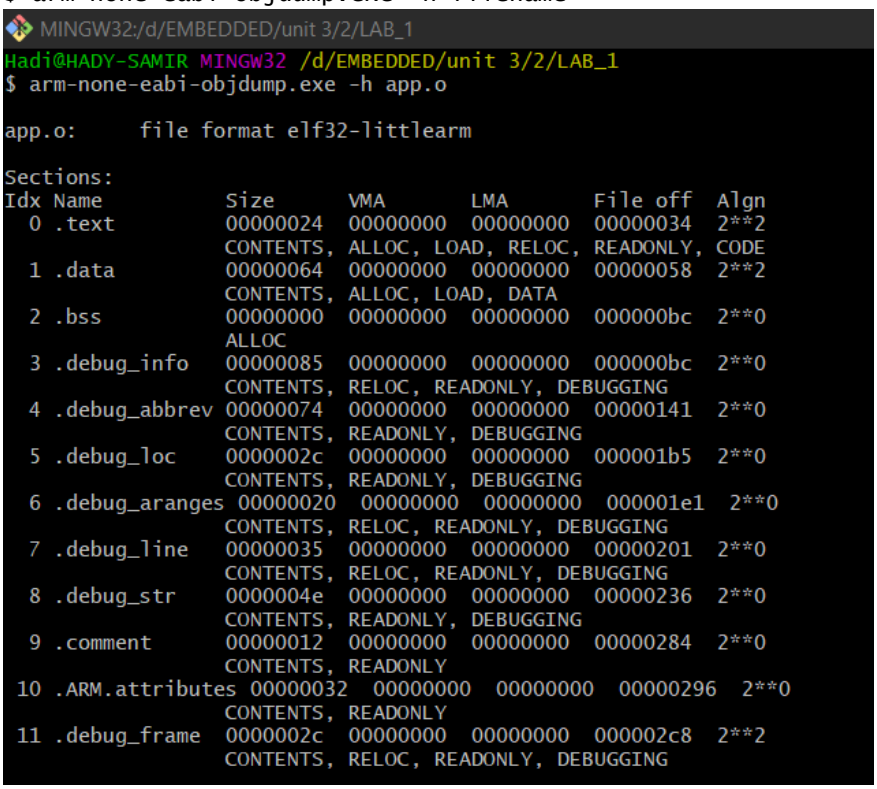
- `$ qemu-system-arm.exe -M versatilepb -m 128M -nographic -kernel learn-in-depth.bin`



Git Commands Used In Analysis Process:

To display the content of the section headers:

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



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

```
Hadi@HADY-SAMIR MINGW32 /d/EMBEDDED/unit 3/2/LAB_1
$ arm-none-eabi-ld.exe  -T  linker_script.ld  app.o uart.o startup.o -o learn_in_depth.elf

Hadi@HADY-SAMIR MINGW32 /d/EMBEDDED/unit 3/2/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 .startup      00000010  00010000  00010000  00008000  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .text         00000074  00010010  00010010  00008010  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  2 .data         00000064  00010084  00010084  00008084  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  3 .ARM.attributes 0000002e  00000000  00000000  000080e8  2**0
                  CONTENTS, READONLY
  4 .comment      00000011  00000000  00000000  00008116  2**0
                  CONTENTS, READONLY

Hadi@HADY-SAMIR MINGW32 /d/EMBEDDED/unit 3/2/LAB_1
```

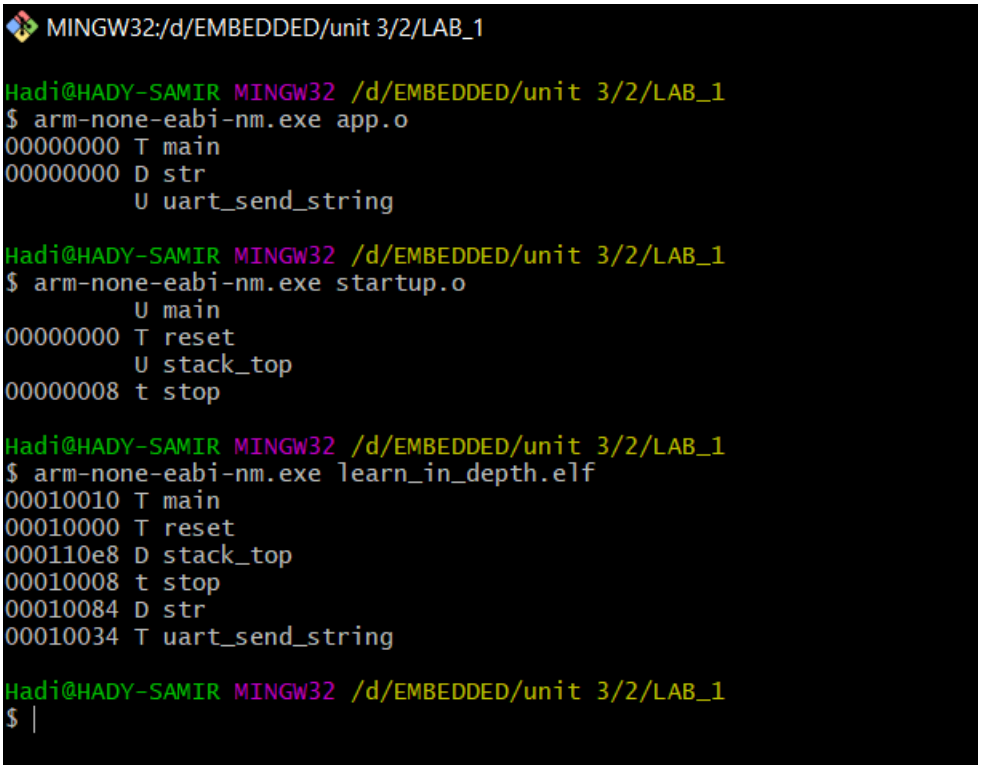
To display the assembler content of all the sections:

- $ arm-none-eabi-objdump.exe -D Filename

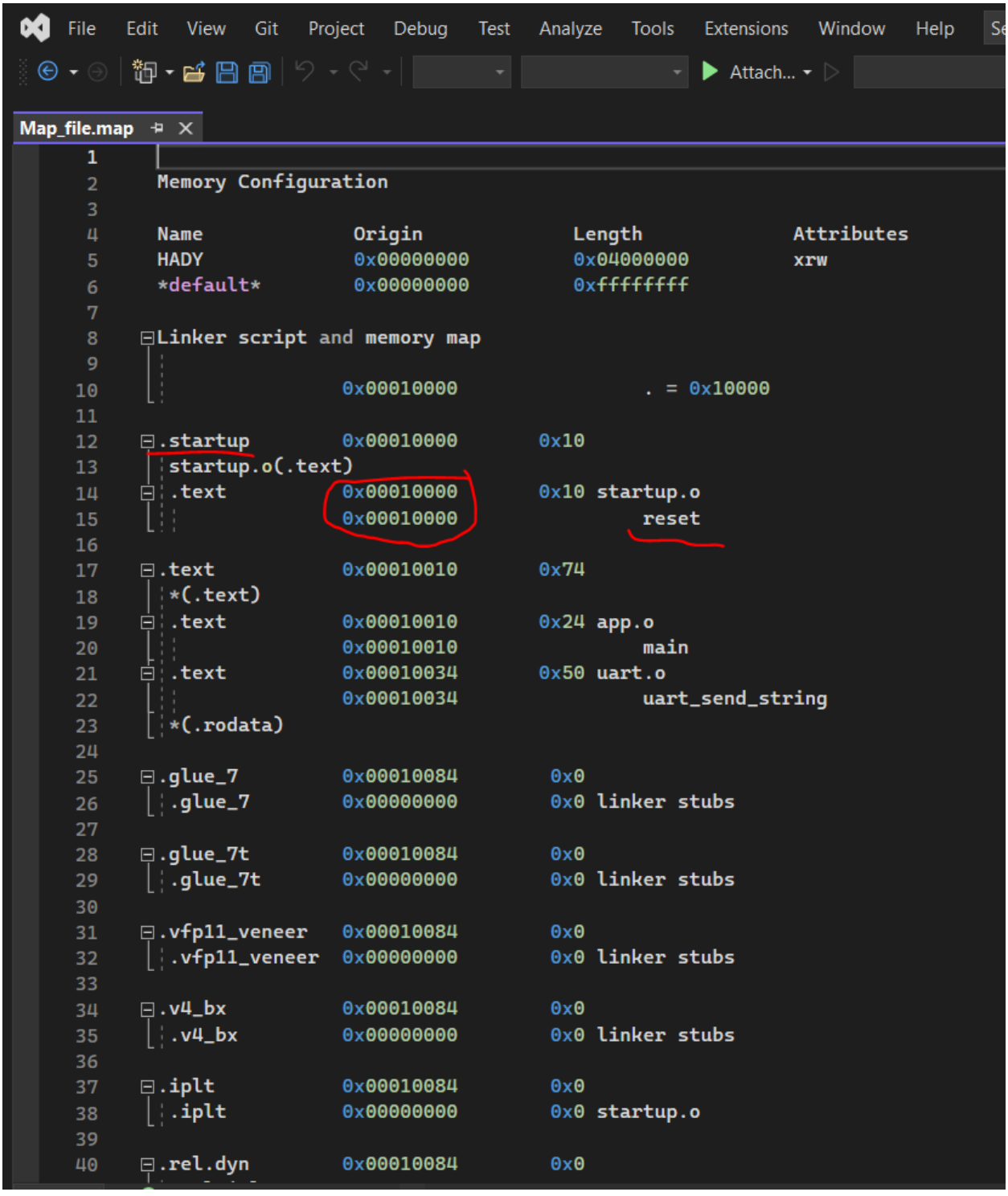
```
Hadi@HADY-SAMIR MINGW32 /d/EMBEDDED/unit 3/2/LAB_1
$ arm-none-eabi-objdump.exe  -D  learn_in_depth.elf

learn_in_depth.elf:     file format elf32-littlearm


Disassembly of section .startup:

00010000 <reset>:
   10000:       e59fd004        ldr     sp, [pc, #4]    ; 1000c <stop+0x4>
   10004:       eb000001        bl      10010 <main>

00010008 <stop>:
   10008:       eafffffe        b       10008 <stop>
   1000c:       000110e8        andeq   r1, r1, r8, ror #1

Disassembly of section .text:

00010010 <main>:
   10010:       e92d4800        push    {fp, lr}
   10014:       e28db004        add     fp, sp, #4
   10018:       e24dd008        sub     sp, sp, #8
   1001c:       e50b0008        str     r0, [fp, #-8]
   10020:       e59f0008        ldr     r0, [pc, #8]    ; 10030 <main+0x20>
   10024:       eb000002        bl      10034 <uart_send_string>
   10028:       e24bd004        sub     sp, fp, #4
   1002c:       e8bd8800        pop     {fp, pc}
   10030:       00010084        andeq   r0, r1, r4, lsl #1

00010034 <uart_send_string>:
   10034:       e52db004        push    {fp}            ; (str fp, [sp, #-4]!)
   10038:       e28db000        add     fp, sp, #0
   1003c:       e24dd00c        sub     sp, sp, #12
   10040:       e50b0008        str     r0, [fp, #-8]
   10044:       ea000006        b       10064 <uart_send_string+0x30>
   10048:       e59f3030        ldr     r3, [pc, #48]   ; 10080 <uart_send_string+0x4c>
```

To read the symbols and check the Entry Point Address:

- `$ arm-none-eabi-nm.exe Filename`



```
MINGW32:/d/EMBEDDED/unit 3/2/LAB_1

Hadi@HADY-SAMIR MINGW32 /d/EMBEDDED/unit 3/2/LAB_1
$ arm-none-eabi-objcopy.exe  -O binary  learn_in_depth.elf  learn_in_depth.bin

Hadi@HADY-SAMIR MINGW32 /d/EMBEDDED/unit 3/2/LAB_1
$ arm-none-eabi-ld.exe  -T  linker_script.ld app.o uart.o startup.o -o learn_in_depth.elf
ap

Hadi@HADY-SAMIR MINGW32 /d/EMBEDDED/unit 3/2/LAB_1
$ arm-none-eabi-readelf.exe  -a learn_in_depth.elf
ELF Header:
  Magic:   7f 45 4c 46 01 01 01 00 00 00 00 00 00 00 00 00
  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:          34288 (bytes into file)
  Flags:                             0x5000002, has entry point, Version5 EABI
  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:         16
  Section header string table index: 13

Section Headers:
  [Nr] Name              Type            Addr     Off    Size   ES Flg Lk Inf Al
  [ 0]                   NULL            00000000 000000 000000 00      0   0  0
  [ 1] .startup          PROGBITS        00010000 008000 000010 00  AX  0   0  4
  [ 2] .text             PROGBITS        00010010 008010 000074 00  AX  0   0  4
  [ 3] .data             PROGBITS        00010084 008084 000064 00  WA  0   0  4
  [ 4] .ARM.attributes   ARM_ATTRIBUTES  00000000 0080e8 00002e 00      0   0  1
  [ 5] .comment          PROGBITS        00000000 008116 000011 01  MS  0   0  1
  [ 6] .debug_line       PROGBITS        00000000 008127 0000ac 00      0   0  1
  [ 7] .debug_info       PROGBITS        00000000 0081d3 00012c 00      0   0  1
  [ 8] .debug_abbrev     PROGBITS        00000000 0082ff 0000d9 00      0   0  1
  [ 9] .debug_aranges    PROGBITS        00000000 0083d8 000060 00      0   0  8
  [10] .debug_loc        PROGBITS        00000000 008438 000058 00      0   0  1
  [11] .debug_str        PROGBITS        00000000 008490 000066 01  MS  0   0  1
  [12] .debug_frame      PROGBITS        00000000 0084f8 000054 00      0   0  4
  [13] .shstrtab         STRTAB          00000000 00854c 0000a1 00      0   0  1
  [14] .symtab           SYMTAB          00000000 008870 000210 10     15  28  4
  [15] .strtab           STRTAB          00000000 008a80 00004d 00      0   0  1
Key to Flags:
  W (write), A (alloc), X (execute), M (merge), S (strings)
  I (info), L (link order), G (group), T (TLS), E (exclude), x (unknown)
  O (extra OS processing required) o (OS specific), p (processor specific)
```


File Edit View Git Project Debug Test Analyze Tools Extensions Window Help
Attach...
Map_file.map
Memory Configuration
Name Origin Length Attributes
HADY 0x00000000 0x04000000 xrw
default 0x00000000 0xffffffff
Linker script and memory map
0x00010000 . = 0x10000
.startup 0x00010000 0x10
startup.o(.text)
.text 0x00010000 0x10 startup.o
0x00010000 reset
.text 0x00010010 0x74
*(.text)
.text 0x00010010 0x24 app.o
0x00010010 main
.text 0x00010034 0x50 uart.o
0x00010034 uart_send_string
*(.rodata)
.glue_7 0x00010084 0x0
.glue_7 0x00000000 0x0 linker stubs
.glue_7t 0x00010084 0x0
.glue_7t 0x00000000 0x0 linker stubs
.vfp11_veneer 0x00010084 0x0
.vfp11_veneer 0x00000000 0x0 linker stubs
.v4_bx 0x00010084 0x0
.v4_bx 0x00000000 0x0 linker stubs
.iplt 0x00010084 0x0
.iplt 0x00000000 0x0 startup.o
.rel.dyn 0x00010084 0x0