LAB 1#

(Using VersatilePB virtual board in QEMU and ARM toolchain)

1.Writing source files, getting object files and analyzing them.

1.1 (with debug information)

```
kinge@DESKTOP-MJQR6DC MINGW32 /d/Courses/Embedded Diploma/Course Content/Unit_3_Embedded_C/2.Lecture_2_/Lab1
$ arm-none-eabi-gcc -c -g -I . -mcpu=arm926ej-s app.c -o app.o

kinge@DESKTOP-MJQR6DC MINGW32 /d/Courses/Embedded Diploma/Course Content/Unit_3_Embedded_C/2.Lecture_2_/Lab1
$ arm-none-eabi-gcc -c -g -I . -mcpu=arm926ej-s uart.c -o uart.o
```

```
kinge@DESKTOP-MJQR6DC MINGW32 /d/Courses/Embedded Diploma/Course Content/Unit_3_Embedded_C/2.Lecture_2_/Lab1
$ 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    0000009b  00000000  00000000  00000279  2**0
                  CONTENTS, READONLY, DEBUGGING
 10 .comment      00000012  00000000  00000000  00000314  2**0
                  CONTENTS, READONLY
 11 .ARM.attributes 00000032  00000000  00000000  00000326  2**0
                  CONTENTS, READONLY
 12 .debug_frame  0000002c  00000000  00000000  00000358  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
```

```
kinge@DESKTOP-MJQR6DC MINGW32 /d/Courses/Embedded Diploma/Course Content/Unit_3_Embedded_C/2.Lecture_2_/Lab1
$ 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 .debug_info   0000005c  00000000  00000000  00000084  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  4 .debug_abbrev 00000051  00000000  00000000  000000e0  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_loc    0000002c  00000000  00000000  00000131  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_aranges 00000020  00000000  00000000  0000015d  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  7 .debug_line   0000003d  00000000  00000000  0000017d  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  8 .debug_str    0000008d  00000000  00000000  000001ba  2**0
                  CONTENTS, READONLY, DEBUGGING
  9 .comment      00000012  00000000  00000000  00000247  2**0
                  CONTENTS, READONLY
 10 .ARM.attributes 00000032  00000000  00000000  00000259  2**0
                  CONTENTS, READONLY
 11 .debug_frame  00000028  00000000  00000000  0000028c  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
```

1.2 (without debug information)

```
kinge@DESKTOP-MJQR6DC MINGW32 /d/Courses/Embedded Diploma/Course Content/Unit_3_Embedded_C/2.Lecture_2_/Lab1
$ arm-none-eabi-gcc -c -I . -mcpu=arm926ej-s app.c -o app.o

kinge@DESKTOP-MJQR6DC MINGW32 /d/Courses/Embedded Diploma/Course Content/Unit_3_Embedded_C/2.Lecture_2_/Lab1
$ arm-none-eabi-gcc -c -I . -mcpu=arm926ej-s uart.c -o uart.o
```

```
kinge@DESKTOP-MJQR6DC MINGW32 /d/Courses/Embedded Diploma/Course Content/Unit_3_Embedded_C/2.Lecture_2_/Lab1
$ 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
```

```
kinge@DESKTOP-MJQR6DC MINGW32 /d/Courses/Embedded Diploma/Course Content/Unit_3_Embedded_C/2.Lecture_2_/Lab1
$ 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
```

2. Writing startup code, getting object file and analyzing it.

```
kinge@DESKTOP-MJQR6DC MINGW32 /d/Courses/Embedded Diploma/Course Content/Unit_3_Embedded_C/2.Lecture_2_/Lab1
$ arm-none-eabi-as.exe -mcpu=arm926ej-s startup.s -o staetup.o
startup.s: Assembler messages:
startup.s: Warning: end of file not at end of a line; newline inserted

kinge@DESKTOP-MJQR6DC MINGW32 /d/Courses/Embedded Diploma/Course Content/Unit_3_Embedded_C/2.Lecture_2_/Lab1
$ 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
```

3. Writing the linker script, linking all objects, getting the elf file and analyzing it.

```
inge@DESKTOP-MJQR6DC MINGW32 /d/Courses/Embedded Diploma/Course Content/Unit_3_Embedded_C/2.Lecture_2_/Lab1
$ arm-none-eabi-ld.exe -T linker_script.ld startup.o uart.o app.o -o learn-in-depth.elf

inge@DESKTOP-MJQR6DC MINGW32 /d/Courses/Embedded Diploma/Course Content/Unit_3_Embedded_C/2.Lecture_2_/Lab1
$ arm-none-eabi-objdump.exe -h learn-in-depth.elf

earn-in-depth.elf:     file format elf32-littlearm

ections:
dx Name          Size      VMA       LMA       File off  Algn
 0 .startup      0000000c  00010000  00010000  00008000  2**2
                 CONTENTS, ALLOC, LOAD, READONLY, CODE
 1 .text         00000068  0001000c  0001000c  0000800c  2**2
                 CONTENTS, ALLOC, LOAD, READONLY, CODE
 2 .rodata       00000064  00010074  00010074  00008074  2**2
                 CONTENTS, ALLOC, LOAD, READONLY, DATA
 3 .data         00000064  000100d8  000100d8  000080d8  2**2
                 CONTENTS, ALLOC, LOAD, DATA
 4 .ARM.attributes 0000002e  00000000  00000000  0000813c  2**0
                 CONTENTS, READONLY
 5 .comment      00000011  00000000  00000000  0000816a  2**0
                 CONTENTS, READONLY
```

3.1 We also can get .map file.

```
kinge@DESKTOP-MJQR6DC MINGW32 /d/Courses/Embedded Diploma/Course Content/Unit_3_Embedded_C/2.Lecture_2_/Lab1
$ arm-none-eabi-ld.exe -T linker_script.ld -Map=mapOut.map startup.o uart.o app.o -o learn-in-depth.elf
```

3.2 We also can use readelf.exe To make sure about the entry point at address.

```
kinge@DESKTOP-MJQR6DC MINGW32 /d/Courses/Embedded Diploma/Course Content/Unit_3_Embedded_C/2.Lecture_2_/Lab1
$ 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:          33228 (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:         10
  Section header string table index: 7

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 00000c 00  AX  0   0  4
  [ 2] .text             PROGBITS        0001000c 00800c 000068 00  AX  0   0  4
  [ 3] .rodata           PROGBITS        00010074 008074 000064 00   A  0   0  4
  [ 4] .data             PROGBITS        000100d8 0080d8 000064 00  WA  0   0  4
  [ 5] .ARM.attributes   ARM_ATTRIBUTES  00000000 00813c 00002e 00      0   0  1
  [ 6] .comment          PROGBITS        00000000 00816a 000011 01  MS  0   0  1
  [ 7] .shstrtab         STRTAB          00000000 00817b 000051 00      0   0  1
  [ 8] .symtab           SYMTAB          00000000 00835c 000190 10      9  19  4
  [ 9] .strtab           STRTAB          00000000 0084ec 000067 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)
```

4. Getting the symbol table for the object files and the final elf file.

```
kinge@DESKTOP-MJQR6DC MINGW32 /d/Courses/Embedded Diploma/Course Content/Unit_3_Embedded_C/2.Lecture_2_/Lab1
$ arm-none-eabi-nm.exe app.o
00000000 T main
00000000 D string_buffer1
00000000 R string_buffer2
         U Uart_Send_String

kinge@DESKTOP-MJQR6DC MINGW32 /d/Courses/Embedded Diploma/Course Content/Unit_3_Embedded_C/2.Lecture_2_/Lab1
$ arm-none-eabi-nm.exe uart.o
00000000 T Uart_Send_String

kinge@DESKTOP-MJQR6DC MINGW32 /d/Courses/Embedded Diploma/Course Content/Unit_3_Embedded_C/2.Lecture_2_/Lab1
$ arm-none-eabi-nm.exe startup.o
         U main
00000000 T reset
00000008 t stop

kinge@DESKTOP-MJQR6DC MINGW32 /d/Courses/Embedded Diploma/Course Content/Unit_3_Embedded_C/2.Lecture_2_/Lab1
$ arm-none-eabi-nm.exe learn-in-depth.elf
0001005c T main
00010000 T reset
0001113c D stack_top
00010008 t stop
000100d8 D string_buffer1
00010074 R string_buffer2
0001000c T Uart_Send_String
```

5. Getting the binary file and simulating the application using QEMU.

```
kinge@DESKTOP-MJQR6DC MINGW32 /d/Courses/Embedded Diploma/Course Content/Unit_3_Em
$ arm-none-eabi-objcopy.exe -O binary learn-in-depth.elf learn-in-depth.bin
```

5.1. go to qemu path.

```
kinge@DESKTOP-MJQR6DC MINGW32 /c/Program Files (x86)/qemu
$ qemu-system-arm -M versatilepb -m 128M -nographic -kernel learn-in-depth.bin
learn-in-depth:Abdelrahman
```

You can see all (app.c , uart.c , uarrt.h , startup.s, linker_script.ld) on my github repo

https://github.com/aaref5720/Master_Embedded_Systems/tree/main/Unit_3_Embedded_C/Lesson_2_Assignments