# Embedded c lesson 2 lab1

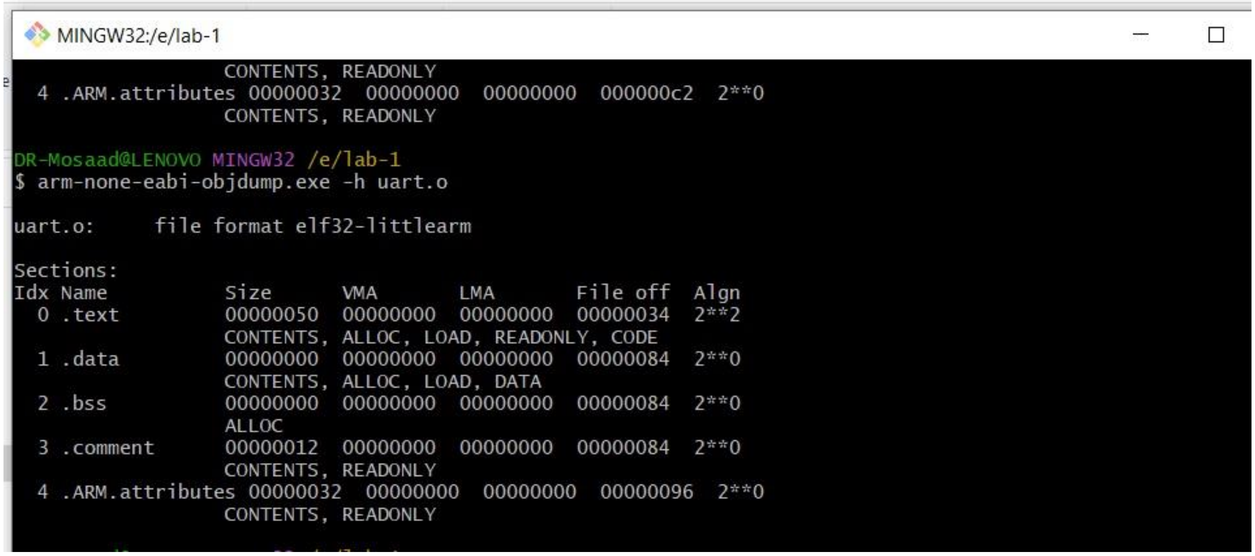
AYMAN_GAMAL

## STEPS

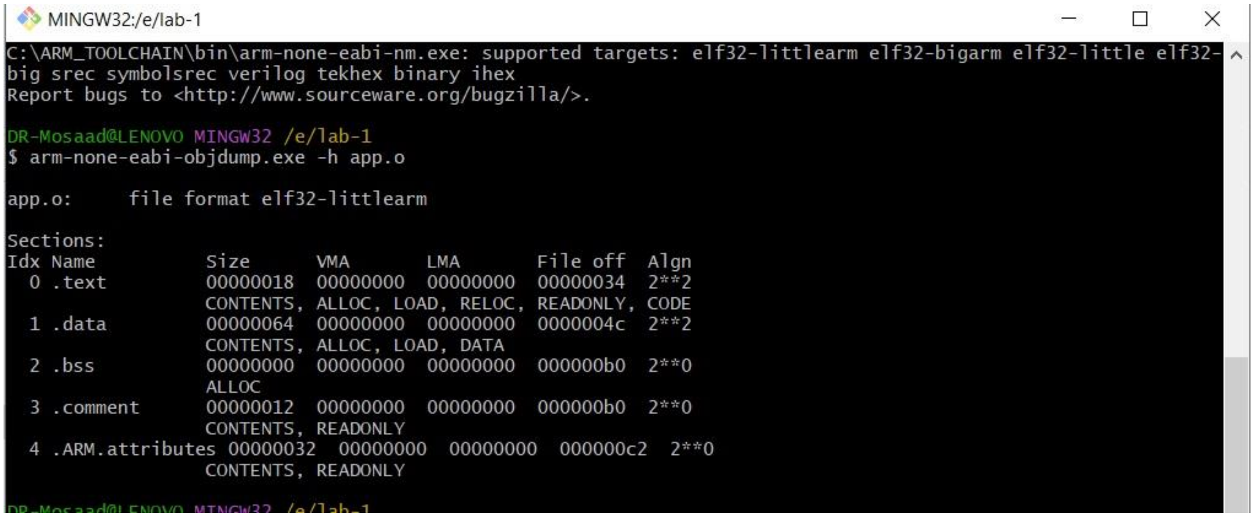
- Create app.c , uart.c and uart.h (using touch)
- Creat linkerscript.ld and startup.s (using touch)
- creat makefile ( to comiple all file in this location with cross toolchain)
- and get learn_in_depth.bin ,app.o,uart.o,startup.o l_i_d.elf (from makefile)
- bass learn_in_depth.bin to our machine (cpu(arm926ej-s))

  Hint(I didn’t use debug from cross toolchain (–g))

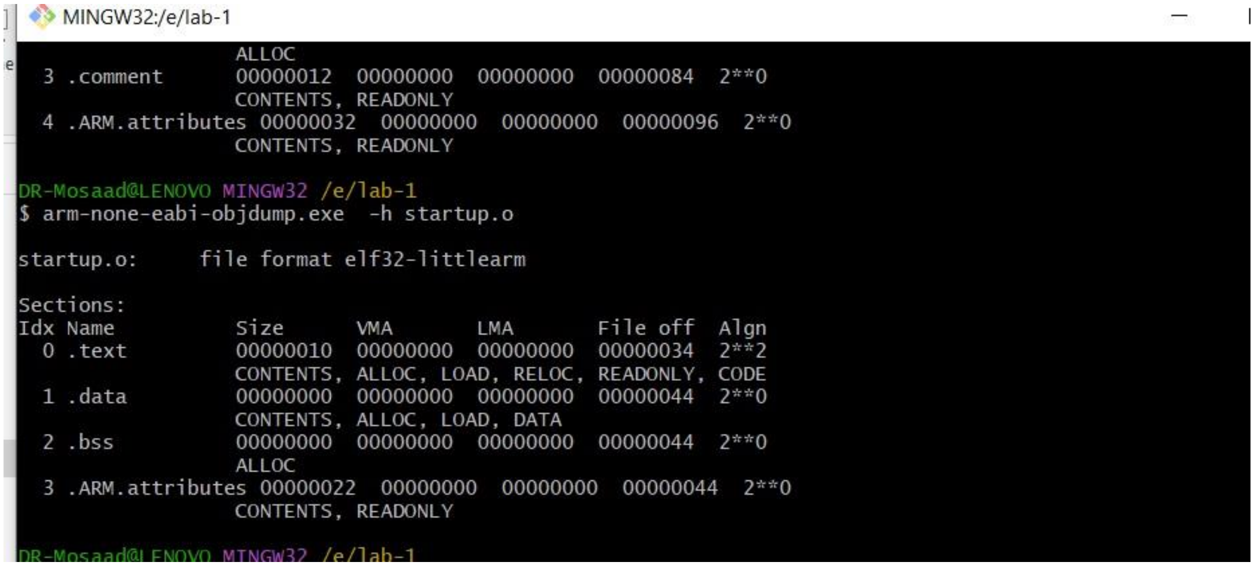
## - **sections**



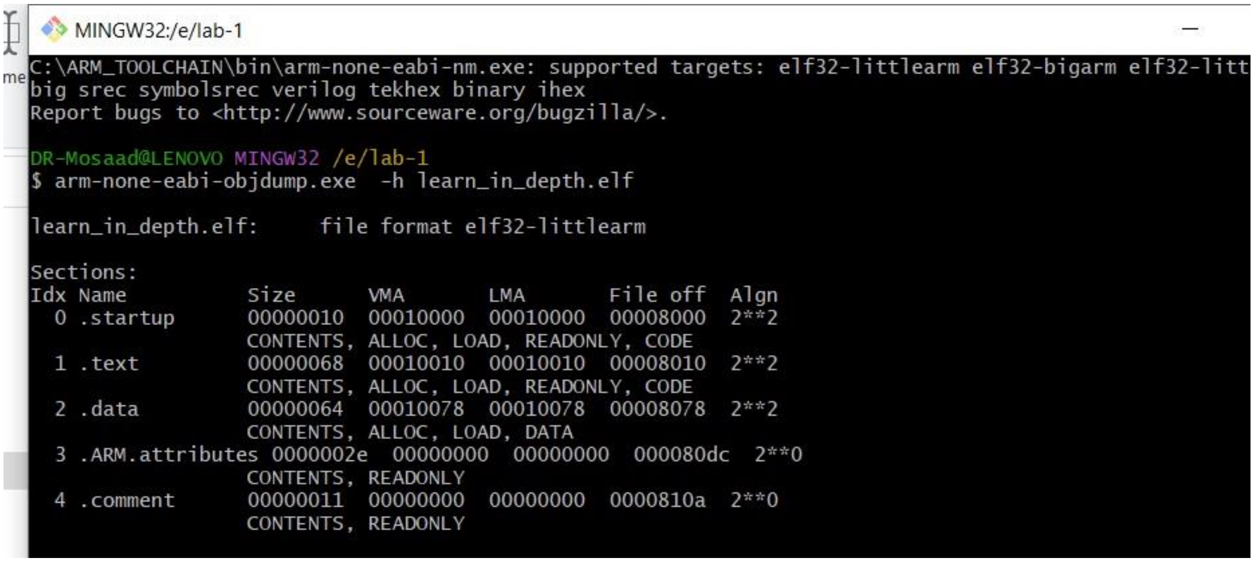
```
                   CONTENTS, READONLY
  4 .ARM.attributes 00000032  00000000  00000000  000000c2  2**0
                   CONTENTS, READONLY

DR-Mosaad@LENOVO MINGW32 /e/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
```



```
C:\ARM_TOOLCHAIN\bin\arm-none-eabi-nm.exe: supported targets: elf32-littlearm elf32-bigarm elf32-little elf32-
big srec symbolsrec verilog tekhex binary ihex
Report bugs to <http://www.sourceware.org/bugzilla/>.

DR-Mosaad@LENOVO MINGW32 /e/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 .comment      00000012  00000000  00000000  000000b0  2**0
                  CONTENTS, READONLY
  4 .ARM.attributes 00000032  00000000  00000000  000000c2  2**0
                  CONTENTS, READONLY

DR-Mosaad@LENOVO MINGW32 /e/lab-1
```

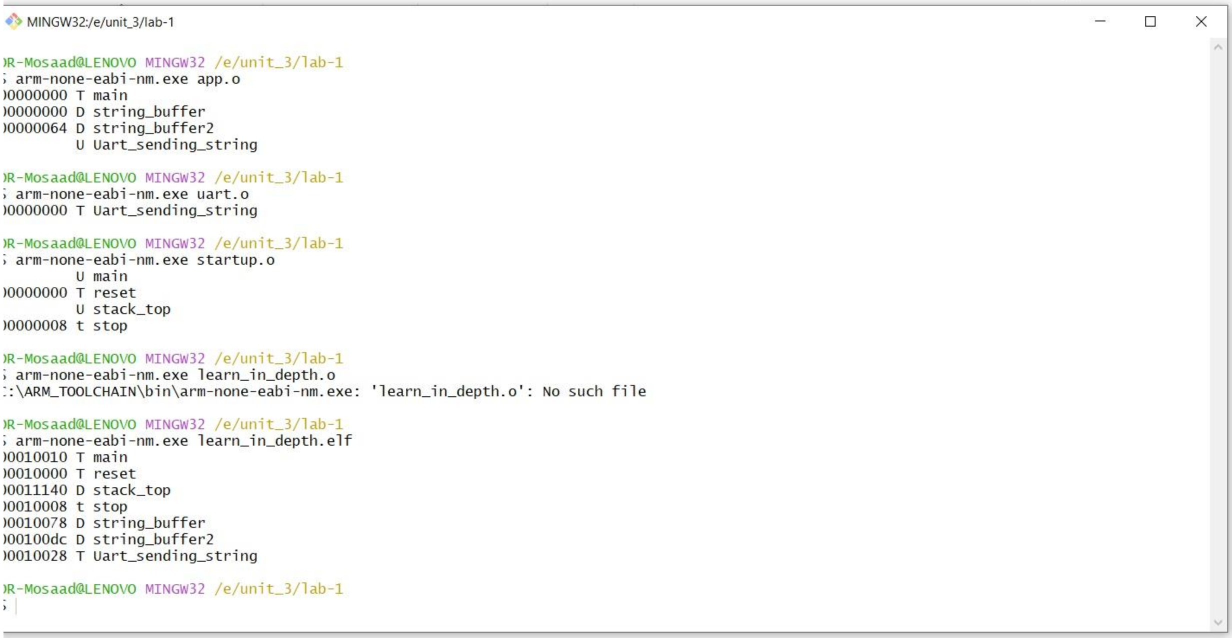


```
                  ALLOC
  3 .comment      00000012  00000000  00000000  00000084  2**0
                  CONTENTS, READONLY
  4 .ARM.attributes 00000032  00000000  00000000  00000096  2**0
                  CONTENTS, READONLY

DR-Mosaad@LENOVO MINGW32 /e/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

DR-Mosaad@LENOVO MINGW32 /e/lab-1
```



```
C:\ARM_TOOLCHAIN\bin\arm-none-eabi-nm.exe: supported targets: elf32-littlearm elf32-bigarm elf32-litt
big srec symbolsrec verilog tekhex binary ihex
Report bugs to <http://www.sourceware.org/bugzilla/>.

DR-Mosaad@LENOVO MINGW32 /e/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         00000068  00010010  00010010  00008010  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  2 .data         00000064  00010078  00010078  00008078  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  3 .ARM.attributes 0000002e  00000000  00000000  000080dc  2**0
                  CONTENTS, READONLY
  4 .comment      00000011  00000000  00000000  0000810a  2**0
                  CONTENTS, READONLY
```

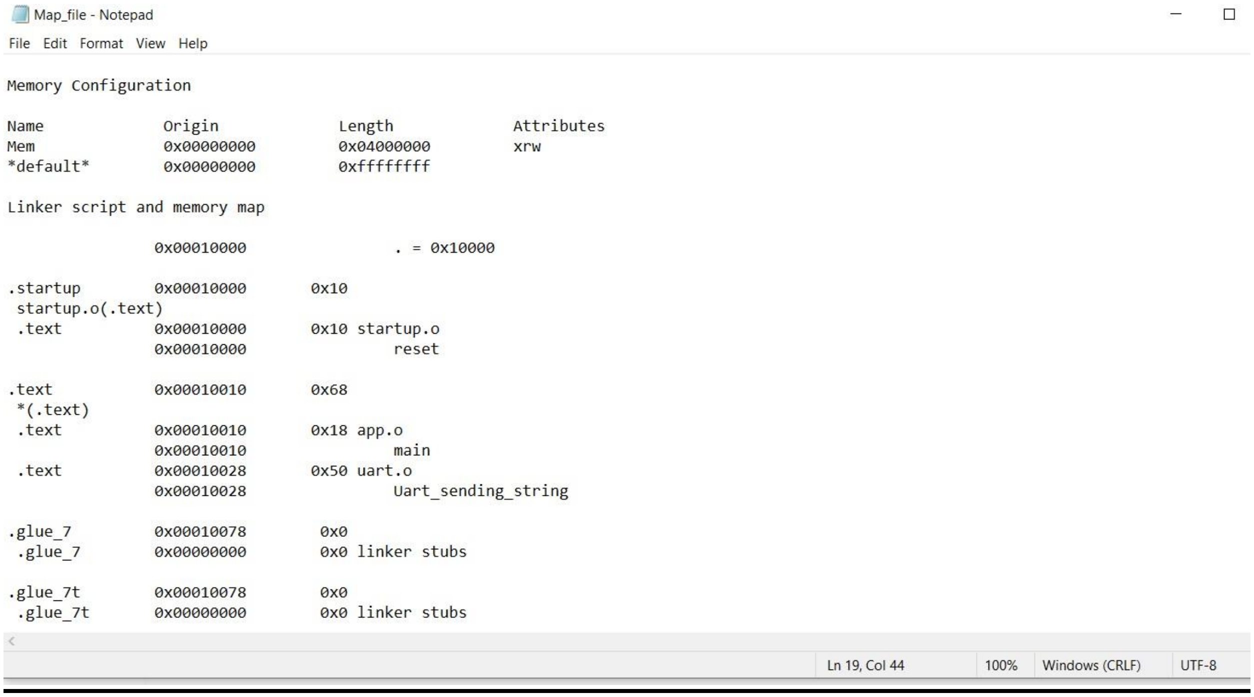
## - symbols



MINGW32:/e/unit_3/lab-1

```
)R-Mosaad@LENOVO MINGW32 /e/unit_3/lab-1
; arm-none-eabi-nm.exe app.o
)0000000 T main
)0000000 D string_buffer
)0000064 D string_buffer2
         U Uart_sending_string

)R-Mosaad@LENOVO MINGW32 /e/unit_3/lab-1
; arm-none-eabi-nm.exe uart.o
)0000000 T Uart_sending_string

)R-Mosaad@LENOVO MINGW32 /e/unit_3/lab-1
; arm-none-eabi-nm.exe startup.o
         U main
)0000000 T reset
         U stack_top
)0000008 t stop

)R-Mosaad@LENOVO MINGW32 /e/unit_3/lab-1
; arm-none-eabi-nm.exe learn_in_depth.o
:\ARM_TOOLCHAIN\bin\arm-none-eabi-nm.exe: 'learn_in_depth.o': No such file

)R-Mosaad@LENOVO MINGW32 /e/unit_3/lab-1
; arm-none-eabi-nm.exe learn_in_depth.elf
)0010010 T main
)0010000 T reset
)0011140 D stack_top
)0010008 t stop
)0010078 D string_buffer
)00100dc D string_buffer2
)0010028 T Uart_sending_string

)R-Mosaad@LENOVO MINGW32 /e/unit_3/lab-1
;
```

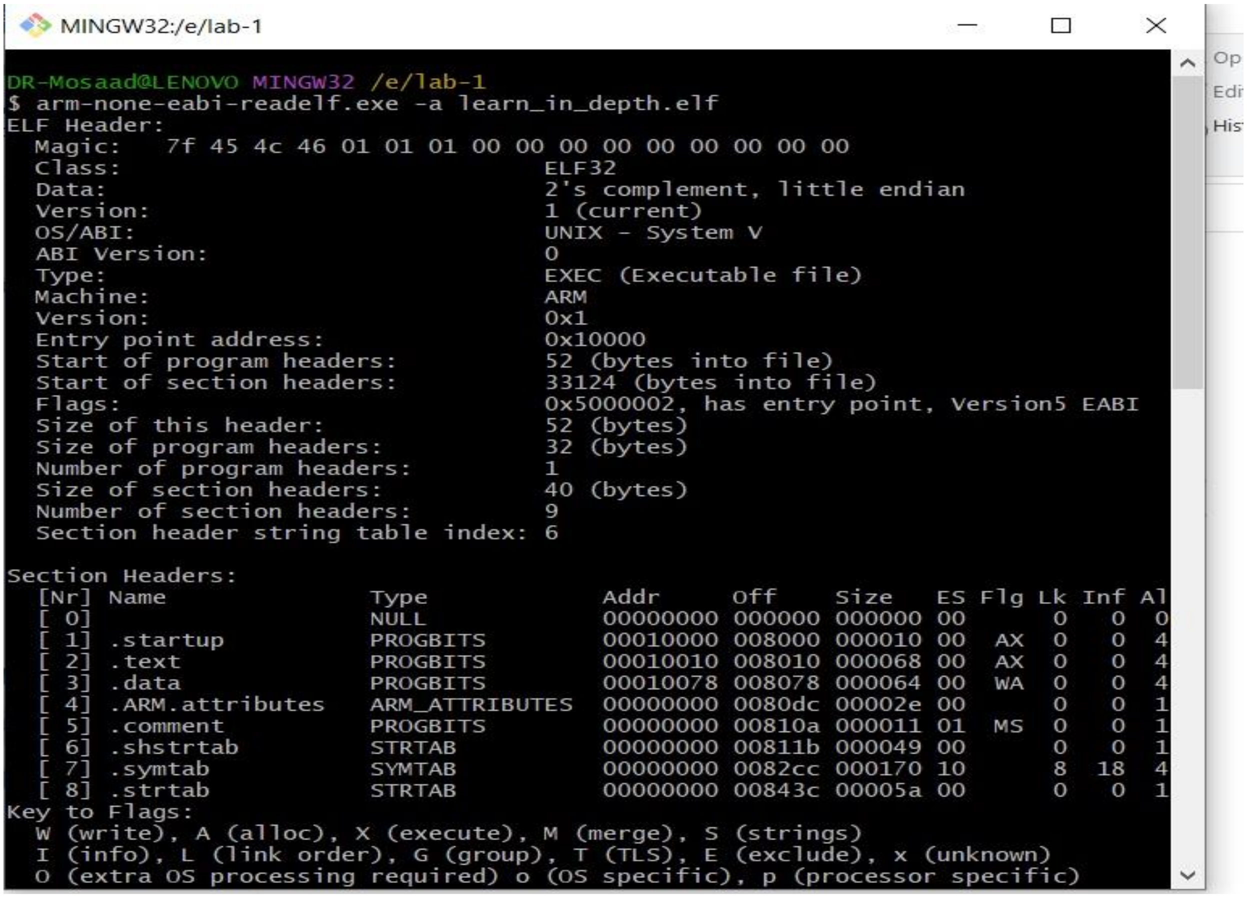
## - Mapfile



Map_file - Notepad

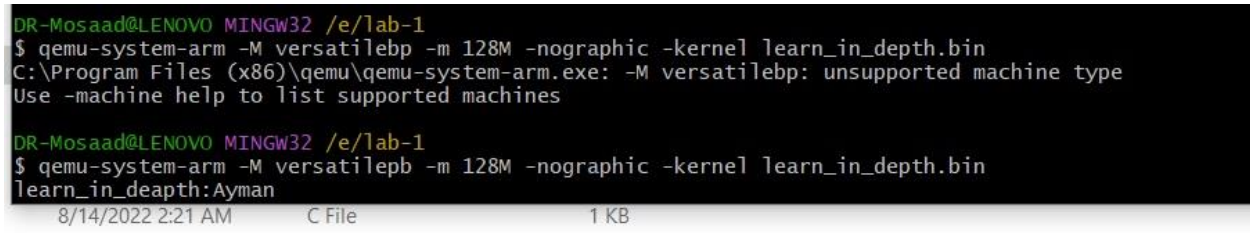
```
Memory Configuration

Name             Origin             Length             Attributes
Mem              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       0x68
 *(.text)
 .text          0x00010010       0x18 app.o
                0x00010010                main
 .text          0x00010028       0x50 uart.o
                0x00010028                Uart_sending_string

.glue_7         0x00010078        0x0
 .glue_7        0x00000000        0x0 linker stubs

.glue_7t        0x00010078        0x0
 .glue_7t       0x00000000        0x0 linker stubs
```

- **Readelf**



```
DR-Mosaad@LENOVO MINGW32 /e/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:          33124 (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:         9
  Section header string table index: 6

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 000068 00  AX  0   0  4
  [ 3] .data             PROGBITS        00010078 008078 000064 00  WA  0   0  4
  [ 4] .ARM.attributes   ARM_ATTRIBUTES  00000000 0080dc 00002e 00      0   0  1
  [ 5] .comment          PROGBITS        00000000 00810a 000011 01  MS  0   0  1
  [ 6] .shstrtab         STRTAB          00000000 00811b 000049 00      0   0  1
  [ 7] .symtab           SYMTAB          00000000 0082cc 000170 10      8  18  4
  [ 8] .strtab           STRTAB          00000000 00843c 00005a 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)
```

- **qemu (burn code on the bord)**



```
DR-Mosaad@LENOVO MINGW32 /e/lab-1
$ qemu-system-arm -M versatilebp -m 128M -nographic -kernel learn_in_depth.bin
C:\Program Files (x86)\qemu\qemu-system-arm.exe: -M versatilebp: unsupported machine type
Use -machine help to list supported machines

DR-Mosaad@LENOVO MINGW32 /e/lab-1
$ qemu-system-arm -M versatilepb -m 128M -nographic -kernel learn_in_depth.bin
learn_in_deapth:Ayman
```