

LAB ONE (Second embedded C Lecture)

Using obj-dump utility (-h) to show sections headers with debugging information.

```
MINGW64//EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
salah@DESKTOP-03RSCJN MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10
/lab1
$ arm-none-eabi-gcc.exe -c -I . -g -mcpu=arm926ej-s app.c -o app.o
salah@DESKTOP-03RSCJN MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10
/lab1
$ arm-none-eabi-gcc.exe -c -I . -g -mcpu=arm926ej-s uart.c -o uart.o
salah@DESKTOP-03RSCJN MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10
/lab1
$ arm-none-eabi-objdump.exe -h app.o
bash: arm-none-eabi-objdump.exe: command not found
salah@DESKTOP-03RSCJN MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10
/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          00000020  00000000  00000000  00000034  2**2
    CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data          00000064  00000000  00000000  00000054  2**2
    CONTENTS, ALLOC, LOAD, DATA
  2 .bss           00000000  00000000  00000000  000000b8  2**0
    ALLOC
  3 .debug_info     00000076  00000000  00000000  000000b8  2**0
    CONTENTS, RELOC, READONLY, DEBUGGING
  4 .debug_abbrev   00000065  00000000  00000000  0000012e  2**0
    CONTENTS, READONLY, DEBUGGING
  5 .debug_loc      0000002c  00000000  00000000  00000193  2**0
    CONTENTS, READONLY, DEBUGGING
  6 .debug_aranges  00000020  00000000  00000000  000001bf  2**0
    CONTENTS, RELOC, READONLY, DEBUGGING
  7 .debug_line     00000036  00000000  00000000  000001df  2**0
    CONTENTS, RELOC, READONLY, DEBUGGING
  8 .debug_str      0000006d  00000000  00000000  00000215  2**0
    CONTENTS, READONLY, DEBUGGING
  9 .comment        00000012  00000000  00000000  00000282  2**0
    CONTENTS, READONLY
10 .ARM.attributes 00000032  00000000  00000000  00000294  2**0
    CONTENTS, READONLY
11 .debug_frame    0000002c  00000000  00000000  000002c8  2**2
    CONTENTS, RELOC, READONLY, DEBUGGING

salah@DESKTOP-03RSCJN MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10
/lab1
$
```

Using obj-dump utility (-h) to show sections headers without debugging information.

```
MINGW64//EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
1 .data          00000064  00000000  00000000  00000054  2**2
  CONTENTS, ALLOC, LOAD, DATA
2 .bss           00000000  00000000  00000000  000000b8  2**0
  ALLOC
3 .debug_info     00000076  00000000  00000000  000000b8  2**0
  CONTENTS, RELOC, READONLY, DEBUGGING
4 .debug_abbrev   00000065  00000000  00000000  0000012e  2**0
  CONTENTS, READONLY, DEBUGGING
5 .debug_loc      0000002c  00000000  00000000  00000193  2**0
  CONTENTS, READONLY, DEBUGGING
6 .debug_aranges  00000020  00000000  00000000  000001bf  2**0
  CONTENTS, RELOC, READONLY, DEBUGGING
7 .debug_line     00000036  00000000  00000000  000001df  2**0
  CONTENTS, RELOC, READONLY, DEBUGGING
8 .debug_str      0000006d  00000000  00000000  00000215  2**0
  CONTENTS, READONLY, DEBUGGING
9 .comment        00000012  00000000  00000000  00000282  2**0
  CONTENTS, READONLY
10 .ARM.attributes 00000032  00000000  00000000  00000294  2**0
  CONTENTS, READONLY
11 .debug_frame    0000002c  00000000  00000000  000002c8  2**2
  CONTENTS, RELOC, READONLY, DEBUGGING

salah@DESKTOP-03RSCJN MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10
/lab1
$ arm-none-eabi-gcc.exe -c -I . -mcpu=arm926ej-s app.c -o app.o
salah@DESKTOP-03RSCJN MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10
/lab1
$ arm-none-eabi-gcc.exe -c -I . -mcpu=arm926ej-s uart.c -o uart.o
salah@DESKTOP-03RSCJN MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10
/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          00000020  00000000  00000000  00000034  2**2
    CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data          00000064  00000000  00000000  00000054  2**2
    CONTENTS, ALLOC, LOAD, DATA
  2 .bss           00000000  00000000  00000000  000000b8  2**0
    ALLOC
  3 .comment       00000012  00000000  00000000  000000b8  2**0
    CONTENTS, READONLY
  4 .ARM.attributes 00000032  00000000  00000000  000000ca  2**0
    CONTENTS, READONLY

salah@DESKTOP-03RSCJN MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10
/lab1
$
```

Using obj-dump utility (-h) to show sections headers after adding constant global data.

```
MINGW64/F:/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
salah@DESKTOP-Q3R5CJN MINGW64 /F:/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ arm-none-eabi-gcc.exe -c -I . -mcpu=arm926ej-s app.c -o app.o
salah@DESKTOP-Q3R5CJN MINGW64 /F:/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/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          00000020  00000000  00000000  00000034  2**2
    CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data           00000004  00000000  00000000  00000054  2**2
    CONTENTS, ALLOC, LOAD, DATA
  2 .bss            00000000  00000000  00000000  000000b8  2**0
    ALLOC
  3 .rodata         00000064  00000000  00000000  000000b8  2**2
    CONTENTS, ALLOC, LOAD, READONLY, DATA
  4 .comment        00000012  00000000  00000000  0000011c  2**0
    CONTENTS, READONLY
  5 .ARM.attributes 00000032  00000000  00000000  0000012e  2**0
    CONTENTS, READONLY

salah@DESKTOP-Q3R5CJN MINGW64 /F:/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ |
```

Using obj-dump utility (-D) to show disassembly of app.o file.

```
app.s - 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: e59f000c      ldr     r0, [pc, #12] ; 1c <main+0x1c>
c: ebfffffe      bl      0 <uart_send_string>
10: e3a03000      mov     r3, #0
14: e1a00003      mov     r0, r3
18: e8bd8800      pop     {fp, pc}
1c: 00000000      andeq   r0, r0, r0

Disassembly of section .data:

00000000 <string>:
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: 533a6874      teqpl   sl, #116, 16 ; 0x740000
10: 48414c41      stmdami r1, {r0, r6, sl, fp, lr}^
...

Disassembly of section .rodata:

00000000 <string2>:
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: 533a6874      teqpl   sl, #116, 16 ; 0x740000
10: 48414c41      stmdami r1, {r0, r6, sl, fp, lr}^
...

Disassembly of section .comment:

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

Using obj-dump utility (-D) to show disassembly of uart.o file.

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

Disassembly of section .text:

00000000 <uart_send_string>:
 0: e52db004  push    {fp}          ; (str fp, [sp, #-4]!)
 4: e28db000  add     fp, sp, #0
 8: e24dd00c  sub     sp, sp, #12
 c: e50b0008  str     r0, [fp, #-8]
10: ea000006  b       30 <uart_send_string+0x30>
14: e59f3030  ldr     r3, [pc, #48]  ; 4c <uart_send_string+0x4c>
18: e51b2008  ldr     r2, [fp, #-8]
1c: e5d22000  ldrb    r2, [r2]
20: e5832000  str     r2, [r3]
24: e51b3008  ldr     r3, [fp, #-8]
28: e2833001  add     r3, r3, #1
2c: e50b3008  str     r3, [fp, #-8]
30: e51b3008  ldr     r3, [fp, #-8]
34: e5d33000  ldrb    r3, [r3]
38: e3530000  cmp     r3, #0
3c: 1affffff  bne     14 <uart_send_string+0x14>
40: e28bd000  add     sp, fp, #0
44: e8bd0800  ldmfd   sp!, {fp}
48: e12ffff1e  bx      lr
4c: 101f1000  andsne  r1, pc, r0

Disassembly of section .comment:

00000000 <.comment>:
 0: 43434700  movtmi  r4, #14080    ; 0x3700
 4: 4728203a  ; <UNDEFINED> instruction: 0x4728203a
 8: 2029554e  eorcs   r5, r9, lr, asr #10
 c: 2e372e34  mrccs   14, 1, r2, cr7, cr4, {1}
10: Address 0x00000010 is out of bounds.
```

Using obj-dump utility (-s) to show all sections information of app.o file.

```
MINGW64://EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
salah@DESKTOP-0JRSJC7N MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10
/!lab1
$ arm-none-eabi-obj
arm-none-eabi-objcopy.exe arm-none-eabi-objdump.exe

salah@DESKTOP-0JRSJC7N MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10
/!lab1
$ arm-none-eabi-objdump.exe -s app.o

app.o:  file format elf32-littlearm

Contents of section .text:
0000 00482de9 04b08de2 0c009fe5 feffffeb .H-----
0010 0030a0e3 0300a0e1 0088bde8 00000000 .O-----
Contents of section .data:
0000 6c656172 6e2d696e 2d646570 74683a53 learn-in-depth:S
0010 414c4148 00000000 00000000 00000000 ALAH-----
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 .rodata:
0000 6c656172 6e2d696e 2d646570 74683a53 learn-in-depth:S
0010 414c4148 00000000 00000000 00000000 ALAH-----
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 .comment:
0000 00474343 3a202847 4e532920 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 00011204 14011501 17031801 19011a01 -----
0030 1e06 ..
```

Similar information for the startup file.

```
MINGW64//EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
startup.s: Warning: end of file not at end of a line; newline inserted

salah@DESKTOP-03RSCJN MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ arm-none-eabi-as.exe -mcpu=arm926ej-s startup.s -o startup.o

salah@DESKTOP-03RSCJN MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/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          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

salah@DESKTOP-03RSCJN MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ arm-none-eabi-objdump.exe -D startup.o

startup.o:      file format elf32-littlearm

Disassembly of section .text:

00000000 <reset>:
0: e59fd004      ldr     sp, [pc, #4]    ; c <stop+0x4>
4: ebfffffe      bl     0 <main>

00000008 <stop>:
8: eaffffff     b       8 <stop>
c: 00000000     andeq  r0, r0, r0

Disassembly of section .ARM.attributes:

00000000 <.ARM.attributes>:
0: 00002141     andeq  r2, r0, r1, asr #2
4: 61656100     cmnvs  r5, r0, lsl #2
8: 01006962     tsteq  r0, r2, ror #18
c: 00000017     andeq  r0, r0, r7, lsl r0
10: 4d524105     ldrmi  f4, [r2, #-20] ; 0xffffffff
14: 45363239     ldrmi  r3, [r6, #-569]! ; 0x239
18: 0053264a     subseq r2, r3, sl, asr #26
1c: 01080506     tsteq  r8, r6, lsl #10
20: Address 0x00000020 is out of bounds.

salah@DESKTOP-03RSCJN MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ |
```

Using obj-dump utility (-h) to show sections headers information of the final executable after linking.

```
MINGW64//EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1

salah@DESKTOP-03RSCJN MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ arm-none-eabi-ld.exe -T linker.ld startup.o app.o uart.o -o learn-in-depth.elf

salah@DESKTOP-03RSCJN MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ arm-none-eabi-ld.exe -T linker.ld startup.o app.o uart.o -o learn-in-depth.elf

salah@DESKTOP-03RSCJN MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ 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 .reset          00000010  00010000  00010000  00008000  2**2
   CONTENTS, ALLOC, LOAD, READONLY, CODE
 1 .text          00000070  00010010  00010010  00008010  2**2
   CONTENTS, ALLOC, LOAD, READONLY, CODE
 2 .rodata         00000064  00010080  00010080  00008080  2**2
   CONTENTS, ALLOC, LOAD, READONLY, DATA
 3 .data           00000064  000100e4  000100e4  000080e4  2**2
   CONTENTS, ALLOC, LOAD, DATA
 4 .ARM.attributes 0000002e  00000000  00000000  00008148  2**0
   CONTENTS, READONLY
 5 .comment        00000011  00000000  00000000  00008176  2**0
   CONTENTS, READONLY

salah@DESKTOP-03RSCJN MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ |
```

Using obj-dump utility (-D) to show disassembly of the final executable after linking

```
MINGW64/F:/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
salah@DESKTOP-Q3RSCJN MINGW64 /F:/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ arm-none-eabi-objdump.exe -D learn-in-depth.elf
learn-in-depth.elf:      file format elf32-littlearm

Disassembly of section .reset:

00010000 <reset>:
1000:      e59fd004      ldr     sp, [pc, #4]      ; 1000c <stop+0x4>
1004:      eb000001      bl      10010 <main>

00010008 <stop>:
10008:      eaffffff      b       10008 <stop>
1000c:      00011148      andeq   r1, r1, r8, asr #2

Disassembly of section .text:

00010010 <main>:
10010:      e92d4800      push    {fp, lr}
10014:      e28db004      add     fp, sp, #4
10018:      e59fd00c      ldr     r0, [pc, #12]    ; 1002c <main+0x1c>
1001c:      eb000003      bl      10030 <uart_send_string>
10020:      e3a03000      mov     r3, #0
10024:      e1a00003      mov     r0, r3
10028:      e8bd8800      pop     {fp, pc}
1002c:      000100e4      andeq   r0, r1, r4, ror #1

00010030 <uart_send_string>:
10030:      e52db004      push    {fp}             ; (str fp, [sp, #-4]!)
10034:      e28db000      add     fp, sp, #0
10038:      e24d000c      sub     sp, sp, #12
1003c:      e50b0008      str     r0, [fp, #-8]
10040:      ea000006      b       10060 <uart_send_string+0x30>
10044:      e59f3030      ldr     r3, [pc, #48]    ; 1007c <uart_send_string+0x4c>
10048:      e51b2008      ldr     r2, [fp, #-8]
1004c:      e5d22000      ldrb    r2, [r2]
10050:      e5832000      str     r2, [r3]
10054:      e51b3008      ldr     r3, [fp, #-8]
10058:      e2833001      add     r3, r3, #1
1005c:      e50b3008      str     r3, [fp, #-8]
10060:      e51b3008      ldr     r3, [fp, #-8]
10064:      e5d33000      ldrb    r3, [r3]
10068:      e3530000      cmp     r3, #0
1006c:      1affffff      bne     10044 <uart_send_string+0x14>
10070:      e28bd000      add     sp, fp, #0
10074:      e8bd0800      ldmfd   sp!, {fp}
10078:      e12ffffe      bx      lr
1007c:      101f1000      andsne  r1, pc, r0

Disassembly of section .rodata:
```

Using nm cross toolchain utility to show symbols of the objects and executable files.

```
MINGW64/F:/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
salah@DESKTOP-Q3RSCJN MINGW64 /F:/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ arm-none-eabi-nm.exe startup.o
00000000 U main
00000000 T reset
00000000 U stack_address
00000008 t stop

salah@DESKTOP-Q3RSCJN MINGW64 /F:/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ arm-none-eabi-nm.exe app.o
00000000 T main
00000000 D string
00000000 R string2
00000000 U uart_send_string

salah@DESKTOP-Q3RSCJN MINGW64 /F:/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ arm-none-eabi-nm.exe uart.o
00000000 T uart_send_string

salah@DESKTOP-Q3RSCJN MINGW64 /F:/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ arm-none-eabi-nm.exe learn-in-depth.elf
00010010 T main
00010008 T reset
00011148 D stack_address
00010008 t stop
000100e4 D string
00010080 R string2
00010030 T uart_send_string

salah@DESKTOP-Q3RSCJN MINGW64 /F:/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ |
```

Using read-elf cross toolchain utility to show the entry point address.

```
MINGW64://EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
-I --histogram      Display histogram of bucket list lengths
-W --wide           Allow output width to exceed 80 characters
<File>             Read options from <File>
-H --help          Display this information
-V --version        Display the version number of readelf

salah@DESKTOP-QJRSJC7N MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ arm-none-eabi-readelf.exe -a learn-in-depth.elf
ELF Header:
  Magic:   7F 45 4C 01 01 01 00 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:
  Type:     EXEC (Executable file)
  Machine:  ARM
  Version:  0x1
  Entry point address: 0x10000
  Start of program headers: 52 (bytes into file)
  Start of section headers: 33240 (bytes into file)
  Flags:    0x0000002, 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          00000000         00000000         00 0 0 0
[ 1] .reset           PROGBITS         00010000          008000          000010         00 AX 0 0 4
[ 2] .text           PROGBITS         00010010          008010          000070         00 AX 0 0 4
[ 3] .rodata         PROGBITS         00010080          008080          000064         00 A 0 0 4
[ 4] .data           PROGBITS         000100e4          0080e4          000064         00 WA 0 0 4
[ 5] .ARM.attributes ARM_ATTRIBUTES   00000000          008148          00002e         00 0 0 1
[ 6] .comment        PROGBITS         00000000          008176          000011         01 MS 0 0 1
[ 7] .shstrtab       STRTAB           00000000          008187          00004F         00 0 0 1
[ 8] .symtab         SYMTAB           00000000          008368          0001a0         10 9 20 4
[ 9] .strtab        STRTAB           00000000          008508          00005c         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)
0 (extra OS processing required) o (OS specific), p (processor specific)

There are no section groups in this file.

Program Headers:
  Type           Offset             VirtAddr           PhysAddr          FileSiz MemSiz  Flg Align
  LOAD           0x008000           0x00010000         0x00010000         0x00148 0x00148  RWE 0x8000

Section to Segment mapping:
```

Run the program on the QEMU simulator.

```
MINGW64://EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1

salah@DESKTOP-QJRSJC7N MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ qemu-
Display all 67 possibilities? (y or n)
qemu-img.exe          qemu-system-i386.exe    qemu-system-mips64el.exe    qemu-system-ppc.exe      qemu-system-sh4.exe      qemu-system-x86_64.exe
qemu-io.exe           qemu-system-i386w.exe  qemu-system-mips64elw.exe  qemu-system-ppc64.exe    qemu-system-sh4ebw.exe   qemu-system-x86_64w.exe
qemu-system-aarch64.exe qemu-system-lm32.exe   qemu-system-mips64w.exe    qemu-system-ppc64w.exe   qemu-system-sh4ebw.exe   qemu-system-xtensa.exe
qemu-system-aarch64w.exe qemu-system-lm32w.exe  qemu-system-mipselw.exe    qemu-system-ppcembw.exe  qemu-system-sh4w.exe     qemu-system-xtensaebw.exe
qemu-system-alpha.exe  qemu-system-m68k.exe   qemu-system-mipselw.exe    qemu-system-ppcembw.exe  qemu-system-sparc.exe    qemu-system-xtensaebw.exe
qemu-system-alphaew.exe qemu-system-m68kw.exe  qemu-system-mips64w.exe    qemu-system-ppcw.exe     qemu-system-sparc64.exe  qemu-system-xtensaw.exe
qemu-system-arm.exe    qemu-system-microblaze.exe  qemu-system-moxie.exe     qemu-system-riscv32.exe  qemu-system-sparc64w.exe qemu-system-uninstall.exe
qemu-system-armw.exe   qemu-system-microblazeel.exe  qemu-system-moxiew.exe    qemu-system-riscv32w.exe qemu-system-sparcw.exe   qemu-system-xtensaw.exe
qemu-system-cris.exe   qemu-system-microblazeelw.exe  qemu-system-mos2.exe      qemu-system-riscv64.exe  qemu-system-tricore.exe  qemu-system-xtensaw.exe
qemu-system-crisw.exe  qemu-system-microblazew.exe  qemu-system-mos2w.exe     qemu-system-riscv64w.exe qemu-system-tricorew.exe  qemu-system-xtensaw.exe
qemu-system-hppa.exe   qemu-system-mips.exe     qemu-system-or1k.exe      qemu-system-s390w.exe    qemu-system-unicore32.exe
qemu-system-hppaw.exe  qemu-system-mips64.exe    qemu-system-or1kw.exe     qemu-system-s390xw.exe   qemu-system-unicore32w.exe

salah@DESKTOP-QJRSJC7N MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ qemu-system-arm -M versatilepb -m 128M -nographic -kernel learn-in-depth.bin
C:\Program Files (x86)\qemu\qemu-system-arm.exe: could not load kernel 'learn-in-depth.bin'

salah@DESKTOP-QJRSJC7N MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ arm-none-eabi-objcopy.exe -O learn-in-depth.elf learn-in-depth.bin
C:\ARM_Toolchain\bin\arm-none-eabi-objcopy.exe: 'learn-in-depth.bin': No such file

salah@DESKTOP-QJRSJC7N MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ arm-none-eabi-objcopy.exe -O binary learn-in-depth.elf learn-in-depth.bin

salah@DESKTOP-QJRSJC7N MINGW64 /F/EmbeddedSystemsDiploma/EmbeddedC_Lectures/lec10/lab1
$ qemu-system-arm -M versatilepb -m 128M -nographic -kernel learn-in-depth.bin
learn-in-depth:SALAH
```