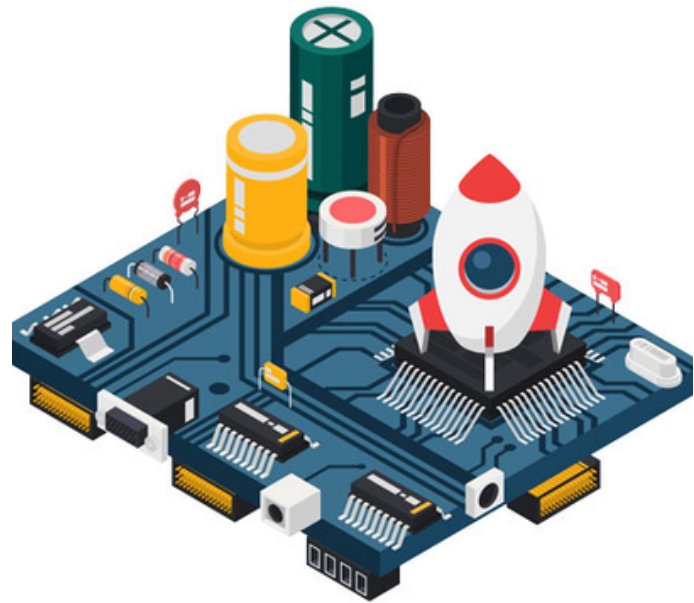


REPORT

LAB2



PREPARED BY
Mohamed kadry Hussien

LAB_2

–Physical Board : VersatilePB

–Processor : Arm926ej-s

5.2v

1–Writing source files, getting object files (without debugging)

```
MINGW64/f/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Embedded_C/Lec_2
mohammed kadry@victus MINGW64 /f/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Embedded_C/Lec_2 (main)
$ arm-none-eabi-gcc.exe -c -I . -mcpu=arm926ej-s app.c -o app.o
mohammed kadry@victus MINGW64 /f/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Embedded_C/Lec_2 (main)
$ arm-none-eabi-gcc.exe -c -I . -mcpu=arm926ej-s uart.c -o uart.o
```

2–analysis object files of (app.c and uart.c)

–app.o

```
MINGW64/f/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Embedded_C/Lec_2
mohammed kadry@victus MINGW64 /f/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Embedded_C/Lec_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          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
```

–uart.o

```
MINGW64/f/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Embedded_C/Lec_2
mohammed kadry@victus MINGW64 /f/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Embedded_C/Lec_2 (main)
$ 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

mohammed kadry@victus MINGW64 /f/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Embedded_C/Lec_2 (main)
$
```

3-Writing startup code, getting object file and analyzing it.

```
MINGW64:/f/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Embedded_C/Lec_2
mohammed kadry@victus MINGW64 /f/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Embedded_C/Lec_2 (main)
$ 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
```

4-analysis object file of startup file

```
$ 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
```

5-Writing the linker script, linking all objects, getting the elf file

```
MINGW64:/f/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Embedded_C/Lec_2
bedded_C/Lec_2 (main)
$ arm-none-eabi-ld.exe -T linker_script.ld startup.o app.o uart.o -o Mohamed_Kadry.elf -Map=Map_File.map
```

6-analysis elf file

```
MINGW64:/f/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Embedded_C/Lec_2
bedded_C/Lec_2 (main)
$ arm-none-eabi-objdump.exe -h Mohamed_Kadry.elf

Mohamed_Kadry.elf:      file format elf32-littlearm

Sections:
Idx Name          Size      VMA           LMA           File off  Algn
  0 .startup        00000010  00001000  00001000  00001000  2**2
    CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .text           00000068  00001010  00001010  00001010  2**2
    CONTENTS, ALLOC, LOAD, READONLY, CODE
  2 .data           00000064  00001078  00001078  00001078  2**2
    CONTENTS, ALLOC, LOAD, DATA
  3 .ARM.attributes 0000002e  00000000  00000000  000010dc  2**0
    CONTENTS, READONLY
  4 .comment         00000011  00000000  00000000  0000110a  2**0
    CONTENTS, READONLY
```

7-Getting the binary file of elf file

```
mohammed kadry@victus MINGW64 /f/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Embedded_C/Lec_2 (main)  
$ arm-none-eabi-objcopy.exe -O binary Mohamed_Kadry.elf Mohamed_Kadry.bin
```

8- simulation the application using QEMU

```
mohammed kadry@victus MINGW64 /f/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Embedded_C/Lec_2 (main)  
$ c:/qemu/qemu-system-arm -M versatilepb -m 128M -nographic -kernel Mohamed_Kadry.bin  
Learn-in-Depth:<Mohamed Kadry>
```

Additional Options

-we can also getting object files (with debugging)

```
MINGW64~/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Embedded_C/Lec_2
mohammed kadry@victus MINGW64 /f/Embedded Systems/Github/Master-Embedded-Systems
/Unit_3_Embedded_C/Lec_2 (main)
$ export PATH="C:\Program Files (x86)\GNU Tools ARM Embedded\7 2017-q4-major\bin
":$PATH

mohammed kadry@victus MINGW64 /f/Embedded Systems/Github/Master-Embedded-Systems
/Unit_3_Embedded_C/Lec_2 (main)
$ arm-none-eabi-gcc.exe -c -g -I . -mcpu=arm926ej-s app.c -o app.o

mohammed kadry@victus MINGW64 /f/Embedded Systems/Github/Master-Embedded-Systems
/Unit_3_Embedded_C/Lec_2 (main)
$ arm-none-eabi-gcc.exe -c -g -I . -mcpu=arm926ej-s uart.c -o uart.o

mohammed kadry@victus MINGW64 /f/Embedded Systems/Github/Master-Embedded-Systems
/Unit_3_Embedded_C/Lec_2 (main)
$
```

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

```
MINGW64~/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Embedded_C/Lec_2
mohammed kadry@victus MINGW64 /f/Embedded Systems/Github/Master-Embedded-Systems
/Unit_3_Embedded_C/Lec_2 (main)
$ arm-none-eabi-readelf.exe -a Mohamed_Kadry.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  0  1
[ 5] .comment              PROGBITS      00000000  00810a  000011  01  MS  0  0  1
[ 6] .shstrtab             STRTAB        00000000  00811b  000049  00  0  0  0  1
[ 7] .symtab               SYMTAB        00000000  0082cc  000170  10  8 18  4
[ 8] .strtab               STRTAB        00000000  00843c  000057  00  0  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)

There are no section groups in this file.
```

-Getting the symbol table for the object files

```
mingw64~/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Embedded_C/Lec_2
mohammed kadry@victus MINGW64 /f/Embedded Systems/Github/Master-Embedded-Systems
/Unit_3_Embedded_C/Lec_2 (main)
$ arm-none-eabi-nm.exe app.o
00000000 T main
00000000 D string_buffer
00000000 U Uart_Send_String

mohammed kadry@victus MINGW64 /f/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Em
bedded_C/Lec_2 (main)
$ arm-none-eabi-nm.exe uart.o
00000000 T Uart_Send_String

mohammed kadry@victus MINGW64 /f/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Em
bedded_C/Lec_2 (main)
$ arm-none-eabi-nm.exe startup.o
00000000 U main
00000000 T reset
00000000 U stack_top
00000008 t stop

mohammed kadry@victus MINGW64 /f/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Em
bedded_C/Lec_2 (main)
$ arm-none-eabi-nm.exe Mohamed_Kadry.elf
00010010 T main
00010000 T reset
000110dc D stack_top
00010008 t stop
00010078 D string_buffer
00010028 T Uart_Send_String

mohammed kadry@victus MINGW64 /f/Embedded Systems/Github/Master-Embedded-Systems/Unit_3_Em
bedded_C/Lec_2 (main)
$ |
```