Unit 3 Lesson 4

Lab 3

Sections:

```
PS E:\Courses\Embedded KS\Programs\ARM_TOOLCHAIN\bin\APP> arm-none-eabi-objdump.exe -h Unit3_less4_T_Led_M4.elf

Unit3_less4_T_Led_M4.elf:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         000001b0  00000000  00000000  00008000  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .bss          00000400  20000000  000001b0  00010000  2**2
                  ALLOC
  2 .debug_info   00000218  00000000  00000000  000081b0  2**0
                  CONTENTS, READONLY, DEBUGGING
  3 .debug_abbrev 00000124  00000000  00000000  000083c8  2**0
                  CONTENTS, READONLY, DEBUGGING
  4 .debug_loc    00000070  00000000  00000000  000084ec  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_aranges 00000040  00000000  00000000  0000855c  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_line   000000e6  00000000  00000000  0000859c  2**0
                  CONTENTS, READONLY, DEBUGGING
  7 .debug_str    0000013f  00000000  00000000  00008682  2**0
                  CONTENTS, READONLY, DEBUGGING
  8 .comment      00000011  00000000  00000000  000087c1  2**0
                  CONTENTS, READONLY
  9 .ARM.attributes 00000033  00000000  00000000  000087d2  2**0
                  CONTENTS, READONLY
 10 .debug_frame  0000005c  00000000  00000000  00008808  2**2
                  CONTENTS, READONLY, DEBUGGING
```

As we can see in the previous image the LMA and VMA of .text section is the same because both are located in the flash memory, but the address of the .bss section is not the same because the startup code copied the values from flash to the SRAB.

Symbol Table:

```
PS E:\Courses\Embedded KS\Programs\ARM_TOOLCHAIN\bin\APP> arm-none-eabi-nm.exe Unit3_less4_T_Led_M4.elf
20000400 B _e_bss
20000000 T _e_data
000001b0 T _e_text
20000000 B _s_bss
20000000 T _s_data
00000058 W BusFault_Handler
00000058 W DebugMonitor_Handler
00000058 W EXTI0_Handler
00000058 W EXTI1_Handler
00000058 W EXTI2_Handler
00000058 W FLASH_Handler
00000000 T g_arr_p_fn
00000058 W HardFault_Handler
00000118 T main
00000058 W MM_mange_Handler
00000058 W NMI_Handler
00000058 W PendSV_Handler
00000058 W PVD_Handler
00000058 W RCC_Handler
00000058 W Reserved_Handler
00000058 T Reset_Handler
00000058 W RTC_Handler
20000000 b stack_top
00000058 W SVCall_Handler
00000058 W SysTick_Handler
00000058 W TAMPER_Handler
00000058 W UsageFault_Handler
00000058 W WWDG_Handler
```

This image shows the symbols found in the elf file and their address as we can see most of the handlers “Interrupt vector table” the not used ones are under the same address as we used the preprocessor “alias”.

Where,

T is abbreviation for .text (code)

B is abbreviation for .bss (uninitialized data)

W is abbreviation for Weak Symbol

Read elf file:

```
PS E:\Courses\Embedded KS\Programs\ARM_TOOLCHAIN\bin\APP> arm-none-eabi-readelf.exe -a Unit3_less4_T_Led_M4.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:               0x0
  Start of program headers:          52 (bytes into file)
  Start of section headers:          35068 (bytes into file)
  Flags:                             0x5000000, Version5 EABI
  Size of this header:               52 (bytes)
  Size of program headers:           32 (bytes)
  Number of program headers:         2
  Size of section headers:           40 (bytes)
  Number of section headers:         15
  Section header string table index: 12

Section Headers:
  [Nr] Name              Type            Addr     Off    Size   ES Flg Lk Inf Al
  [ 0]                   NULL            00000000 000000 000000 00      0   0  0
  [ 1] .text             PROGBITS        00000000 008000 0001b0 00  AX  0   0  4
  [ 2] .bss              NOBITS          20000000 010000 000400 00  WA  0   0  4
  [ 3] .debug_info       PROGBITS        00000000 0081b0 000218 00      0   0  1
  [ 4] .debug_abbrev     PROGBITS        00000000 0083c8 000124 00      0   0  1
  [ 5] .debug_loc        PROGBITS        00000000 0084ec 000070 00      0   0  1
  [ 6] .debug_aranges    PROGBITS        00000000 00855c 000040 00      0   0  1
  [ 7] .debug_line       PROGBITS        00000000 00859c 0000e6 00      0   0  1
  [ 8] .debug_str        PROGBITS        00000000 008682 00013f 01  MS  0   0  1
  [ 9] .comment          PROGBITS        00000000 0087c1 000011 01  MS  0   0  1
  [10] .ARM.attributes   ARM_ATTRIBUTES  00000000 0087d2 000033 00      0   0  1
  [11] .debug_frame      PROGBITS        00000000 008808 00005c 00      0   0  4
  [12] .shstrtab         STRTAB          00000000 008864 000097 00      0   0  1
  [13] .symtab           SYMTAB          00000000 008b54 000310 10     14  22  4
```

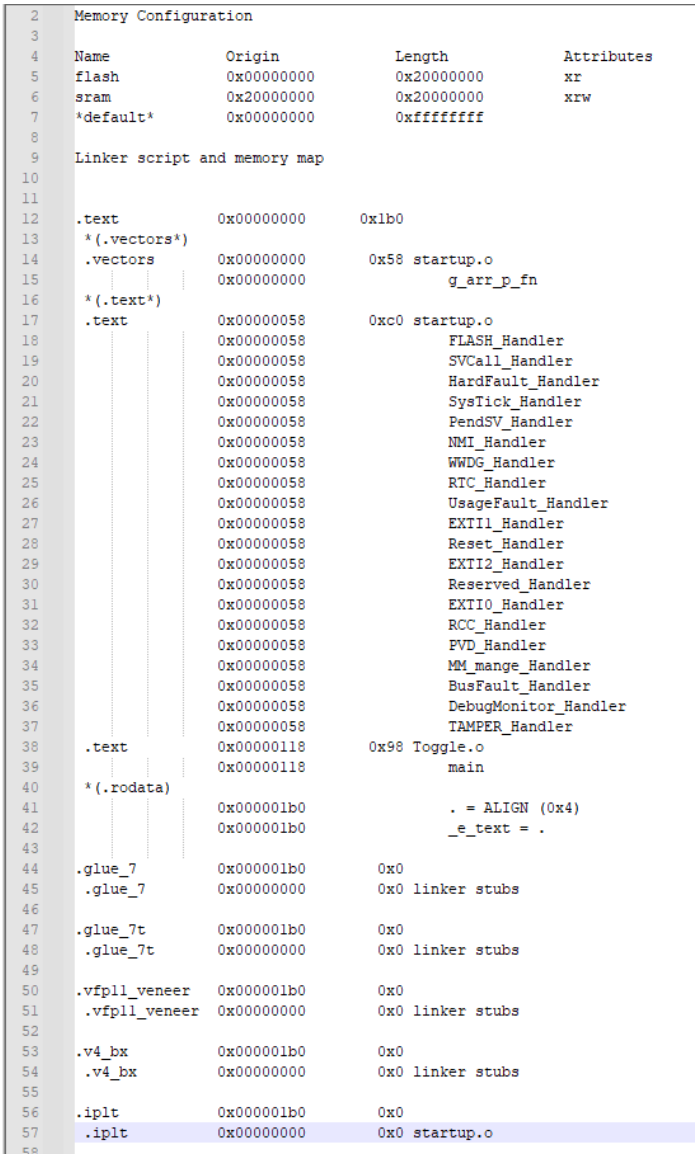
As we can see the start address of .text is at 0x00000000 which is the same as the start address of the flash memory as stated in the linker and the start of the bss section is 0x20000000 which is the same as the start address of the SRAM memory.

```
  [12] .shstrtab         STRTAB          00000000 008864 000097 00      0   0  1
  [13] .symtab           SYMTAB          00000000 008b54 000310 10     14  22  4
  [14] .strtab           STRTAB          00000000 008e64 000187 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)

There are no section groups in this file.

Program Headers:
  Type           Offset   VirtAddr   PhysAddr   FileSiz MemSiz  Flg Align
  LOAD           0x008000 0x00000000 0x00000000 0x001b0 0x001b0 R E 0x8000
  LOAD           0x010000 0x20000000 0x000001b0 0x00000 0x00400 RW  0x8000

 Section to Segment mapping:
  Segment Sections...
   00     .text
   01     .bss

There is no dynamic section in this file.

There are no relocations in this file.

There are no unwind sections in this file.

Symbol table '.symtab' contains 49 entries:
   Num:    Value  Size Type    Bind   Vis      Ndx Name
     0: 00000000     0 NOTYPE  LOCAL  DEFAULT  UND
     1: 00000000     0 SECTION LOCAL  DEFAULT    1
     2: 20000000     0 SECTION LOCAL  DEFAULT    2
     3: 00000000     0 SECTION LOCAL  DEFAULT    3
     4: 00000000     0 SECTION LOCAL  DEFAULT    4
     5: 00000000     0 SECTION LOCAL  DEFAULT    5
     6: 00000000     0 SECTION LOCAL  DEFAULT    6
     7: 00000000     0 SECTION LOCAL  DEFAULT    7
     8: 00000000     0 SECTION LOCAL  DEFAULT    8
     9: 00000000     0 SECTION LOCAL  DEFAULT    9
```

```
     9: 00000000     0 SECTION LOCAL  DEFAULT    9
    10: 00000000     0 SECTION LOCAL  DEFAULT   10
    11: 00000000     0 SECTION LOCAL  DEFAULT   11
    12: 00000000     0 FILE    LOCAL  DEFAULT  ABS startup.c
    13: 20000000     0 NOTYPE  LOCAL  DEFAULT    2 $d
    14: 20000000     0 NOTYPE  LOCAL  DEFAULT    2 stack_top
    15: 00000000     0 NOTYPE  LOCAL  DEFAULT    1 $d
    16: 00000058     0 NOTYPE  LOCAL  DEFAULT    1 $t
    17: 00000010     0 NOTYPE  LOCAL  DEFAULT   11 $d
    18: 00000000     0 FILE    LOCAL  DEFAULT  ABS Toggle.c
    19: 00000118     0 NOTYPE  LOCAL  DEFAULT    1 $t
    20: 00000040     0 NOTYPE  LOCAL  DEFAULT   11 $d
    21: 00000000     0 FILE    LOCAL  DEFAULT  ABS
    22: 00000000    88 OBJECT  GLOBAL DEFAULT    1 g_arr_p_fn
    23: 00000059   192 FUNC    WEAK   DEFAULT    1 FLASH_Handler
    24: 00000059   192 FUNC    WEAK   DEFAULT    1 SVCall_Handler
    25: 00000059   192 FUNC    WEAK   DEFAULT    1 HardFault_Handler
    26: 00000059   192 FUNC    WEAK   DEFAULT    1 SysTick_Handler
    27: 00000059   192 FUNC    WEAK   DEFAULT    1 PendSV_Handler
    28: 00000059   192 FUNC    WEAK   DEFAULT    1 NMI_Handler
    29: 00000059   192 FUNC    WEAK   DEFAULT    1 WWDG_Handler
    30: 000001b0     0 NOTYPE  GLOBAL DEFAULT    1 _e_text
    31: 00000059   192 FUNC    WEAK   DEFAULT    1 RTC_Handler
    32: 00000059   192 FUNC    WEAK   DEFAULT    1 UsageFault_Handler
    33: 20000000     0 NOTYPE  GLOBAL DEFAULT    2 _s_bss
    34: 20000000     0 NOTYPE  GLOBAL DEFAULT    1 _s_data
    35: 20000000     0 NOTYPE  GLOBAL DEFAULT    1 _e_data
    36: 00000059   192 FUNC    WEAK   DEFAULT    1 EXTI1_Handler
    37: 00000059   192 FUNC    GLOBAL DEFAULT    1 Reset_Handler
    38: 00000059   192 FUNC    WEAK   DEFAULT    1 EXTI2_Handler
    39: 20000400     0 NOTYPE  GLOBAL DEFAULT    2 _e_bss
    40: 00000059   192 FUNC    WEAK   DEFAULT    1 Reserved_Handler
    41: 00000119   152 FUNC    GLOBAL DEFAULT    1 main
    42: 00000059   192 FUNC    WEAK   DEFAULT    1 EXTI0_Handler
    43: 00000059   192 FUNC    WEAK   DEFAULT    1 RCC_Handler
    44: 00000059   192 FUNC    WEAK   DEFAULT    1 PVD_Handler
    45: 00000059   192 FUNC    WEAK   DEFAULT    1 MM_mange_Handler
    46: 00000059   192 FUNC    WEAK   DEFAULT    1 BusFault_Handler
```

```
    46: 00000059   192 FUNC    WEAK   DEFAULT    1 BusFault_Handler
    47: 00000059   192 FUNC    WEAK   DEFAULT    1 DebugMonitor_Handler
    48: 00000059   192 FUNC    WEAK   DEFAULT    1 TAMPER_Handler

No version information found in this file.
Attribute Section: aeabi
File Attributes
  Tag_CPU_name: "Cortex-M4"
  Tag_CPU_arch: v7E-M
  Tag_CPU_arch_profile: Microcontroller
  Tag_THUMB_ISA_use: Thumb-2
  Tag_ABI_PCS_wchar_t: 4
  Tag_ABI_FP_denormal: Needed
  Tag_ABI_FP_exceptions: Needed
  Tag_ABI_FP_number_model: IEEE 754
  Tag_ABI_align_needed: 8-byte
  Tag_ABI_align_preserved: 8-byte, except leaf SP
  Tag_ABI_enum_size: small
  Tag_ABI_optimization_goals: Aggressive Debug
  Tag_CPU_unaligned_access: v6
```

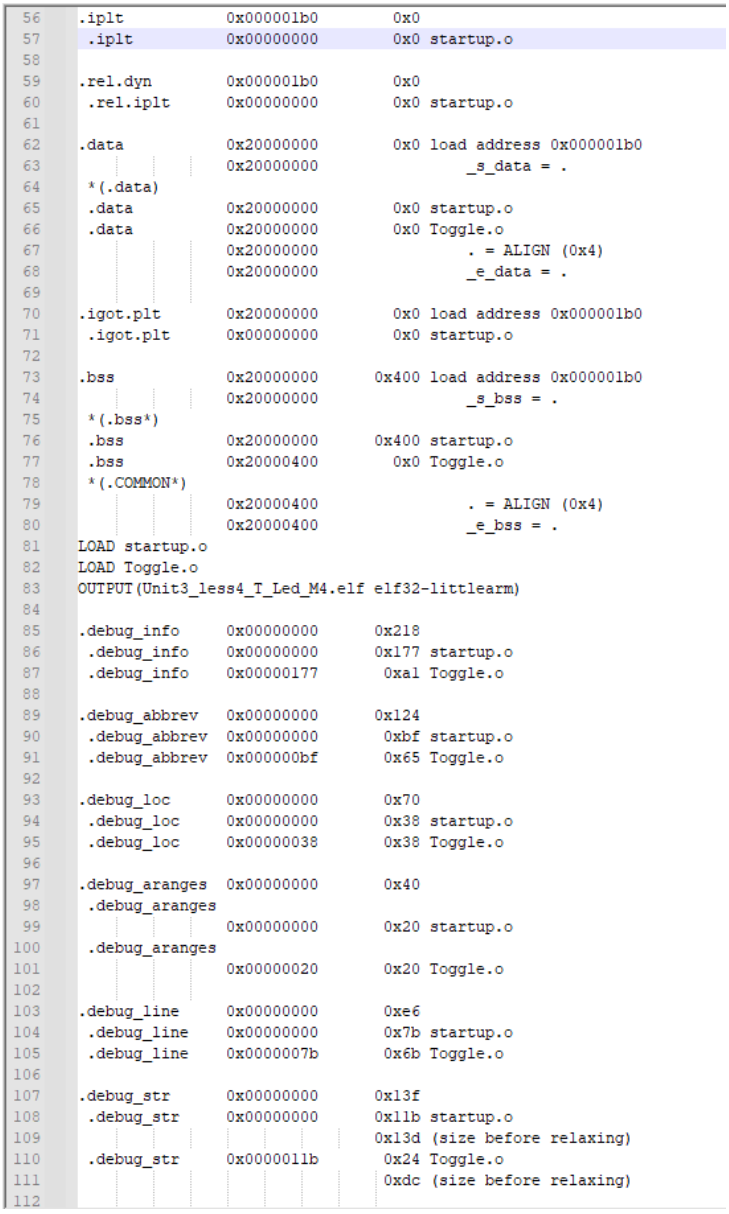
Map File:



```
Memory Configuration

Name             Origin             Length             Attributes
flash            0x00000000         0x20000000         xr
sram             0x20000000         0x20000000         xrw
*default*        0x00000000         0xffffffff

Linker script and memory map


.text           0x00000000      0x1b0
 *(.vectors*)
 .vectors       0x00000000       0x58 startup.o
                0x00000000                g_arr_p_fn
 *(.text*)
 .text          0x00000058       0xc0 startup.o
                0x00000058                FLASH_Handler
                0x00000058                SVCall_Handler
                0x00000058                HardFault_Handler
                0x00000058                SysTick_Handler
                0x00000058                PendSV_Handler
                0x00000058                NMI_Handler
                0x00000058                WWDG_Handler
                0x00000058                RTC_Handler
                0x00000058                UsageFault_Handler
                0x00000058                EXTI1_Handler
                0x00000058                Reset_Handler
                0x00000058                EXTI2_Handler
                0x00000058                Reserved_Handler
                0x00000058                EXTI0_Handler
                0x00000058                RCC_Handler
                0x00000058                PVD_Handler
                0x00000058                MM_mange_Handler
                0x00000058                BusFault_Handler
                0x00000058                DebugMonitor_Handler
                0x00000058                TAMPER_Handler
 .text          0x00000118       0x98 Toggle.o
                0x00000118                main
 *(.rodata)
                0x000001b0                . = ALIGN (0x4)
                0x000001b0                _e_text = .

.glue_7         0x000001b0        0x0
 .glue_7        0x00000000        0x0 linker stubs

.glue_7t        0x000001b0        0x0
 .glue_7t       0x00000000        0x0 linker stubs

.vfp11_veneer   0x000001b0        0x0
 .vfp11_veneer  0x00000000        0x0 linker stubs

.v4_bx          0x000001b0        0x0
 .v4_bx         0x00000000        0x0 linker stubs

.iplt           0x000001b0        0x0
 .iplt          0x00000000        0x0 startup.o
```

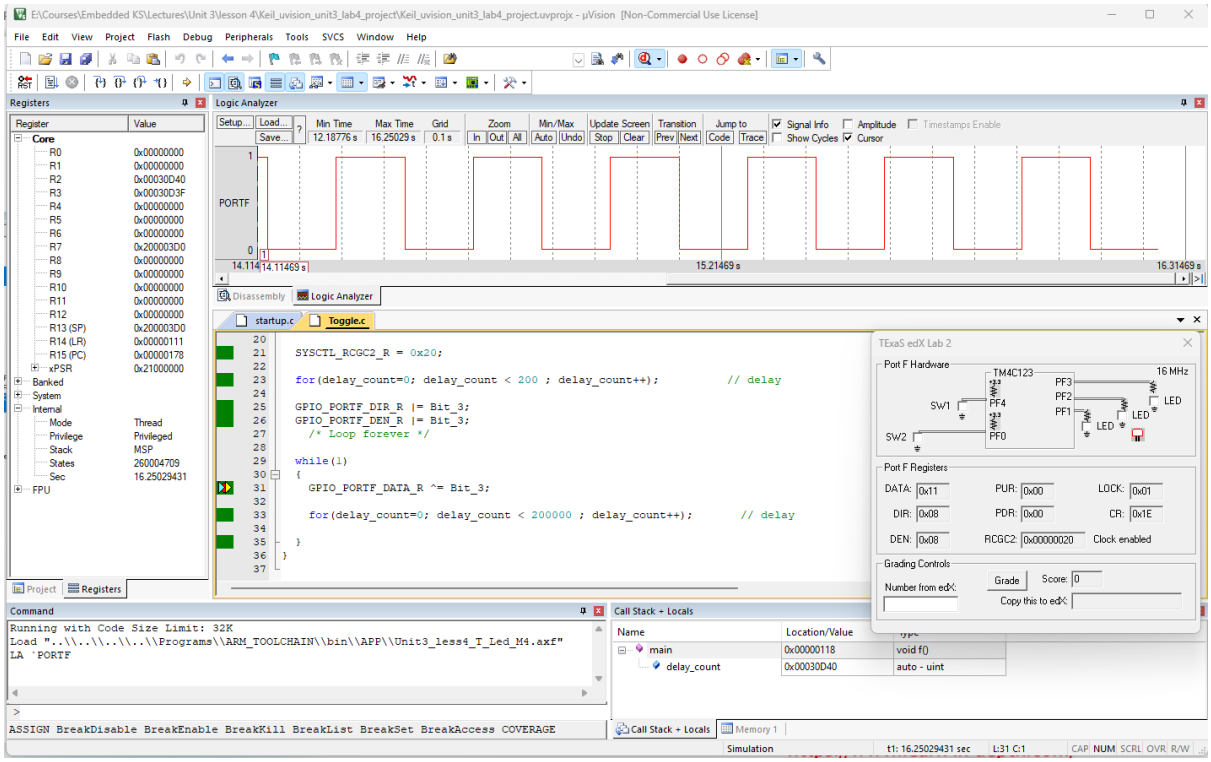
This photo shows the first address of the flash memory and the SRAM and their length, also shows where the vectors are stored and the text and the rodata if it is found and the symbols with respect to their address.

The next photo shows the loading address of the start SRAM section which is here .data but we have no values to be stored so the next section will be found which is the .bss and most of the rest is for the debugging information.



```
.iplt           0x000001b0        0x0
 .iplt          0x00000000        0x0 startup.o

.rel.dyn        0x000001b0        0x0
 .rel.iplt      0x00000000        0x0 startup.o

.data           0x20000000        0x0 load address 0x000001b0
                0x20000000                _s_data = .
 *(.data)
 .data          0x20000000        0x0 startup.o
 .data          0x20000000        0x0 Toggle.o
                0x20000000                . = ALIGN (0x4)
                0x20000000                _e_data = .

.igot.plt       0x20000000        0x0 load address 0x000001b0
 .igot.plt      0x00000000        0x0 startup.o

.bss            0x20000000      0x400 load address 0x000001b0
                0x20000000                _s_bss = .
 *(.bss*)
 .bss           0x20000000      0x400 startup.o
 .bss           0x20000400        0x0 Toggle.o
 *(.COMMON*)
                0x20000400                . = ALIGN (0x4)
                0x20000400                _e_bss = .
LOAD startup.o
LOAD Toggle.o
OUTPUT(Unit3_less4_T_Led_M4.elf elf32-littlearm)

.debug_info     0x00000000      0x218
 .debug_info    0x00000000      0x177 startup.o
 .debug_info    0x00000177       0xa1 Toggle.o

.debug_abbrev   0x00000000      0x124
 .debug_abbrev  0x00000000       0xbf startup.o
 .debug_abbrev  0x000000bf       0x65 Toggle.o

.debug_loc      0x00000000       0x70
 .debug_loc     0x00000000       0x38 startup.o
 .debug_loc     0x00000038       0x38 Toggle.o

.debug_aranges  0x00000000       0x40
 .debug_aranges
                0x00000000       0x20 startup.o
 .debug_aranges
                0x00000020       0x20 Toggle.o

.debug_line     0x00000000       0xe6
 .debug_line    0x00000000       0x7b startup.o
 .debug_line    0x0000007b       0x6b Toggle.o

.debug_str      0x00000000      0x13f
 .debug_str     0x00000000      0x11b startup.o
                                0x13d (size before relaxing)
 .debug_str     0x0000011b       0x24 Toggle.o
                                 0xdc (size before relaxing)

.debug_str      0x0000011b       0x24 Toggle.o
                                 0xdc (size before relaxing)

.comment        0x00000000       0x11
 .comment       0x00000000       0x11 startup.o
                                 0x12 (size before relaxing)
 .comment       0x00000000       0x12 Toggle.o

.ARM.attributes
                0x00000000       0x33
 .ARM.attributes
                0x00000000       0x33 startup.o
 .ARM.attributes
                0x00000033       0x33 Toggle.o

.debug_frame    0x00000000       0x5c
 .debug_frame   0x00000000       0x30 startup.o
 .debug_frame   0x00000030       0x2c Toggle.o
```

Keil vision:



This photo shows the simulation of toggling a LED of Port F pin 3 on TM4c123 “Tiva C” as we can see the 0 and 1 from the logic analyzer and the TExaS simulation.