Unit3. lesson3. lab2. Report

Eng. Farha Emad Mohamed

main.c

```
#include <stdint.h>
#define RCC  0x40021000
#define PortA 0x40010800
#define PortA_CRH   *((volatile uint32_t*)(PortA+0x04))
#define PortA_ODR   *((volatile uint32_t*)(PortA+0x0C))
#define RCC_APB2ENR *((volatile uint32_t*)(RCC+0x18))
/** another way to access ODR **/
typedef union{
        struct{
                volatile uint32_t Reserved:13;
                volatile uint32_t Pin13:1;
        }SPIN;
        volatile uint32_t All_Pins;

}UODR_t;
volatile UODR_t* R_ODR = (volatile UODR_t*)(PortA + 0x0C);
```

```
int main(void)
{
        RCC_APB2ENR |= (1<<2);
        PortA_CRH &= 0xFF0FFFFF;
        PortA_CRH |= 0x00200000;
        int i;
        while(1)
        {
                PortA_ODR |= (1<<13);
                for(i=0 ; i<50000 ; i++);
                PortA_ODR &= ~(1<<13);
                for(i=0 ; i<5000 ; i++);
        }

}
```

startup.c

```
#include <stdint.h>
extern uint32_t STACK_TOP;
extern void main();
void Reset_handler(void);
void Default_handler(){
        Reset_handler();
}
void NMI_handler() __attribute__((weak,alias("Default_handler")));
void HardFault_handler() __attribute__((weak,alias("Default_handler")));
void MMFault_handler() __attribute__((weak,alias("Default_handler")));
void BusFault_handler() __attribute__((weak,alias("Default_handler")));
void UsageFault_handler() __attribute__((weak,alias("Default_handler")));

uint32_t vector [] __attribute__((section(".vectors")))= {
        (uint32_t) &STACK_TOP,
        (uint32_t) &Reset_handler,
        (uint32_t) &NMI_handler,
        (uint32_t) &HardFault_handler,
        (uint32_t) &MMFault_handler,
        (uint32_t) &BusFault_handler,
        (uint32_t) &UsageFault_handler
};
```

```
uint32_t i;
extern uint32_t _E_text;
extern uint32_t _S_data;
extern uint32_t _E_data;
extern uint32_t _S_bss;
extern uint32_t _E_bss;
void Reset_handler(void){
        /*copying .data from Flash to RAM*/
        uint32_t _data_size = (uint8_t*)&_E_data - (uint8_t*)&_S_data;
        uint8_t *ptr_scr = &_E_text;
        uint8_t *ptr_dest = &_S_data;
        for(i=0 ; i< _data_size ; i++){
                *((uint8_t*)ptr_dest++) = *((uint8_t*)ptr_scr++);
        }
        /*create .bss section*/
        uint32_t _bss_size = (uint8_t*)&_E_bss - (uint8_t*)&_S_bss;
        ptr_dest = &_S_data;
        for(i=0 ; i< _data_size ; i++)
        {
                *((uint8_t*)ptr_dest++) = (uint8_t*)0;
        }
        /*branching to main*/
        main();
}
```

linker_script.ld

```
MEMORY
{
        Flash (RX) : ORIGIN = 0x08000000 ,LENGTH = 128K
        SRAM (RWX): ORIGIN = 0x20000000 ,LENGTH = 20K
}

SECTIONS
{
        .text :{
        *(.vectors*)
        *(.text*)
        *(.rodata*)
        _E_text = .;
        }>Flash

        .data :{
        _S_data = .;
        *(.data*)
        _E_data = .;
        }>SRAM AT> Flash

        .bss :{
        _S_bss = .;
        *(.bss*)
        _E_bss = .;
        . = ALIGN(4);
        . = . + 1000;
        STACK_TOP = .;
        }>SRAM
}
```

main.o sections

```
WIN 10@DESKTOP-BHGVA79 MINGW32 /d/Embedded/Learn_in_Depth/Unit3_embedded_c/EmbeddedC_lesson3/Lab2
$ arm-none-eabi-objdump.exe -h main.o

main.o:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         0000007c  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .data         00000004  00000000  00000000  000000b0  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  000000b4  2**0
                  ALLOC
  3 .debug_info   00000120  00000000  00000000  000000b4  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  4 .debug_abbrev 000000bd  00000000  00000000  000001d4  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_loc    00000038  00000000  00000000  00000291  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_aranges 00000020  00000000  00000000  000002c9  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  7 .debug_line   0000012b  00000000  00000000  000002e9  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  8 .debug_str    0000017c  00000000  00000000  00000414  2**0
                  CONTENTS, READONLY, DEBUGGING
  9 .comment      0000007f  00000000  00000000  00000590  2**0
                  CONTENTS, READONLY
 10 .debug_frame  0000002c  00000000  00000000  00000610  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
 11 .ARM.attributes 00000033  00000000  00000000  0000063c  2**0
                  CONTENTS, READONLY
```

main.o symbols

```
WIN 10@DESKTOP-BHGVA79 MINGW32 /d/Embedded/Learn_in_Depth/Unit3_embedded_c/EmbeddedC_lesson3/Lab2
$ arm-none-eabi-nm.exe main.o
00000000 T main
00000000 D R_ODR
```

startup.o sections

```
WIN 10@DESKTOP-BHGVA79 MINGW32 /d/Embedded/Learn_in_Depth/Unit3_embedded_c/EmbeddedC_lesson3/Lab2
$ 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         000000a4  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000000  00000000  00000000  000000d8  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  000000d8  2**0
                  ALLOC
  3 .vectors      0000001c  00000000  00000000  000000d8  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, DATA
  4 .debug_info   00000186  00000000  00000000  000000f4  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  5 .debug_abbrev 000000c4  00000000  00000000  0000027a  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_loc    0000007c  00000000  00000000  0000033e  2**0
                  CONTENTS, READONLY, DEBUGGING
  7 .debug_aranges 00000020  00000000  00000000  000003ba  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  8 .debug_line   0000013c  00000000  00000000  000003da  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  9 .debug_str    000001dd  00000000  00000000  00000516  2**0
                  CONTENTS, READONLY, DEBUGGING
 10 .comment      0000007f  00000000  00000000  000006f3  2**0
                  CONTENTS, READONLY
 11 .debug_frame  00000050  00000000  00000000  00000774  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
 12 .ARM.attributes 00000033  00000000  00000000  000007c4  2**0
                  CONTENTS, READONLY
```

startup.o symbols

```
WIN 10@DESKTOP-BHGVA79 MINGW32 /d/Embedded/Learn_in_Depth/Unit3_embedded_c/EmbeddedC_lesson3/Lab2
$ arm-none-eabi-nm.exe startup.o
         U _E_bss
         U _E_data
         U _E_text
         U _S_bss
         U _S_data
00000000 W BusFault_handler
00000000 T Default_handler
00000000 W HardFault_handler
00000004 C i
         U main
00000000 W MMFault_handler
00000000 W NMI_handler
0000000c T Reset_handler
         U STACK_TOP
00000000 W UsageFault_handler
00000000 D vector
```

toggle_led.elf sections

```
WIN 10@DESKTOP-BHGVA79 MINGW32 /d/Embedded/Learn_in_Depth/Unit3_embedded_c/EmbeddedC_lesson3/Lab2
$ arm-none-eabi-objdump.exe -h toggle_led.elf

toggle_led.elf:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         0000013c  08000000  08000000  00010000  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .data         00000004  20000000  0800013c  00020000  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          000003ec  20000004  08000140  00020004  2**2
                  ALLOC
  3 .debug_info   000002a6  00000000  00000000  00020004  2**0
                  CONTENTS, READONLY, DEBUGGING
  4 .debug_abbrev 00000181  00000000  00000000  000202aa  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_loc    000000b4  00000000  00000000  0002042b  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_aranges 00000040  00000000  00000000  000204df  2**0
                  CONTENTS, READONLY, DEBUGGING
  7 .debug_line   00000267  00000000  00000000  0002051f  2**0
                  CONTENTS, READONLY, DEBUGGING
  8 .debug_str    000001ce  00000000  00000000  00020786  2**0
                  CONTENTS, READONLY, DEBUGGING
  9 .comment      0000007e  00000000  00000000  00020954  2**0
                  CONTENTS, READONLY
 10 .ARM.attributes 00000033  00000000  00000000  000209d2  2**0
                  CONTENTS, READONLY
 11 .debug_frame  0000007c  00000000  00000000  00020a08  2**2
                  CONTENTS, READONLY, DEBUGGING
```

toggle_led.elf symbols

```
WIN 10@DESKTOP-BHGVA79 MINGW32 /d/Embedded/Learn_in_Depth/Unit3_embedded_c/EmbeddedC_lesson3/Lab2
$ arm-none-eabi-nm.exe toggle_led.elf
20000004 B _E_bss
20000004 D _E_data
0800013c T _E_text
20000004 B _S_bss
20000000 D _S_data
08000098 W BusFault_handler
08000098 T Default_handler
08000098 W HardFault_handler
200003ec B i
0800001c T main
08000098 W MMFault_handler
08000098 W NMI_handler
20000000 D R_ODR
080000a4 T Reset_handler
200003ec B STACK_TOP
08000098 W UsageFault_handler
08000000 T vector
```

Led_toggle - Proteus 8 Professional - Schematic Capture

File Edit View Tool Design Graph Debug Library Template System Help

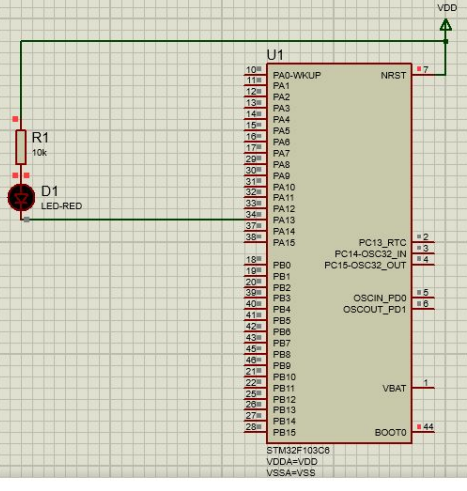
test3.c | test1.c | linker_script.ld | startup.c | startup.s-orig | linker_script.ld | MakeFile | fifo.c | fifo.h | main.c | map_file.map

```
58  .data           0x20000004        0x0 startup.o
59                  0x20000004                _E_data = .
60
61  .igot.plt       0x20000004        0x0 load address 0x08000140
62  .igot.plt       0x20000004        0x0 main.o
63
64  .bss            0x20000004        0x0 load address 0x08000140
65                  0x20000004                _S_bss = .
66  *(.bss*)
67  .bss            0x20000004        0x0 main.o
68  .bss            0x20000004        0x0 startup.o
69                  0x20000004                _E_bss = .
70                  0x20000004                . = ALIGN (0x4)
71                  0x200003ec                . = (. + 0x3e8)
72  *fill*          0x20000004      0x3e8
73                  0x200003ec                STACK_TOP = .
74  COMMON          0x200003ec        0x4 startup.o
75                  0x200003ec                i
```

CM3 Source Code - U1

..\EmbeddedC_lesson3\Lab2\startup.c

```
-------- void MMFault_handler() __attribute__((weak,alias("Default_h
-------- void BusFault_handler() __attribute__((weak,alias("Default_
-------- void UsageFault_handler() __attribute__((weak,alias("Defaul
--------
-------- uint32_t vector [] __attribute__((section(".vectors")))= {
--------         (uint32_t) &STACK_TOP,
--------         (uint32_t) &Reset_handler,
--------         (uint32_t) &NMI_handler,
--------         (uint32_t) &HardFault_handler,
--------         (uint32_t) &MMFault_handler,
--------         (uint32_t) &BusFault_handler,
--------         (uint32_t) &UsageFault_handler
-------- };
-------- uint32_t i;
-------- extern uint32_t _E_text;
-------- extern uint32_t _S_data;
-------- extern uint32_t _E_data;
-------- extern uint32_t _S_bss;
-------- extern uint32_t _E_bss;
--------
80000A4  void Reset_handler(void){
--------         /*copying .data from Flash to RAM*/
80000AA          uint32_t _data_size = (uint8_t*)&_E_data - (uint8_t
80000B2          uint8_t *ptr_scr = &_E_text;
80000B6          uint8_t *ptr_dest = &_S_data;
80000BA          for(i=0 ; i< _data_size ; i++)
--------         {
80000C2                  *((uint8_t*)ptr_dest++) = *((uint8_t*)ptr_s
--------         }
--------
--------         /*create .bss section*/
80000E6          uint32_t _bss_size = (uint8_t*)&_E_bss - (uint8_t*)
80000EE          ptr_dest = &_S_data;
80000F2          for(i=0 ; i< _data_size ; i++)
--------         {
80000FA                  *((uint8_t*)ptr_dest++) = (uint8_t*)0;
--------         }
--------
--------         /*branching to main*/
80000118         main();
800011C  }
--------
--------
```



CM3 RAM at 0x20000000 - U1

```
20000390  00 00 00 00  ....
20000394  00 00 00 00  ....
20000398  00 00 00 00  ....
2000039C  00 00 00 00  ....
200003A0  00 00 00 00  ....
200003A4  00 00 00 00  ....
200003A8  00 00 00 00  ....
200003AC  00 00 00 00  ....
200003B0  00 00 00 00  ....
200003B4  00 00 00 00  ....
200003B8  00 00 00 00  ....
200003BC  00 00 00 00  ....
200003C0  00 00 00 00  ....
200003C4  00 00 00 00  ....
200003C8  00 00 00 00  ....
200003CC  00 00 00 00  ....
200003D0  00 00 00 00  ....
200003D4  00 00 00 00  ....
200003D8  04 00 00 00  ....
200003DC  04 00 00 20  ...
200003E0  40 01 00 08  @..
200003E4  00 00 00 00  ....
200003E8  FF FF FF FF  ....
200003EC  04 00 00 00  ....
200003F0  00 00 00 00  ....
200003F4  00 00 00 00  ....
200003F8  00 00 00 00  ....
200003FC  00 00 00 00  ....
20000400  00 00 00 00  ....
20000404  00 00 00 00  ....
20000408  00 00 00 00  ....
2000040C  00 00 00 00  ....
20000410  00 00 00 00  ....
20000414  00 00 00 00  ....
20000418  00 00 00 00  ....
2000041C  00 00 00 00  ....
20000420  00 00 00 00  ....
```

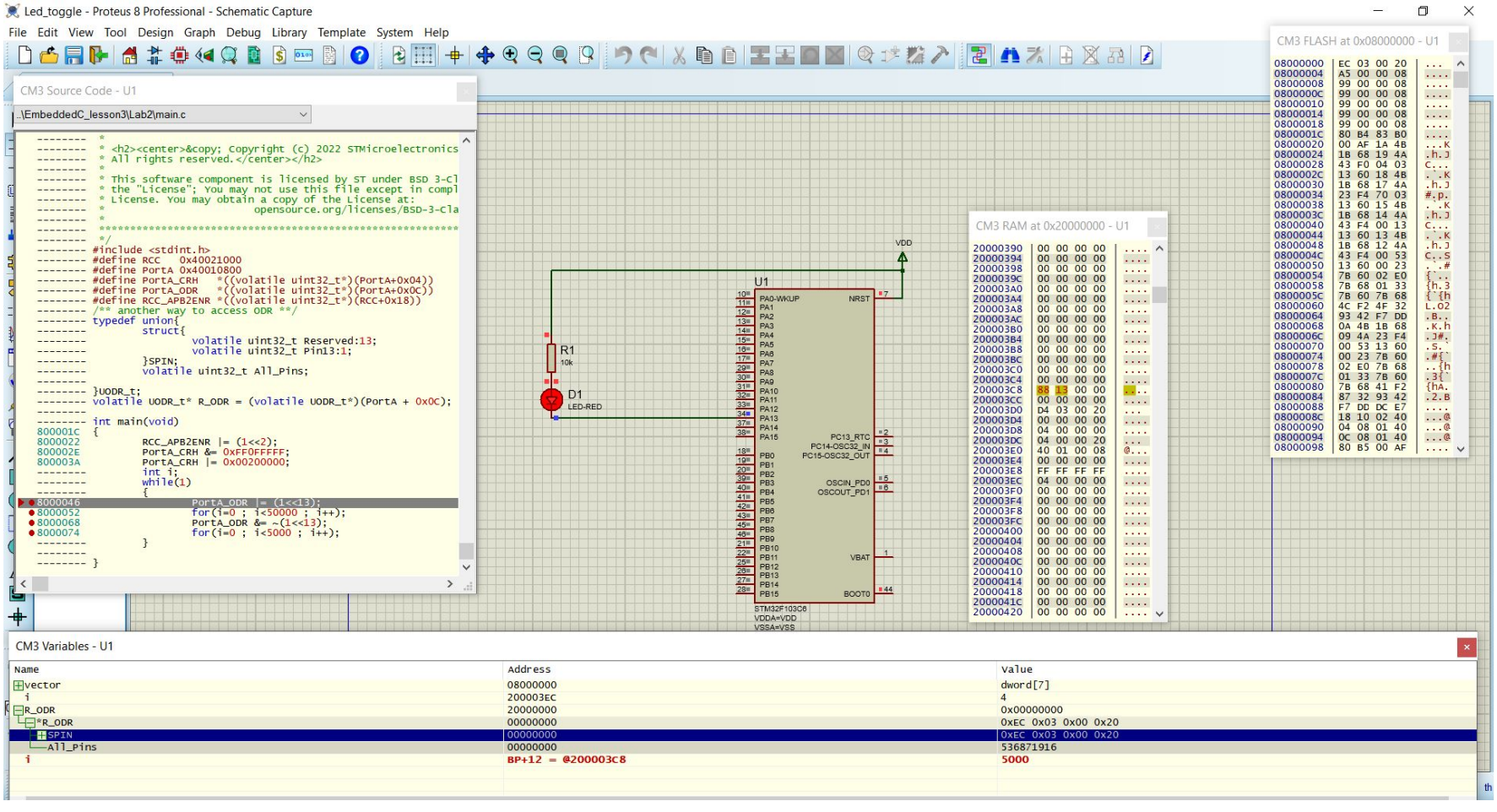
CM3 FLASH at 0x08000000 - U1

```
08000000  EC 03 00 20  ...
08000004  A5 00 00 08  ....
08000008  99 00 00 08  ....
0800000C  99 00 00 08  ....
08000010  99 00 00 08  ....
08000014  99 00 00 08  ....
08000018  99 00 00 08  ....
0800001C  80 B4 83 B0  ....
08000020  00 AF 1A 4B  ...K
08000024  1B 68 19 4A  .h.J
08000028  43 F0 04 03  C...
0800002C  13 60 18 4B  .`.K
08000030  1B 68 17 4A  .h.J
08000034  23 F4 70 03  #.p.
08000038  13 60 15 4B  .`.K
0800003C  1B 68 14 4A  .h.J
08000040  43 F4 00 13  C...
08000044  13 60 13 4B  .`.K
08000048  1B 68 12 4A  .h.J
0800004C  43 F4 00 53  C..S
08000050  13 60 00 23  .`.#
08000054  7B 60 02 E0  {`..
08000058  7B 68 01 33  {h.3
0800005C  7B 60 7B 68  {`{h
08000060  4C F2 4F 32  L.O2
08000064  93 42 F7 DD  .B..
08000068  0A 4B 1B 68  .K.h
0800006C  09 4A 23 F4  .J#.
08000070  00 53 13 60  .S.`
08000074  00 23 7B 60  .#{`
08000078  02 E0 7B 68  ..{h
0800007C  01 33 7B 60  .3{`
08000080  7B 68 41 F2  {hA.
08000084  87 32 93 42  .2.B
08000088  F7 DD DC E7  ....
0800008C  18 10 02 40  ...@
08000090  04 08 01 40  ...@
08000094  0C 08 01 40  ...@
08000098  80 B5 00 AF  ....
```

CM3 Variables - U1

Name	Address	Value
vector	08000000	dword[7]
i	200003EC	4
R_ODR	20000000	0x00000000
*R_ODR	00000000	0xEC 0x03 0x00 0x20
SPIN	00000000	0xEC 0x03 0x00 0x20
Reserved	00000000	1004
Pin13	00000000	0
All_Pins	00000000	536871916

Output


Led_toggle - Proteus 8 Professional - Schematic Capture
File Edit View Tool Design Graph Debug Library Template System Help
CM3 Source Code - U1
..\EmbeddedC_lesson3\Lab2\main.c
#include <stdint.h>
#define RCC 0x40021000
#define PortA 0x40010800
#define PortA_CRH *((volatile uint32_t*)(PortA+0x04))
#define PortA_ODR *((volatile uint32_t*)(PortA+0x0C))
#define RCC_APB2ENR *((volatile uint32_t*)(RCC+0x18))
typedef union{
struct{
volatile uint32_t Reserved:13;
volatile uint32_t Pin13:1;
}SPIN;
volatile uint32_t All_Pins;
}UODR_t;
volatile UODR_t* R_ODR = (volatile UODR_t*)(PortA + 0x0C);
int main(void)
{
RCC_APB2ENR |= (1<<2);
PortA_CRH &= 0xFF0FFFFF;
PortA_CRH |= 0x00200000;
int i;
while(1)
{
PortA_ODR |= (1<<13);
for(i=0 ; i<50000 ; i++);
PortA_ODR &= ~(1<<13);
for(i=0 ; i<5000 ; i++);
}
}
U1
PA0-WKUP
NRST
PC13_RTC
PC14-OSC32_IN
PC15-OSC32_OUT
OSCIN_PD0
OSCOUT_PD1
VBAT
BOOT0
STM32F103C6
VDDA=VDD
VSSA=VSS
R1
10k
D1
LED-RED
VDD
CM3 RAM at 0x20000000 - U1
CM3 FLASH at 0x08000000 - U1
CM3 Variables - U1
Name
Address
Value
vector
08000000
dword[7]
i
200003EC
4
R_ODR
20000000
0x00000000
*R_ODR
00000000
0xEC 0x03 0x00 0x20
SPIN
00000000
0xEC 0x03 0x00 0x20
All_Pins
00000000
536871916
i
BP+12 = @200003C8
5000