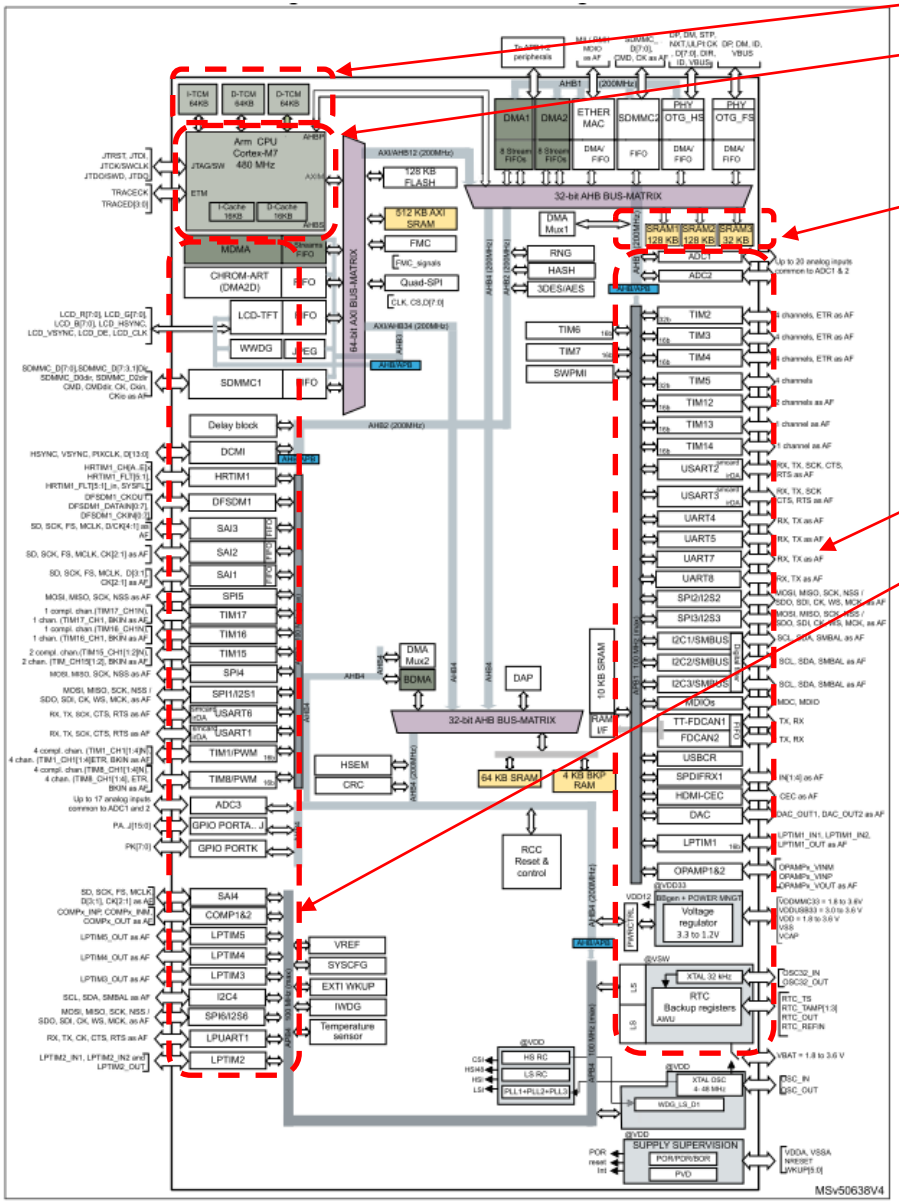
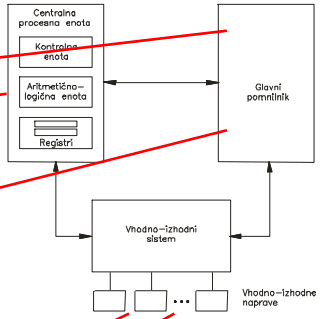
STM32H7

Vhodno / izhodne naprave

Prekinitve
+
SysTick Časovnik

STM32H750XB





Delo na STM32H7 razvojnem sistemu

Priključitev :

- **Mikro USB** priklop na **daljši stranici (nad LCD, srednji !!!)**

Poseben začetni projekt (github) in info za *STM32H7* (e-učilnica):

- ***dodajanje vsebine (Main.s):***

CubeIDEWorkspace - stm32h7-asm/Core/Src/Main.s - STM32CubeIDE

File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer

- CubeIDE_Workspace
  - stm32f4-asm-qemu
- Delo
  - ARM9Template
  - stm32f4-asm (in STM32AsmTemplate)
  - ARM9Template.zip
- Node_V4 (in node_v4)
- Sluzba
  - CAN_IEX_Module
  - ORLab-STM32H7
    - stm32h7-asm
      - Binaries
      - Includes
      - Core
        - Src
          - Main.s
        - Startup
          - startup_stm32h750xbhx.s
      - Debug
      - out
      - makefile
      - README.md
      - STM32H750X.svd
      - STM32H750XBHX_FLASH.ld
      - STM32H750XBHX_RAM.ld
    - README.md
  - RALab-STM32H7
    - stm32h7-asm_RA_LED
    - README.md
  - STM32_USB_Key_AdvDebug
  - STM32_USB_Key_FreeRTOS_AdvDebug
  - STM32CubeIDE_Adv_Debug
  - STM32F4_Discovery_VIN_Projects

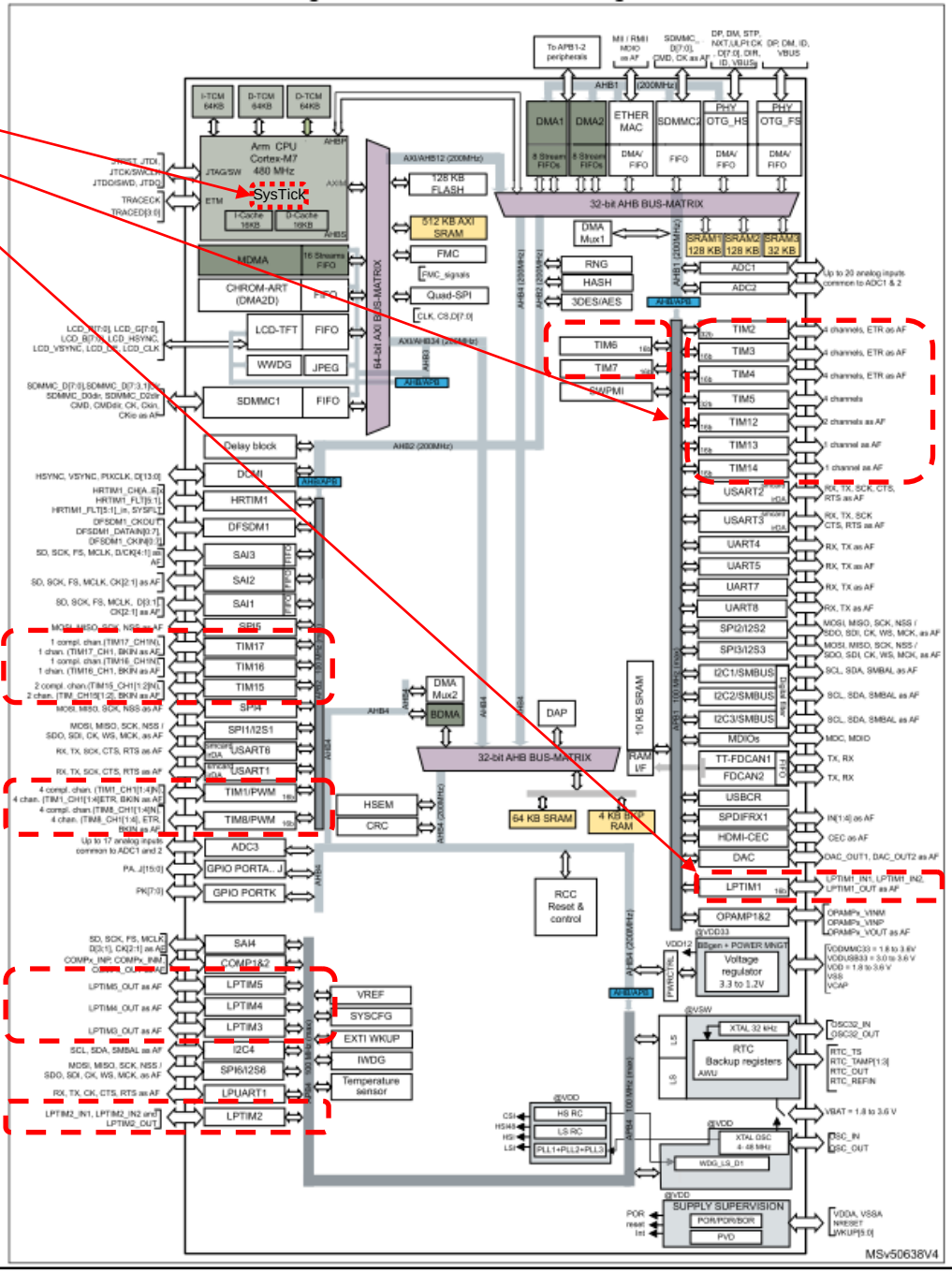
Main.s × startup_stm32h750xbhx.s

```
12
13 ////////////////////////////////////////////////////////////////////////////////
14 // Definitions
15 ////////////////////////////////////////////////////////////////////////////////
16 // Definitions section. Define all the registers and
17 // constants here for code readability.
18
19 // Constants
20
21
22 // Start of data section
23         .data
24
25         .align
26
27 STEV1:  .word   0x10    // 32-bitna spr.
28 STEV2:  .word   0x40    // 32-bitna spr.
29 VSOTA:  .word   0       // 32-bitna spr.
30
31
32 // Start of text section
33         .text
34
35         .type  main, %function
36         .global main
37
38         .align
39 main:
40         ldr r0, =STEV1   // Naslov od STEV1 -> r0
41         ldr r1, [r0]     // Vsebina iz naslova v r0 -> r1
42
43         ldr r0, =STEV2   // Naslov od STEV1 -> r0
44         ldr r2, [r0]     // Vsebina iz naslova v r0 -> r2
45
46         add r3,r1,r2     // r1 + r2 -> r3
47
48         ldr r0, =VSOTA   // Naslov od STEV1 -> r0
49         str r3,[r0]      // iz registra r3 -> na naslov v r0
50
51 __end:  b    __end
52
```

--------- Razvojni sistem STM32H750-DK ----------

- STM32H750B-DK Discovery kit with STM32H750XB MCU
- ORLab-STM32H7 - GitHub repozitorij
- User Manual Discovery kit stm32h750xb Uploaded 11/11/22, 10.15
- DataSheet_stm32h750xb Uploaded 11/11/22, 10.16
- Reference Manual rm0433-stm32h750xb Uploaded 11/11/22, 10.17
- Programming_Manual_pm0253-stm32h750xb Uploaded 11/11/22, 10.17
- Errata_es0396-stm32h750xb Uploaded 11/11/22, 10.19

Časovniki


SysTick
TIM2
TIM3
TIM4
TIM5
TIM12
TIM13
TIM14
TIM6
TIM7
TIM17
TIM16
TIM15
TIM1/PWM
TIM8/PWM
LPTIM1
LPTIM5
LPTIM4
LPTIM3
LPTIM2
MSv50638V4

Vira: Reference & Programming manuals

life.augmented

PM0253
Programming manual

STM32F7 Series and STM32H7 Series Cortex®-M7 processor programming manual

4 Cortex-M7 peripherals

4.1 About the Cortex-M7 peripherals

The address map of the *Private peripheral bus* (PPB)

4.5 SysTick timer (STK)

0xE000E100-0xE000E4EF	Nested Vectored Interrupt Controller	Table 40 on page 184
0xE000ED00-0xE000ED3F	System control block	Table 50 on page 192
0xE000ED78- 0xE000ED84	Processor features	Table 77 on page 217
0xE000ED90-0xE000EDB8	Memory Protection Unit	Table 84 on page 222
0xE000EF00-0xE000EF03	Nested Vectored Interrupt Controller	Table 40 on page 184
0xE000EF30-0xE000EF44	Floating-Point Unit	Table 94 on page 233
0xE000EF50-0xE000EF78	Cache maintenance operations	Table 100 on page 240
0xE000EF90-0xE000EFA8	Access control	Table 104 on page 245

Table 71. System timer registers summary

Address	Name	Type	Required privilege	Reset value	Description
0xE000E010	SYST_CSR	RW	Privileged	0x00000004	*SysTick control and status register*
0xE000E014	SYST_RVR	RW	Privileged	UNKNOWN	*SysTick reload value register*
0xE000E018	SYST_CVR	RW	Privileged	UNKNOWN	*SysTick current value register*
0xE000E01C	SYST_CALIB	RO	Privileged	0xC0000000	*SysTick calibration value register*

life.augmented

RM0433
Reference manual

STM32H742, STM32H743/753 and STM32H750 Value line advanced Arm®-based 32-bit MCUs

SysTick časovnik – stanje , nastavitve

Bazni naslov za registre SysTick je 0xE000E010

Figure 40. SysTick SYST_CSR bit assignments

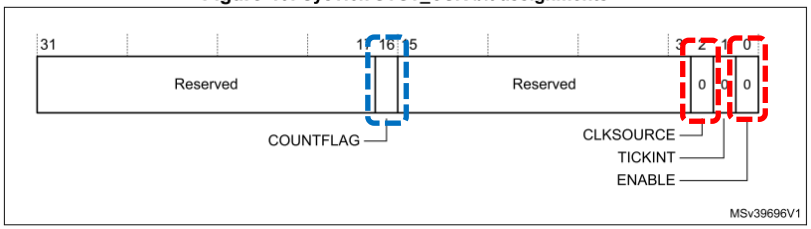


Figure 41. SYST_RVR bit assignments

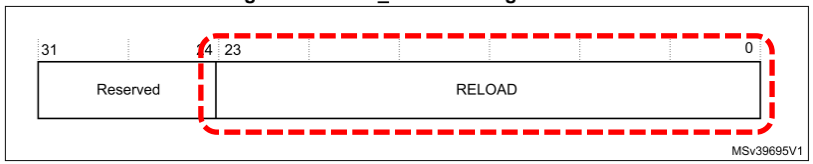
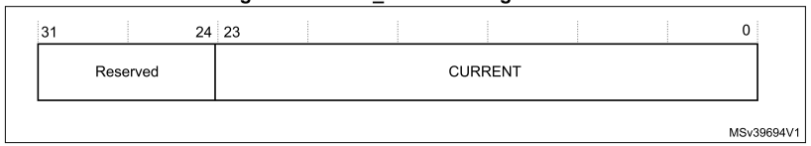


Figure 42. SYST_CVR bit assignments:



Osnovni registri za delovanje SysTick časovnika:

SYST_CSR : vklop časovnika

CLKSOURCE=1, ENABLE=1

COUNTERFLAG=1, ko prešteje do 0 (postavi na SYST_RVR in nadaljuje)

SYST_RVR : zač. vrednost štetja (šteje proti 0)

SYST_RVR = število period

SYST_CVR : trenutna vrednost števca

SYST_CVR = nekje med SYST_RVR in 0

SysTick časovnik *(Registri za nastavitve delovanja)*

Figure 40. SysTick SYST_CSR bit assignments

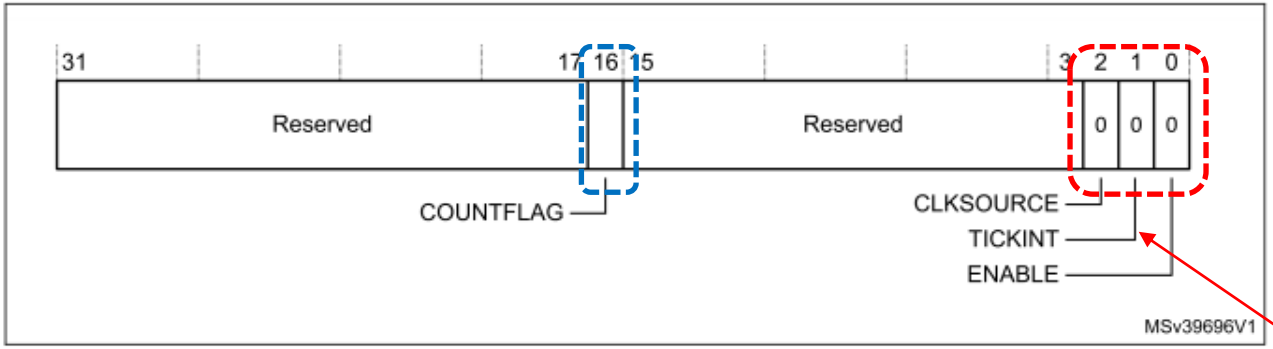


Table 72. SysTick SYST_CSR bit assignments

Bits	Name	Function
[31:17]	-	Reserved.
[16]	COUNTFLAG	Returns 1 if timer counted to 0 since last time this was read.
[15:3]	-	Reserved.
[2]	CLKSOURCE	Indicates the clock source: – 0: External clock. – 1: Processor clock.
[1]	TICKINT	Enables SysTick exception request: 0: Counting down to zero does not assert the SysTick exception request. 1: Counting down to zero asserts the SysTick exception request. Software can use COUNTFLAG to determine if SysTick has ever counted to zero.
[0]	ENABLE	Enables the counter: 0: Counter disabled. 1: Counter enabled.

Vklop prekinitve

SysTick časovnik *(Registri za nastavitve delovanja)*

4.4.2 SysTick reload value register

The SYST_RVR register specifies the start value to load into the SYST_CVR register. See the register summary in *Table 71 on page 213* for its attributes. The bit assignments are:

Figure 41. SYST_RVR bit assignments

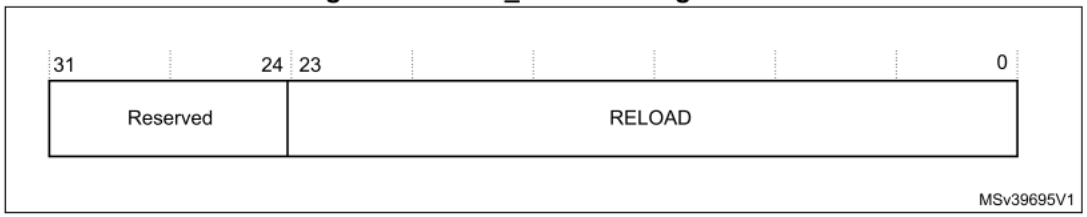


Table 73. SYST_RVR bit assignments

Bits	Name	Function
[31:24]	-	Reserved.
[23:0]	RELOAD	Value to load into the SYST_CVR register when the counter is enabled and when it reaches 0, see *Calculating the RELOAD value*.

Calculating the RELOAD value

The RELOAD value can be any value in the range 0x00000001-0x00FFFFFF. A start value of 0 is possible, but has no effect because the SysTick exception request and COUNTFLAG are activated when counting from 1 to 0.

The RELOAD value is calculated according to its use. For example, to generate a multi-shot timer with a period of N processor clock cycles, use a RELOAD value of N-1. If the SysTick interrupt is required every 100 clock pulses, set RELOAD to 99.

SysTick časovnik *(Registri za nastavitve delovanja)*

4.4.3 SysTick current value register

The SYST_CVR register contains the current value of the SysTick counter. See the register summary in *Table 71 on page 213* for its attributes. The bit assignments are

Figure 42. SYST_CVR bit assignments:

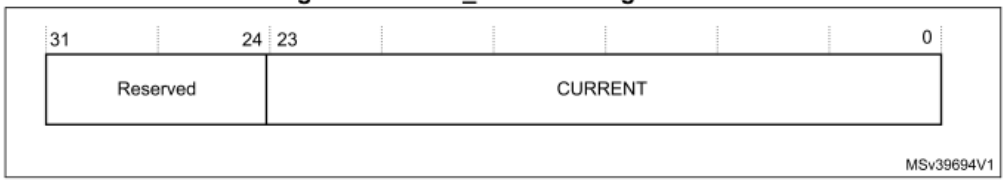


Table 74. SYST_CVR bit assignments

Bits	Name	Function
[31:24]	-	Reserved.
[23:0]	CURRENT	Reads return the current value of the SysTick counter. A write of any value clears the field to 0, and also clears the SYST_CSR COUNTFLAG bit to 0.

Prekinitveni vektorji

Table 143. NVIC[1]

Signal	Priority	NVIC position	Acronym	Description	Address offset
-	-	-	-	Reserved	0x0000 0000
-	-3	-	Reset	Reset	0x0000 0004
-	-2	-	NMI	Non maskable interrupt. The RCC Clock Security System (CSS) is linked to the NMI vector.	0x0000 0008
-	-1	-	HardFault	All classes of fault	0x0000 000C
-	0	-	MemManage	Memory management	0x0000 0010
-	1	-	BusFault	Prefetch fault, memory access fault	0x0000 0014
-	2	-	UsageFault	Undefined instruction or illegal state	0x0000 0018
-	-	-	-	Reserved	0x0000 001C-0x0000 002B
-	3	-	SVCall	System service call via SWI instruction	0x0000 002C
-	4	-	DebugMonitor	Debug monitor	0x0000 0030
-	-	-	-	Reserved	0x0000 0034
-	5	-	PendSV	Pendable request for system service	0x0000 0038
-	6	-	SysTick	System tick timer	0x0000 003C
wwdg1_it	7	0	WWDG1	Window Watchdog interrupt	0x0000 0040
exti_pwr_pvd_wkup	8	1	PVD_PVM	PVD through EXTI line detection interrupt	0x0000 0044
exti_tamp_rtc_wkup	9	2	RTC_TAMP_STAMP_CSS_LSE	RTC tamper, timestamp	0x0000 0048
lsecss_rcc_it				CSS LSE	
exti_wkup_rtc_wkup	10	3	RTC_WKUP	RTC Wakeup interrupt through the EXTI line	0x0000 004C
flash_it	11	4	FLASH	Flash memory global interrupt	0x0000 0050
rcc_it	12	5	RCC	RCC global interrupt	0x0000 0054
exti_exti0_wkup	13	6	EXTI0	EXTI Line 0 interrupt	0x0000 0058
exti_exti1_wkup	14	7	EXTI1	EXTI Line 1 interrupt	0x0000 005C
exti_exti2_wkup	15	8	EXTI2	EXTI Line 2 interrupt	0x0000 0060

Table 143. NVIC[1] (continued)

Signal	Priority	NVIC position	Acronym	Description	Address offset
exti_exti3_wkup	16	9	EXTI3	EXTI Line 3interrupt	0x0000 0064
exti_exti4_wkup	17	10	EXTI4	EXTI Line 4interrupt	0x0000 0068
dma1_it0	18	11	DMA_STR0	DMA1 Stream0 global interrupt	0x0000 006C
dma1_it1	19	12	DMA_STR1	DMA1 Stream1 global interrupt	0x0000 0070
dma1_it2	20	13	DMA_STR2	DMA1 Stream2 global interrupt	0x0000 0074
dma1_it3	21	14	DMA_STR3	DMA1 Stream3 global interrupt	0x0000 0078
dma1_it4	22	15	DMA_STR4	DMA1 Stream4 global interrupt	0x0000 007C
dma1_it5	23	16	DMA_STR5	DMA1 Stream5 global interrupt	0x0000 0080
dma1_it6	24	17	DMA_STR6	DMA1 Stream6 global interrupt	0x0000 0084
adc1_it	25	18	ADC1_2	ADC1 and ADC2 global interrupt	0x0000 0088
adc2_it					
ttfdcan_intr0_it	26	19	FDCAN1_IT0	FDCAN1 Interrupt 0	0x0000 008C
fdcan_intr0_it	27	20	FDCAN2_IT0	FDCAN2 Interrupt 0	0x0000 0090
ttfdcan_intr1_it	28	21	FDCAN1_IT1	FDCAN1 Interrupt 1	0x0000 0094
fdcan_intr1_it	29	22	FDCAN2_IT1	FDCAN2 Interrupt 1	0x0000 0098
exti_exti5_wkup	30	23	EXTI9_5	EXTI Line[9:5] interrupts	0x 0000 009C
exti_exti6_wkup					
exti_exti7_wkup					
exti_exti8_wkup					
exti_exti9_wkup					
tim1_brk_it	31	24	TIM1_BRK	TIM1 break interrupt	0x0000 00A0
tim1_upd_it	32	25	TIM1_UP	TIM1 update interrupt	0x0000 00A4
tim1_trg_it	33	26	TIM1_TRG_COM	TIM1 trigger and commutation interrupts	0x0000 00A8
tim1_cc_it	34	27	TIM_CC	TIM1 capture / compare interrupt	0x0000 00AC
tim2_it	35	28	TIM2	TIM2 global interrupt	0x0000 00B0
tim3_it	36	29	TIM3	TIM3 global interrupt	0x0000 00B4
tim4_it	37	30	TIM4	TIM4 global interrupt	0x0000 00B8

Prekinitveni vektorji

```
// Start of text section
.section .text
////////////////////////////////////////////////////////////////////////////////
// Vectors
////////////////////////////////////////////////////////////////////////////////
// Vector table start
// Add all other processor specific exceptions/interrupts in order here
.long    __StackTop                  // Top of the stack. from linker script
.long    _start +1                   // reset location, +1 for thumb mode
  .word  NMI_Handler
  .word  HardFault_Handler
  .word  MemManage_Handler
  .word  BusFault_Handler
  .word  UsageFault_Handler
  .word  0
  .word  0
  .word  0
  .word  0
  .word  SVC_Handler
  .word  DebugMon_Handler
  .word  0
  .word  PendSV_Handler
  .word  SysTick_Handler

  /* External Interrupts */
  .word     WWDG_IRQHandler                  /* Window WatchDog              */
  .word     PVD_IRQHandler                   /* PVD through EXTI Line detection */
  .word     TAMP_STAMP_IRQHandler            /* Tamper and TimeStamps through the EXTI line */
  .word     RTC_WKUP_IRQHandler              /* RTC Wakeup through the EXTI line */
```

Prekinitveni vektorji

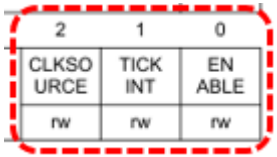
```
/*******************************************************************************
*
* Provide weak aliases for each Exception handler to the Default_Handler.
* As they are weak aliases, any function with the same name will override
* this definition.
*
*******************************************************************************/
   .weak      NMI_Handler
   .thumb_set NMI_Handler,Default_Handler

   .weak      HardFault_Handler
   .thumb_set HardFault_Handler,Default_Handler

   .weak      MemManage_Handler
   .thumb_set MemManage_Handler,Default_Handler

   .weak      BusFault_Handler
   .thumb_set BusFault_Handler,Default_Handler

   .weak      UsageFault_Handler
   .thumb_set UsageFault_Handler,Default_Handler

   .weak      SVC_Handler
   .thumb_set SVC_Handler,Default_Handler

   .weak      DebugMon_Handler
   .thumb_set DebugMon_Handler,Default_Handler

   .weak      PendSV_Handler
   .thumb_set PendSV_Handler,Default_Handler

   .weak      SysTick_Handler
   .thumb_set SysTick_Handler,Default_Handler
```

```
/**
 * @brief  This is the code that gets called when the
processor receives an
 *         unexpected interrupt.  This simply enters
an infinite loop, preserving
 *         the system state for examination by a
debugger.
 * @param  None
 * @retval None
*/
    .section  .text.Default_Handler,"ax",%progbits
Default_Handler:
Infinite_Loop:
  b  Infinite_Loop
  .size  Default_Handler, .-Default_Handler
```

SysTick Časovnik – krmiljenje

Potrebni koraki za krmiljenje časovnika SysTick:

1. **STK_LOAD (Reload Value Register): Value** SYSTICK_RELOAD_1MS
2. **STK_VAL (Current Value Register): 0, reset to zero**
3. **STK_CTRL (Control/Status Register): 0b111 :**
   Proc. Clock, Tick Interrupt, enable -> Start SysTick



4. **Delovanje:**
   Proženje SysTick_Handler vsako 1 ms

SysTick_Handler :

```
.global SysTick_Handler

.section  .text.SysTick_Handler,"ax",%progbits
.type  SysTick_Handler, %function


SysTick_Handler:
    push {r3, r4, r5, lr}
    …
RET: pop {r3, r4, r5, pc}
```

Števec v r8 šteje 500 ms
Zastavica v r7 pove stanje LED diod
Vse se dogaja v PSP !

SysTick Časovnik – krmiljenje

SysTick_Handler :

```
.global SysTick_Handler
.section  .text.SysTick_Handler,"ax",%progbits
.type  SysTick_Handler, %function

SysTick_Handler:
     push {r3, r4, r5, lr}

     ldr r3,=MSECCNT  // Load MSecs Counter value
     ldr r4,[r3]
     add r4,r4,#1 // Increment (+1)
     str r4,[r3]
     ldr r3,=MSECMAX  // Load MAX value
     ldr r5,[r3]
     cmp r4,r5        // End of interval ?
     blo RET

// End of interval - reset counter and toggle LED
     movr4,#0    // reset MSecs Counter
     ldr r3,=MSECCNT
     str r4,[r3]

     ldr r3,=LEDSTAT // Check LED STATE
     ldr r4,[r3]
     cmp r4,#0    // is it OFF ?
     beq LON
     …
```

```
     …

// Set state and LED Off
     mov r4,#0
     str r4,[r3]

// Set GPIOx Pins to 1 (through BSSR register)
     ldr    r3, =GPIOI_BASE      // Load GPIOI BASE addr.
     mov    r4, #LEDs_OFF
     str    r4, [r3,#GPIOx_BSSR] // Write to BSRR register

     b RET


LON:// Set state and LED On
     mov r4,#1
     str r4,[r3]

// Set GPIOx Pins to 0 (through BSSR register)
     ldr    r3, =GPIOI_BASE       // Load GPIOI BASE addr.
     mov    r4, #LEDs_ON
     str    r4, [r3,#GPIOx_BSSR] // Write to BSRR register


RET:   pop {r3, r4, r5, pc}
```

SysTick Časovnik – krmiljenje

Ostala koda :

```
// Start of data section
  .data
LEDSTAT: .word   0   // LED state
MSECCNT: .word   0   //MSecs counter for SysTick_Handler
MSECMAX: .word   500 //MSecs interval for SysTick_Handler
…

// Start of text section
  .text

  .type  main, %function
  .global main

      .align
main:

    bl INIT_IO     // Priprava za kontrolo LED diode
    bl INIT_TC_PSP // Priprava SysTick časovnika s
prekinitvami

endloop: b endloop
```

```
      …

INIT_TC_PSP:

push {r0, r1, lr}
      ldr r1, =SCS_BASE

      ldr r0, =SYSTICK_RELOAD_1MS
      str r0, [r1, #SCS_SYST_RVR]

      ldr r0, =0
      str r0, [r1, #SCS_SYST_CVR]

      ldr r0, =7      // TickINT Bit also set to 1
      str r0, [r1, #SCS_SYST_CSR]

pop {r0, r1, pc}
```

CubeIDE – SFR okno