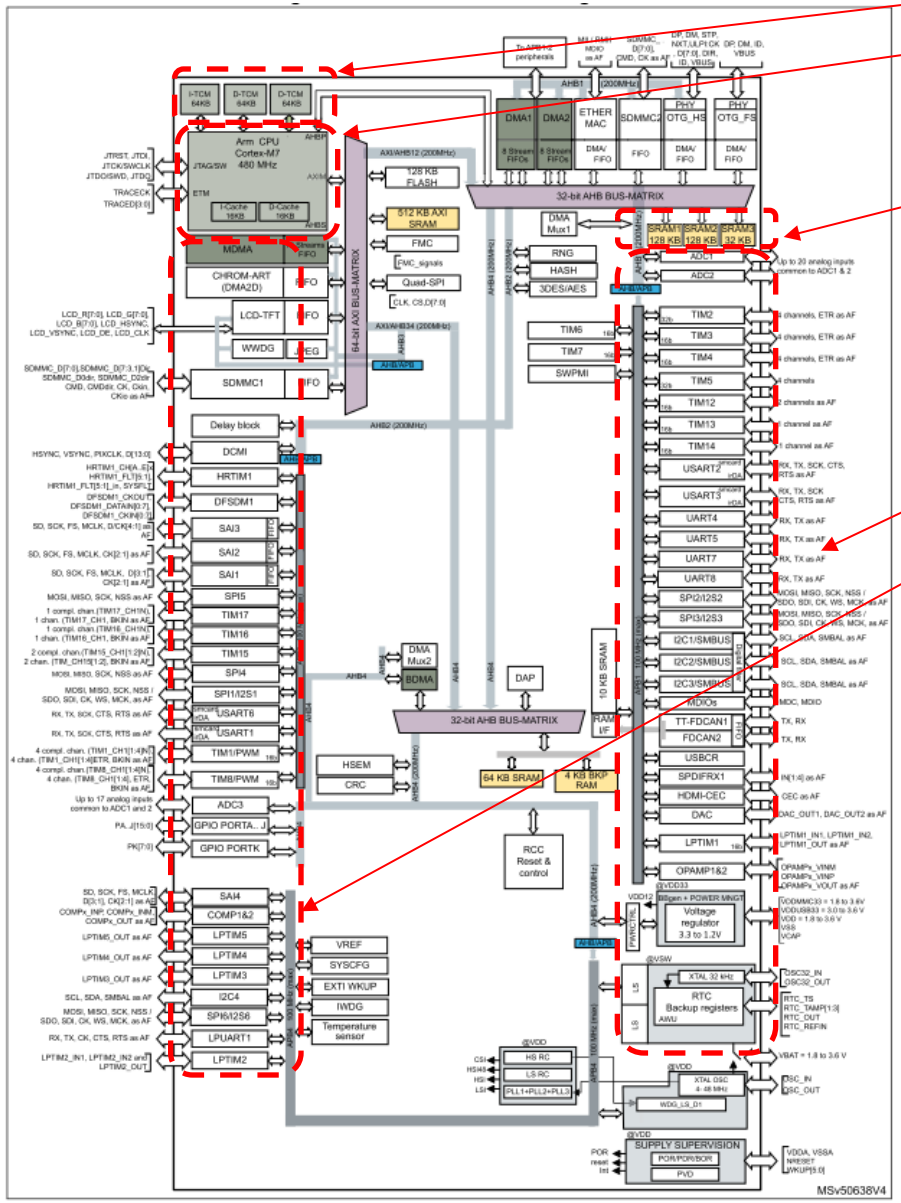
# *STM32H750B-DK Discovery Kit*

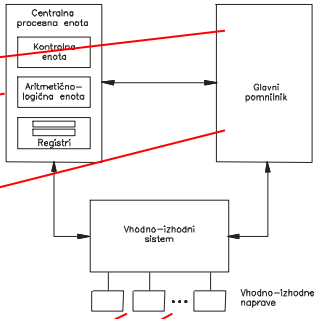
*Vhodno / izhodne naprave*

*GPIO Krmilnik II*

*Utripanje LED diode*

# STM32H750XB



# Delo na STM32H7 razvojnem sistemu

Priključitev :

- **Mikro USB** priklop na **daljši stranici (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

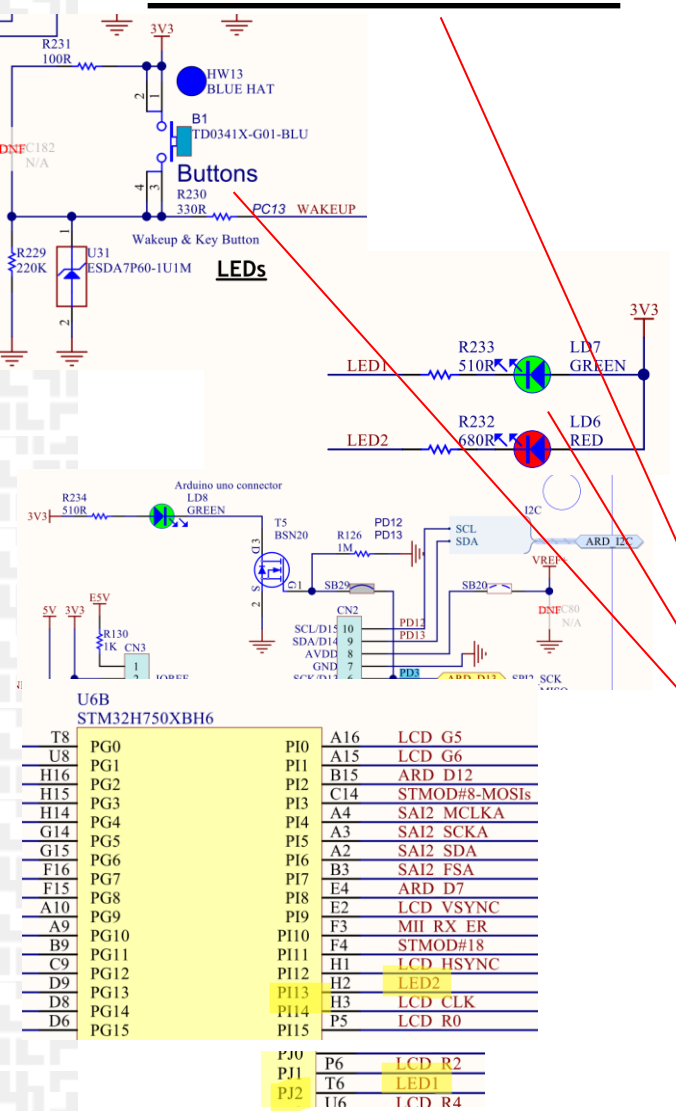
Main.s × startup_stm32h750xbhx.s

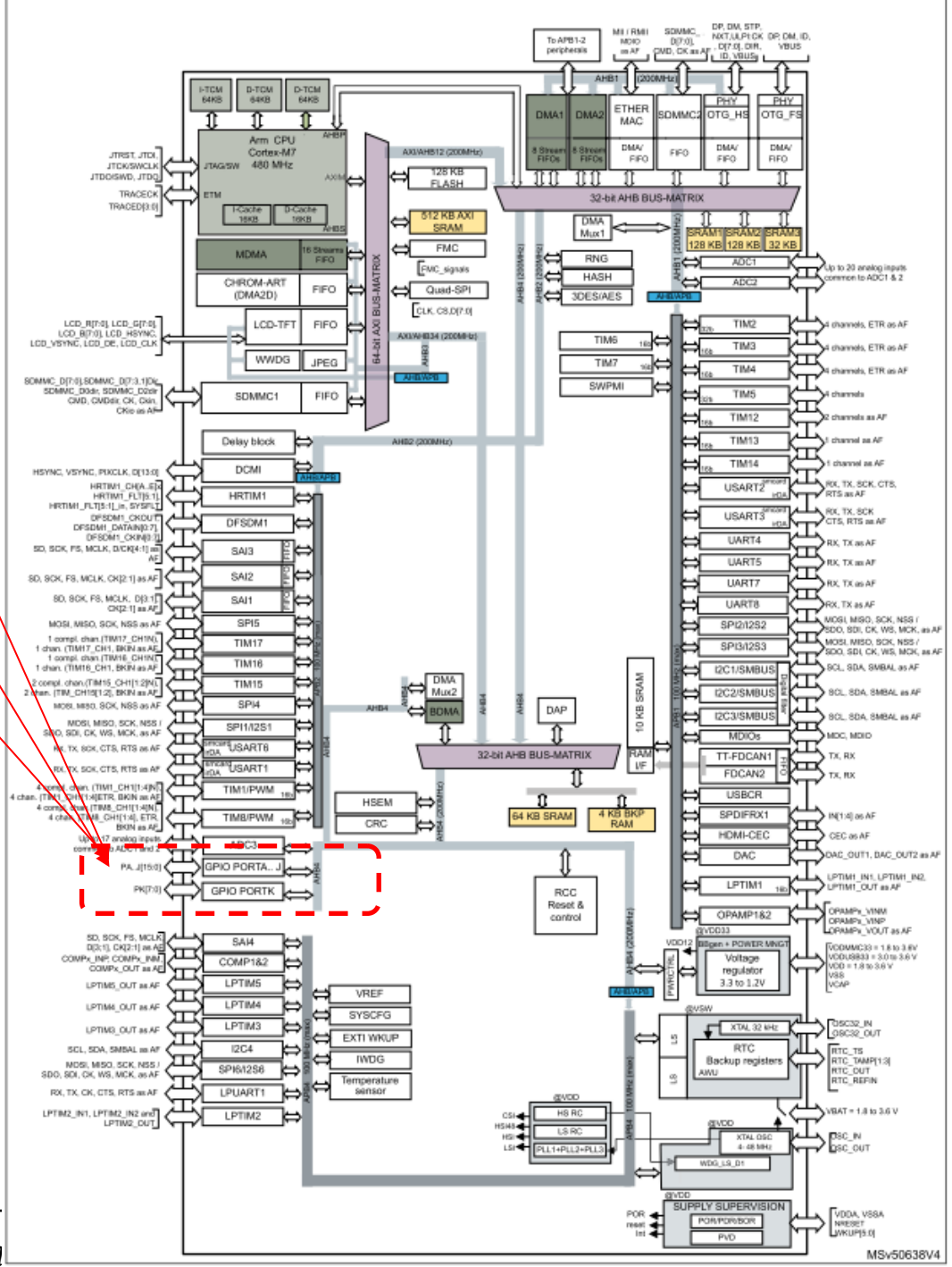
```
12
13 ////////////////////////////////////////////////////////////////////////////
14 // Definitions
15 ////////////////////////////////////////////////////////////////////////////
16 // Definitions section. Define all the registers and
17 // constants here for code readability.
18
19 // Constants
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

# GPIO Krmilnik




*LED: rdeča PI13, zelena PJ2*
*zelena PD3*

# *Vir: RM0433 Reference manual*

Strani: 131, 455, 527-545

life.augmented

**RM0433**
**Reference manual**

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



**Table 8. Register boundary addresses**(1)

| Boundary address | Peripheral | Bus | Register map |
|---|---|---|---|
| 0x58027000 - 0x580273FF | RAMECC3 | | Section 3.4: RAMECC registers |
| 0x58026400 - 0x580267FF | HSEM | | Section 10.4: HSEM registers |
| 0x58026000 - 0x580263FF | ADC3 | | Section 25.7: ADC common registers |
| 0x58025800 - 0x58025BFF | DMAMUX2 | | Section 17.6: DMAMUX registers |
| 0x58025400 - 0x580257FF | BDMA | | Section 16.6: BDMA registers |
| 0x58024C00 - 0x58024FFF | CRC | | Section 21.4: CRC registers |
| 0x58024800 - 0x58024BFF | PWR | | Section 6.8: PWR register description |
| 0x58024400 - 0x580247FF | RCC | | Section 8.7: RCC register description |
| 0x58022800 - 0x58022BFF | GPIOK | | Section 11.4: GPIO registers |
| 0x58022400 - 0x580227FF | GPIOJ | AHB4 (D3) | Section 11.4: GPIO registers |
| 0x58022000 - 0x580223FF | GPIOI | | Section 11.4: GPIO registers |
| 0x58021C00 - 0x58021FFF | GPIOH | | Section 11.4: GPIO registers |
| 0x58021800 - 0x58021BFF | GPIOG | | Section 11.4: GPIO registers |
| 0x58021400 - 0x580217FF | GPIOF | | Section 11.4: GPIO registers |
| 0x58021000 - 0x580213FF | GPIOE | | Section 11.4: GPIO registers |
| 0x58020C00 - 0x58020FFF | GPIOD | | Section 11.4: GPIO registers |
| 0x58020800 - 0x58020BFF | GPIOC | | Section 11.4: GPIO registers |
| 0x58020400 - 0x580207FF | GPIOB | | Section 11.4: GPIO registers |
| 0x58020000 - 0x580203FF | GPIOA | | Section 11.4: GPIO registers |

RM0433 Rev 7 131/3319

**11 General-purpose I/Os (GPIO)**

**11.1 Introduction**

Each general-purpose I/O port has four 32-bit configuration registers (GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR and GPIOx_PUPDR), two 32-bit data registers (GPIOx_IDR and GPIOx_ODR) and a 32-bit set/reset register (GPIOx_BSRR). In addition all GPIOs have a 32-bit locking register (GPIOx_LCKR) and two 32-bit alternate function selection registers (GPIOx_AFRH and GPIOx_AFRL).



**8.7.43 RCC AHB4 Clock Register (RCC_AHB4ENR)**

This register can be accessed via two different offset address.

**Table 68. RCC_AHB4ENR address offset and reset value**

| Register Name | Address Offset | Reset Value |
|---|---|---|
| RCC_AHB4ENR | 0x0E0 | 0x0000 0000 |
| RCC_C1_AHB4ENR | 0x140 | |


RM0433 Rev 7 455/3319

rm0433-stm32h750_GPIO_Ch11.pdf

# Vir: RM0433 Reference manual



### 8.7.43 RCC AHB4 Clock Register (RCC_AHB4ENR)

This register can be accessed via two different offset address.

Bit 9 **GPIOJEN**: GPIOJ peripheral clock enable
Set and reset by software.
0: GPIOJ peripheral clock disabled (default after reset)
1: GPIOJ peripheral clock enabled

Bit 8 **GPIOIEN**: GPIOI peripheral clock enable
Set and reset by software.
0: GPIOI peripheral clock disabled (default after reset)
1: GPIOI peripheral clock enabled

**Table 68. RCC_AHB4ENR address offset and reset value**

| Register Name | Address Offset | Reset Value |
|---|---|---|
| RCC_AHB4ENR | 0x0E0 | 0x0000 0000 |
| RCC_C1_AHB4ENR | 0x140 | |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Res. | Res. | Res. | BKPRAMEN | Res. | Res. | HSEMEN | ADC3EN | Res. | Res. | BDMAEN | Res. | CRCEN | Res. | Res. | Res. |
| | | | rw | | | rw | rw | | | rw | | rw | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Res. | Res. | Res. | Res. | Res. | GPIOKEN | GPIOJEN | GPIOIEN | GPIOHEN | GPIOGEN | GPIOFEN | GPIOEEN | GPIODEN | GPIOCEN | GPIOBEN | GPIOAEN |
| | | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

# Vir: RM0433 Reference manual

life.augmented

**RM0090**
**Reference manual**

STM32F405/415, STM32F407/417, STM32F427/437 and STM32F429/439 advanced Arm®-based 32-bit MCUs

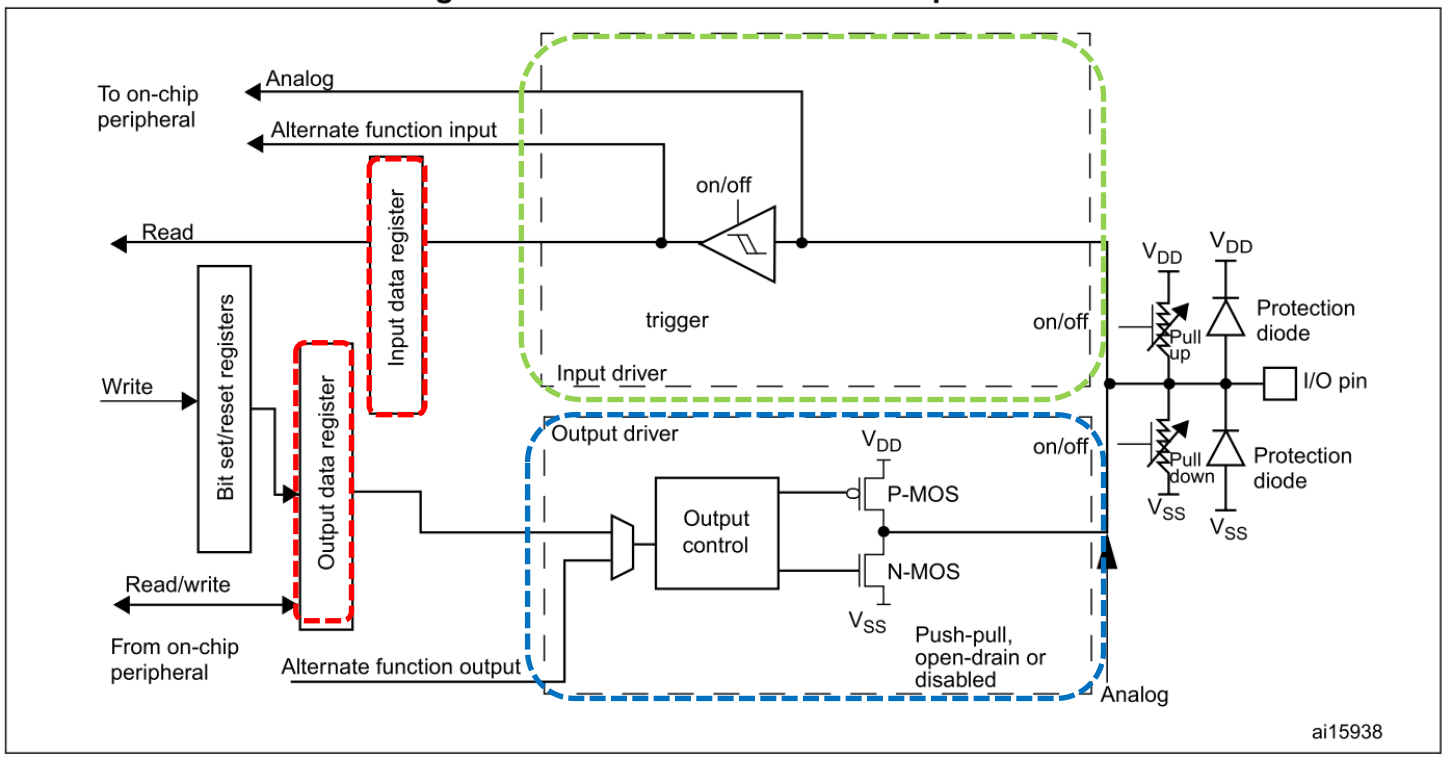
**Table 8. Register boundary addresses(1)**

| Boundary address | Peripheral | Bus | Register map |
|---|---|---|---|
| 0x58027000 - 0x580273FF | RAMECC3 | | *Section 3.4: RAMECC registers* |
| 0x58026400 - 0x580267FF | HSEM | AHB4 (D3) | *Section 10.4: HSEM registers* |
| 0x58026000 - 0x580263FF | ADC3 | | *Section 25.7: ADC common registers* |
| 0x58025800 - 0x58025BFF | DMAMUX2 | | *Section 17.6: DMAMUX registers* |
| 0x58025400 - 0x580257FF | BDMA | | *Section 16.6: BDMA registers* |
| 0x58024C00 - 0x58024FFF | CRC | | *Section 21.4: CRC registers* |
| 0x58024800 - 0x58024BFF | PWR | | *Section 6.8: PWR register description* |
| 0x58024400 - 0x580247FF | RCC | | *Section 8.7: RCC register description* |
| 0x58022800 - 0x58022BFF | GPIOK | | *Section 11.4: GPIO registers* |
| 0x58022400 - 0x580227FF | GPIOJ | | *Section 11.4: GPIO registers* |
| 0x58022000 - 0x580223FF | GPIOI | | *Section 11.4: GPIO registers* |
| 0x58021C00 - 0x58021FFF | GPIOH | | *Section 11.4: GPIO registers* |
| 0x58021800 - 0x58021BFF | GPIOG | | *Section 11.4: GPIO registers* |
| 0x58021400 - 0x580217FF | GPIOF | | *Section 11.4: GPIO registers* |
| 0x58021000 - 0x580213FF | GPIOE | | *Section 11.4: GPIO registers* |
| 0x58020C00 - 0x58020FFF | GPIOD | | *Section 11.4: GPIO registers* |
| 0x58020800 - 0x58020BFF | GPIOC | | *Section 11.4: GPIO registers* |
| 0x58020400 - 0x580207FF | GPIOB | | *Section 11.4: GPIO registers* |
| 0x58020000 - 0x580203FF | GPIOA | | *Section 11.4: GPIO registers* |

RM0433 Rev 7 131/3319

# GPIO krmilnik – vhod/izhod

Figure 70. Basic structure of an I/O port bit




## Osnovni registri za GPIO priključke :

**RCC_AHBxENR : vklop urinega signala (enote) : Port I: RCC_AHB4ENR($b_8$=1 .. Port I Enable)**

**MODER (Mode Register): 00: Input (reset) / 01: General purpose output mode**

**OTYPER (Output TYPE Register): 0: Output push-pull (reset) / 1: Output open-drain**

**OSPEEDR (Output SPEED Register): 00 – Low speed (reset) .. 11: Very high speed**

**PUPDR (Pull Up/Down Register): 00 – No pull (reset) .. 01: Pull-Up .. 10: Pull-Down**

**IDR (Input Data Register): stanje vhoda 1 / 0**

**ODR (Output Data Register): stanje izhoda 1 / 0**

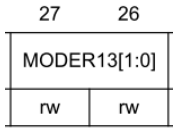
# GPIO krmilnik – izhod *(Registri za nastavitve delovanja)*

## RCC_AHB4ENR(Peripheral Clock Register):

Bit 8 **GPIOIEN:** GPIOI peripheral clock enable
Set and reset by software.
0: GPIOI peripheral clock disabled (default after reset)
1: GPIOI peripheral clock enabled


## MODER (Mode Register):

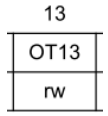
Bits 31:0 **MODER[15:0][1:0]:** Port x configuration I/O pin y (y = 15 to 0)
These bits are written by software to configure the I/O mode.
00: Input mode
01: General purpose output mode
10: Alternate function mode
11: Analog mode (reset state)




## OTYPER (Output TYPE Register):

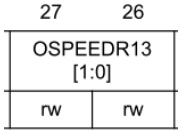
Bits 31:16 Reserved, must be kept at reset value.

Bits 15:0 **OT[15:0]:** Port x configuration I/O pin y (y = 15 to 0)
These bits are written by software to configure the I/O output type.
0: Output push-pull (reset state)
1: Output open-drain



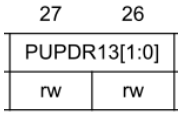

## OSPEEDR (Output SPEED Register):

Bits 31:0 **OSPEEDR[15:0][1:0]**: Port x configuration I/O pin y (y = 15 to 0)
These bits are written by software to configure the I/O output speed.
00: Low speed
01: Medium speed
10: High speed
11: Very high speed




## PUPDR (Pull Up/Down Register):

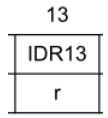
Bits 31:0 **PUPDR[15:0][1:0]:** Port x configuration I/O pin y (y = 15 to 0)
These bits are written by software to configure the I/O pull-up or pull-down
00: No pull-up, pull-down
01: Pull-up
10: Pull-down
11: Reserved




## IDR (Input Data Register): stanje vhoda 1 / 0

Bits 31:16 Reserved, must be kept at reset value.

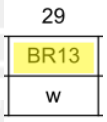
Bits 15:0 **IDR[15:0]:** Port x input data I/O pin y (y = 15 to 0)
These bits are read-only. They contain the input value of the corresponding I/O port.




## ODR (Output Data Register): stanje izhoda 1 / 0

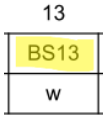
Bits 31:16 Reserved, must be kept at reset value.

Bits 15:0 **ODR[15:0]:** Port output data I/O pin y (y = 15 to 0)
These bits can be read and written by software.
*Note: For atomic bit set/reset, the ODR bits can be individually set and/or reset by writing to the GPIOx_BSRR register (x = A..F).*




Bits 31:16 **BR[15:0]:** Port x reset I/O pin y (y = 15 to 0)
These bits are write-only. A read to these bits returns the value 0x0000.
0: No action on the corresponding ODRx bit
1: Resets the corresponding ODRx bit
*Note: If both BSx and BRx are set, BSx has priority.*

Bits 15:0 **BS[15:0]:** Port x set I/O pin y (y = 15 to 0)
These bits are write-only. A read to these bits returns the value 0x0000.
0: No action on the corresponding ODRx bit
1: Sets the corresponding ODRx bit



# GPIO krmilnik – izhod *(Registri za določanje stanja izhodov)*

**2 možnosti za določanje stanja izhodov:**

1. Spreminjanje bitov v registru ODR
   Read-Modify-Write operacija na registru ODR
   ODR (Output Data Register): stanje izhoda 1 / 0

Bits 31:16 Reserved, must be kept at reset value.

Bits 15:0 **ODR[15:0]:** Port output data I/O pin y (y = 15 to 0)
These bits can be read and written by software.
*Note: For atomic bit set/reset, the ODR bits can be individually set and/or reset by writing to the GPIOx_BSRR register (x = A..F).*

| 13 |
|---|
| ODR13 |
| rw |

2. Spreminjanje bitov v registru ODR s pomočjo vpisa v register BSRR
   Write operacija v register BSRR -> sprememba bitov v registru ODR
   + atomična operacija
   BSRR (Port Set/Reset Register): stanje izhoda 1 / 0

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BR15 | BR14 | BR13 | BR12 | BR11 | BR10 | BR9 | BR8 | BR7 | BR6 | BR5 | BR4 | BR3 | BR2 | BR1 | BR0 |
| w | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| BS15 | BS14 | BS13 | BS12 | BS11 | BS10 | BS9 | BS8 | BS7 | BS6 | BS5 | BS4 | BS3 | BS2 | BS1 | BS0 |
| w | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w |

| 29 |
|---|
| BR13 |
| w |

Bits 31:16 **BR[15:0]:** Port x reset I/O pin y (y = 15 to 0)
These bits are write-only. A read to these bits returns the value 0x0000.
0: No action on the corresponding ODRx bit
1: Resets the corresponding ODRx bit
*Note: If both BSx and BRx are set, BSx has priority.*

Bits 15:0 **BS[15:0]:** Port x set I/O pin y (y = 15 to 0)
These bits are write-only. A read to these bits returns the value 0x0000.
0: No action on the corresponding ODRx bit
1: Sets the corresponding ODRx bit

| 13 |
|---|
| BS13 |
| w |

# GPIO krmilnik – krmiljenje izhodov

**Potrebni koraki za krmiljenje izhoda:**

1. RCC_AHB4ENR(Peripheral Clock Register): $b_8$=1 .. Port I Enable
2. **MODER (Mode Register): 01: General purpose output mode**
3. Default vrednosti že ustrezne v registrih :
   - **OTYPER (Output TYPE Register): 0: Output push-pull** (reset state)
   - **OSPEEDR (Output SPEED Register): 00 – Low speed** (reset state)
   - **PUPDR (Pull Up/Down Register): 00 – No pull** (reset state)
4. določi stanje izhoda s pisanjem v ODR ali BSRR (nastavljamo na 1/0)

**Naslovi registrov:**

```
// RCC   base address is 0x58024400
//   AHB4ENR register offset is 0xE0
.equ     RCC_AHB4ENR,   0x580244E0 // RCC AHB4 peripheral clock reg

.equ     GPIOI_BASE,   0x58022000  // GPIOI base address)
.equ     GPIOx_MODER,    0x00      // GPIOx port mode register
.equ     GPIOx_ODR,      0x14      // GPIOx output data register
.equ     GPIOx_BSRR,     0x18      // GPIOx port set/reset register

// Values for BSSR register - pin 13: LED is on, when GPIO is off
.equ     LEDs_OFF,       0x00002000   // Setting pin to 1 -> LED is off
.equ     LEDs_ON,        0x20000000   // Setting pin to 0 -> LED is on
```

# GPIO krmilnik – krmiljenje izhodov

**Potrebni koraki za krmiljenje izhoda:**

1. RCC_AHB4ENR(Peripheral Clock Register): $b_8$=1 .. Port I Enable

```
// Enable GPIOI Peripheral Clock (bit 8 in AHB4ENR register)
ldr r6, = RCC_AHB4ENR       // Load peripheral clock reg address to r6
ldr r5, [r6]                // Read its content to r5
orr r5, #0x00000100          // Set bit 8 to enable GPIOI clock
str r5, [r6]                // Store result in peripheral clock register
```

2. **MODER (Mode Register): 01: General purpose output mode**

```
// Make GPIOI Pin13 as output pin (bits 27:26 in MODER register)
ldr r6, =GPIOI_BASE       // Load GPIOD BASE address to r6
ldr r5, [r6,#GPIOx_MODER]  // Read GPIOD_MODER content to r5
and r5, #0xF3FFFFFF          // Clear bits 27-26 for P13
orr r5, #0x04000000          // Write 01 to bits 27-26 for P13
str r5, [r6]                // Store result in GPIO MODER register
```

3. Default vrednosti že ustrezne v registrih : **OTYPER, OSPEEDR ,PUPDR**

4. določi stanje izhoda s pisanjem v BSRR (ali tudi ODR) (nastavljamo na 1/0)

```
LED_ON:
 push {r5, r6, lr}
// Set GPIOx Pins to 0 (through BSSR register)
 ldr    r6, =GPIOI_BASE       // Load GPIOI BASE address to r6
 mov    r5, #LEDs_ON
 str    r5, [r6,#GPIOx_BSRR] // Write to BSRR register
 pop {r5, r6, pc}
```

```
LED_OFF:
 push {r5, r6, lr}
// Set GPIOx Pins to 1 (through BSSR register)
 ldr    r6, =GPIOI_BASE       // Load GPIOI BASE address to r6
 mov    r5, #LEDs_OFF
 str    r5, [r6,#GPIOx_BSRR] // Write to BSRR register
 pop {r5, r6, pc}
```

# GPIO krmilnik – krmiljenje izhodov – „C“

**Potrebni koraki za krmiljenje izhoda:**

1. RCC_AHB4ENR(Peripheral Clock Register): $b_8$=1 .. Port I Enable
2. **MODER (Mode Register): 01: General purpose output mode**
3. Default vrednosti že ustrezne v registrih :
   - **OTYPER (Output TYPE Register): 0: Output push-pull** (reset state)
   - **OSPEEDR (Output SPEED Register): 00 – Low speed** (reset state)
   - **PUPDR (Pull Up/Down Register): 00 – No pull** (reset state)
4. določi stanje izhoda s pisanjem v ODR ali BSRR (nastavljamo na 1/0)

**Naslovi registrov:**

```
#define RCC_AHB4ENR ((volatile uint32_t *)0x580244E0)

typedef struct {
     volatile uint32_t MODER;
     volatile uint32_t OTYPER;
     volatile uint32_t OSPEEDR;
     volatile uint32_t PUPDR;
     volatile uint32_t IDR;
     volatile uint32_t ODR;
     volatile uint32_t BSRR;
} GPIO_device;
```

```
#define GPIOC ((GPIO_device *)0x58020800)
#define GPIOD ((GPIO_device *)0x58020C00)
#define GPIOI ((GPIO_device *)0x58022000)
#define GPIOJ ((GPIO_device *)0x58022400)
```

https://rok-cesnovar.github.io/ORS/vaja_2.html

# GPIO krmilnik – krmiljenje izhodov – „C“

**Potrebni koraki za krmiljenje izhoda PD3:**

1. RCC_AHB4ENR(Peripheral Clock Register): $b_3$=1 .. Port D Enable

```
*RCC_AHB4ENR = *RCC_AHB4ENR | (1 << 3); // postavimo GPIODEN
```

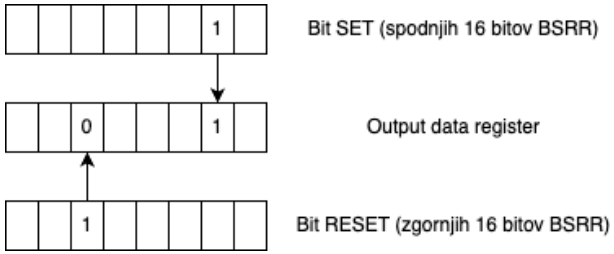
2. **MODER (Mode Register): 01: General purpose output mode**

```
// pin 3 inicializiramo kot izhod
GPIOD->MODER = GPIOD->MODER & ~(3 << (2 * 3));
GPIOD->MODER = GPIOD->MODER | (1 << (2 * 3));
```

3. Default vrednosti že ustrezne v registrih : **OTYPER, OSPEEDR ,PUPDR**

4. določi stanje izhoda s pisanjem v BSRR (ali tudi ODR) (nastavljamo na 1/0)

```
// prizgemo LED
    GPIOD->BSRR = 1 << 3;
// ugasnimo LED
    GPIOD->BSRR = 1 << (3 + 16);
```

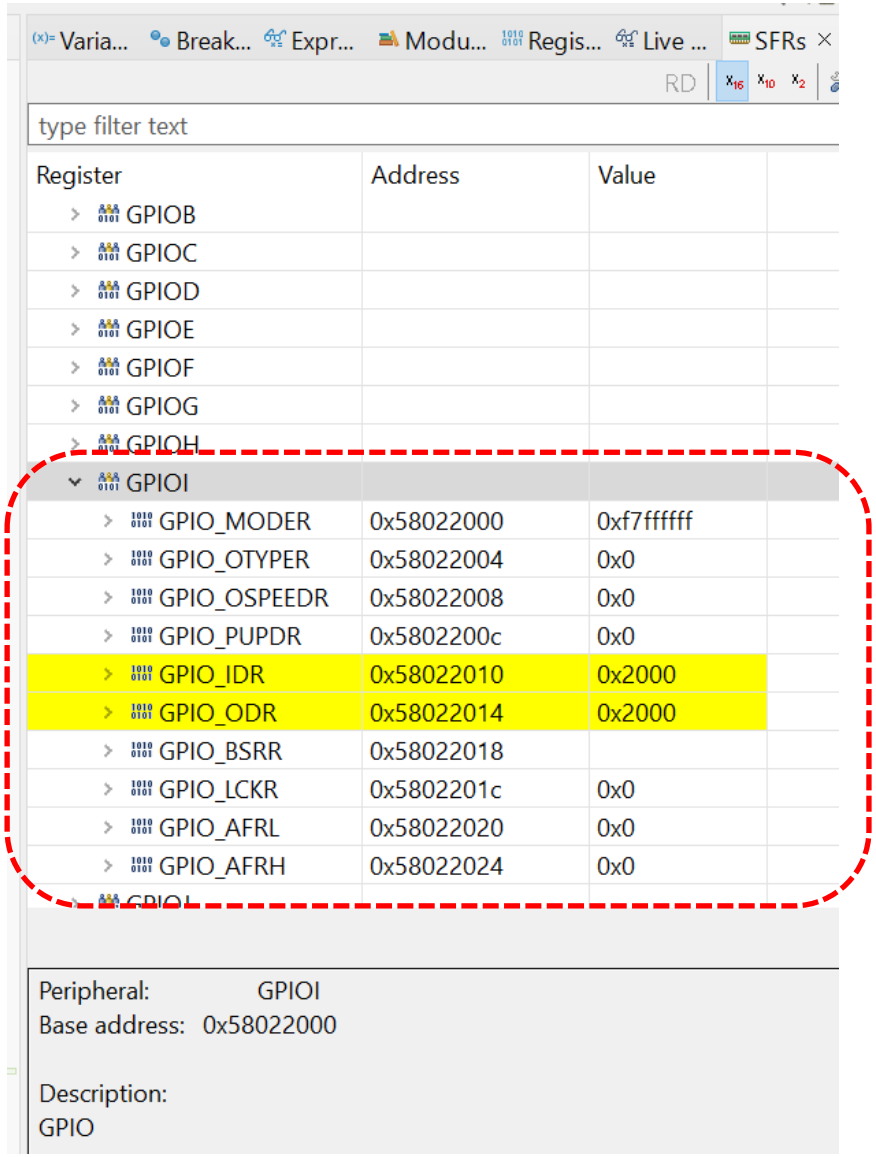



https://rok-cesnovar.github.io/ORS/vaja_2.html

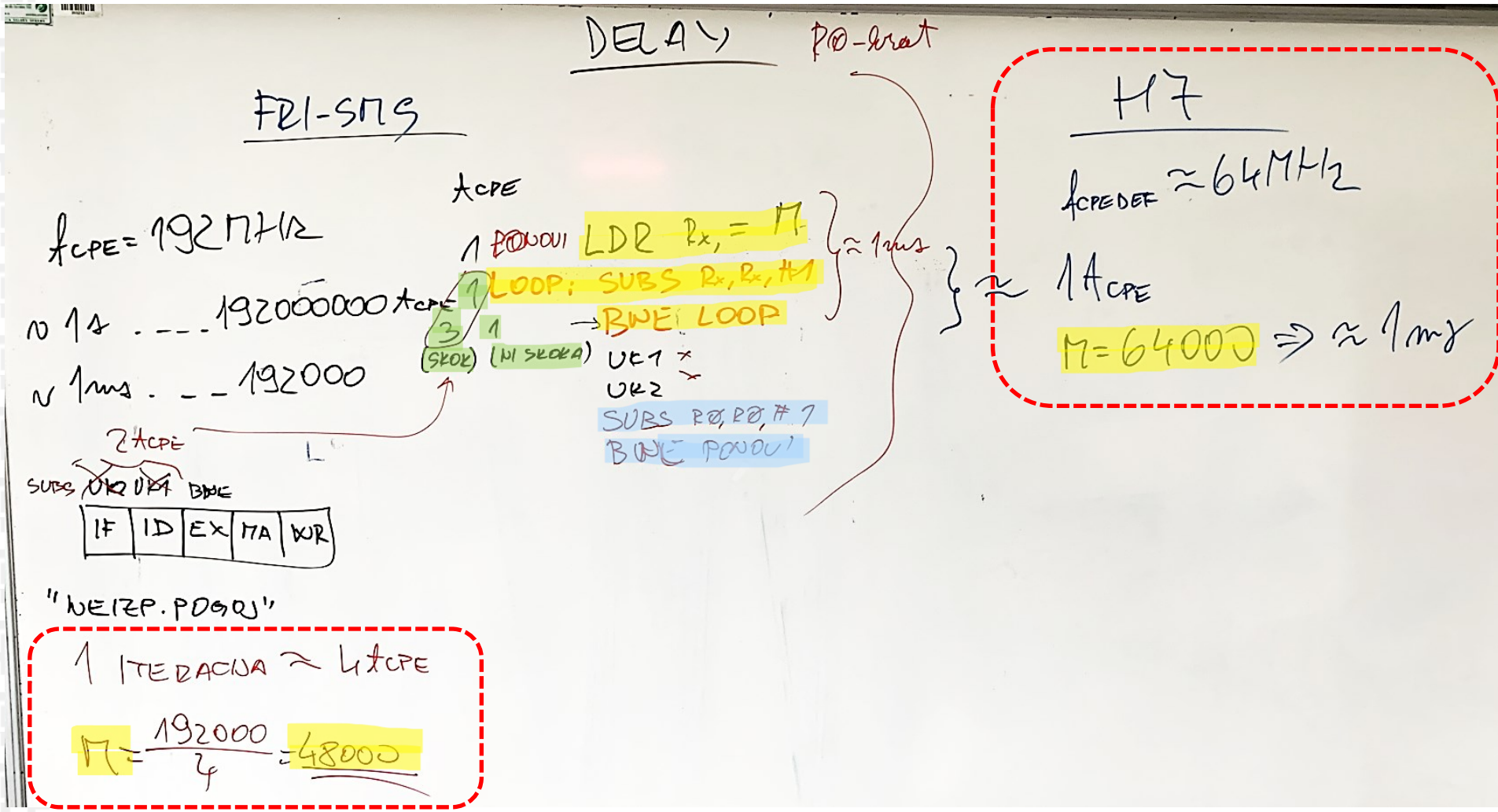
# CubeIDE – Registers okno

| Name | Value |
|---|---|
| General Registers | |
| r0 | 0xe |
| r1 | 0x3f95 |
| r2 | 0x40 |
| r3 | 0x50 |
| r4 | 0x20000030 |
| r5 | 0x0 |
| r6 | 0x0 |
| r7 | 0x0 |
| r8 | 0x0 |
| r9 | 0x0 |
| r10 | 0x0 |
| r11 | 0x0 |
| r12 | 0x0 |
| sp | 0x2001fff8 |
| lr | 0x24000305 |
| pc | 0x24000344 |
| xpsr | 0x21000000 |
| d0 | 0x0 |

# CubeIDE – SFR okno



# DELAY: zakasnitvena zanka



# DELAY: Meritve STM32H750

```
////////////////////////////////////////////////////////////////////////////////
// Meritve
////////////////////////////////////////////////////////////////////////////////

//-------------- subs,bne -------------------------
//      ldr r5,=N
// ------------- odsek kode ------------------------
// tloop:  subs r5,r5,#1
//         bne  tloop
// ------------- konec kode ------------------------
/*
// Timings - usually in second or more repetition (on first one cycles are higher)
```

| N | DWT_CYCCNT(1st) |
|---|---|
| 50 | 56 (78) |
| 100 | 106 (128) |
| 200 | 206 |
| 500 | 506 |
| 1000 | 1006 |
| 64000 | 64006 (64028) |

```
// Read DWT Counter before value
        ldr r0, [r1,#DWT_CYCCNT]

// ------------------------------
// Start of measurement code - use registers r3+ only !
// ------------------------------
tloop:  subs r5,r5,#1
        bne  tloop

// ------------------------------
//   End of measurement code
// ------------------------------

// Read DWT Counter after value
        ldr r2, [r1,#DWT_CYCCNT]

        sub r0,r2,r0    // Difference in r0
```

```
Comment: difference : r5 instructions are 16bit, r8 instructions are 32 bit, but both with same timing.
      if nop is added, for N=64000, results are 96030 and 96006
Conclusion: branch prediction is the main influencer here.
*/
```

# Utripanje LED diod(e) – Zgradba programa

```
main:
        bl INIT_IO

loop:   bl LED_ON

        mov r0,#500
        bl DELAY

        bl LED_OFF

        mov r0,#500
        bl DELAY

        b loop
```

```
INIT_IO:
        push {r5,r6,lr}
        …
        pop {r5,r6,pc}

LED_ON:
        push {r5,r6,lr}
        …
        pop {r5,r6,pc}

LED_OFF:
        push {r5,r6,lr}
        …
        pop {r5,r6,pc}

DELAY:
        push {r1,lr}
        …
        pop {r1,pc}
```