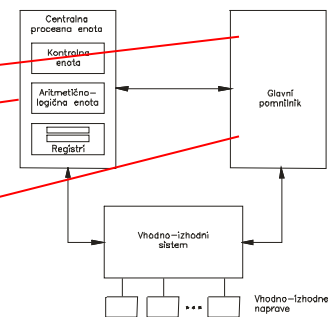


STM32H750B-DK Discovery Kit

Vhodno / izhodne naprave

GPIO Krmilnik



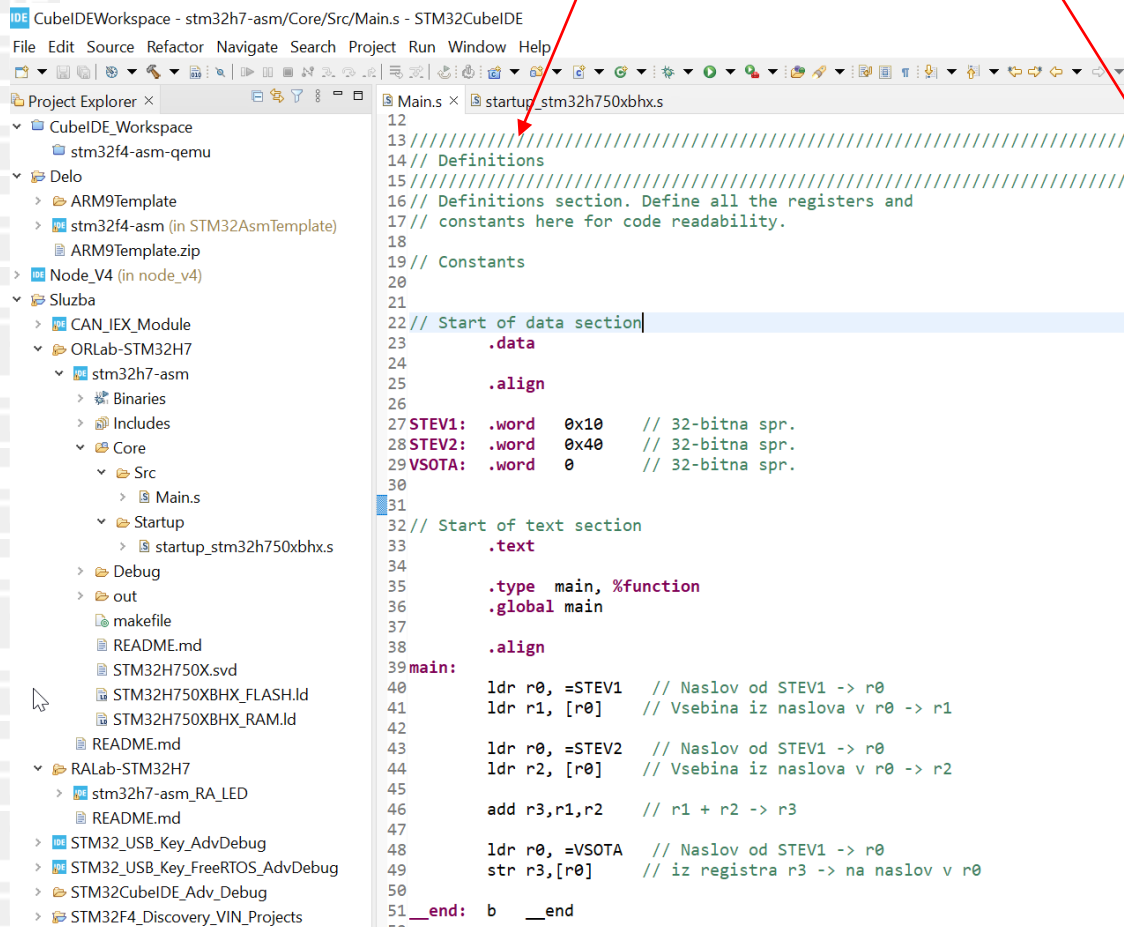
Delo na STM32H7 razvojnem sistemu

Priključitev :

- **Mikro USB** prikllop na **daljši stranici (srednji !!!)**

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

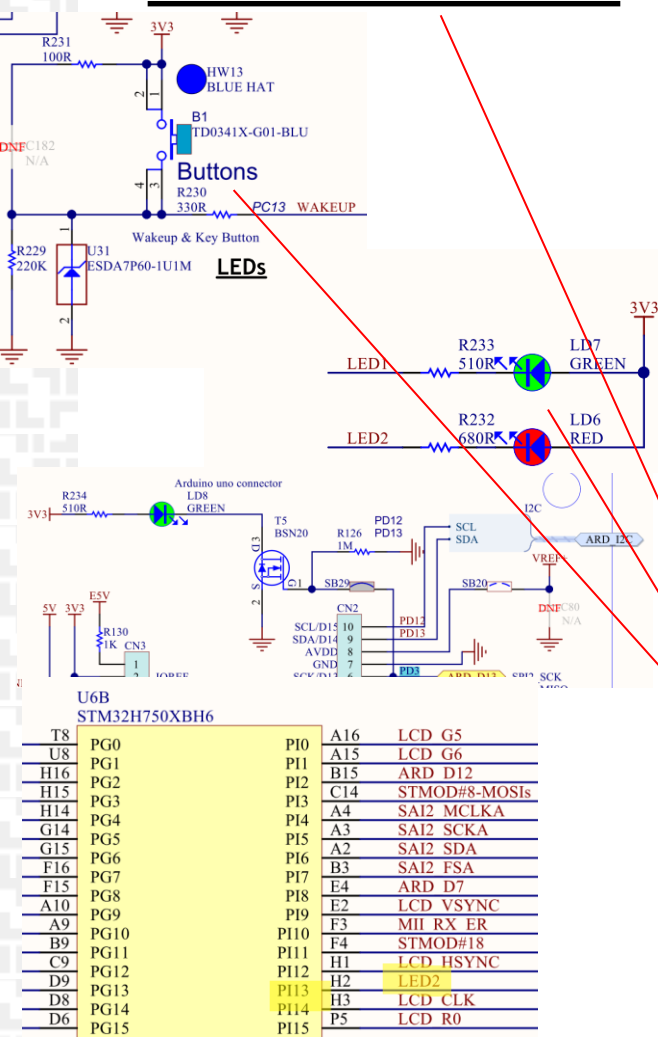
- **dodajanje vsebine (Main.s):**



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

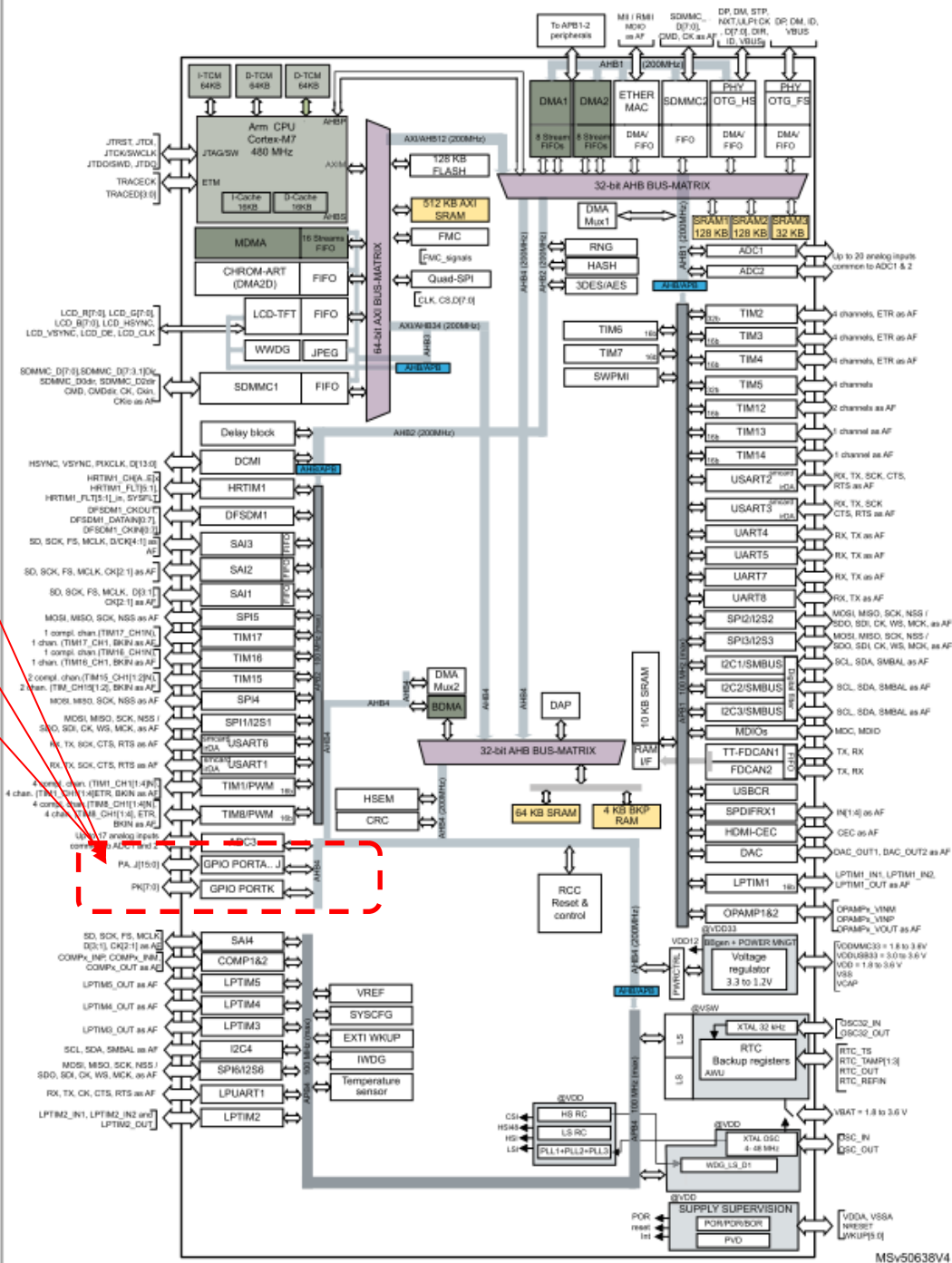
- STM32H750B-DK Discovery kit with STM32H750XB MCU
- ORLab-STM32H7 - GitHub repozitorij
- User Manual Discovery kit stm32h750xb Uploaded 11/11/22, 10.15
- DataSheet_stm32h750xb Uploaded 11/11/22, 10.16
- Reference Manual rm0433-stm32h750xb Uploaded 11/11/22, 10.17
- Programming_Manual_pm0253-stm32h750xb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

GPIO Krmilnik



LED: rdeča PI13, zelena PJ2
zelena PD3

OR – Organizacija računa



MS50638V4

Vir: RM0433 Reference manual

Strani: 131, 455,
527-545



RM0433 Reference manual

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

11

General-purpose I/Os (GPIO)	527
11.1 Introduction	527
11.2 GPIO main features	527
11.3 GPIO functional description	530
11.3.1 General-purpose I/O (GPIO)	530
11.3.2 I/O pin alternate function multiplexer and mapping	531
11.3.3 I/O port control registers	531

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).

[rm0433-stm32h750_GPIO_Ch11.pdf](#)

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	AHB4 (D3)	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

Reset and Clock Control (RCC)

RM0433

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	

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RCC	RCC	RCC	BPFRAMEN	RCC	RCC	HSEMEN	ADCCEN	RCC	RCC	BDMAEN	RCC	CRCEN	RCC	RCC	RCC
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RCC	RCC	RCC	RCC	RCC	RCC	GPIOEN	GPIOEN	GPIOEN	GPIOEN	GPIOEN	GPIOEN	GPIOEN	GPIOEN	GPIOEN	GPIOEN

OR – Organizacija računalnikov



RM0433 Rev 7

455/3319

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	

Bit 9 **GPIODEN**: GPIOD peripheral clock enable

Set and reset by software.

0: GPIOD peripheral clock disabled (default after reset)

1: GPIOD peripheral clock enabled

Bit 8 **GPIODEN**: GPIOD peripheral clock enable

Set and reset by software.

0: GPIOD peripheral clock disabled (default after reset)

1: GPIOD peripheral clock enabled

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.	GPIODEN	GPIODEN	GPIODEN	GPIODEN	GPIODEN	GPIODEN	GPIODEN	GPIODEN	GPIODEN	GPIODEN	GPIODEN
					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw



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

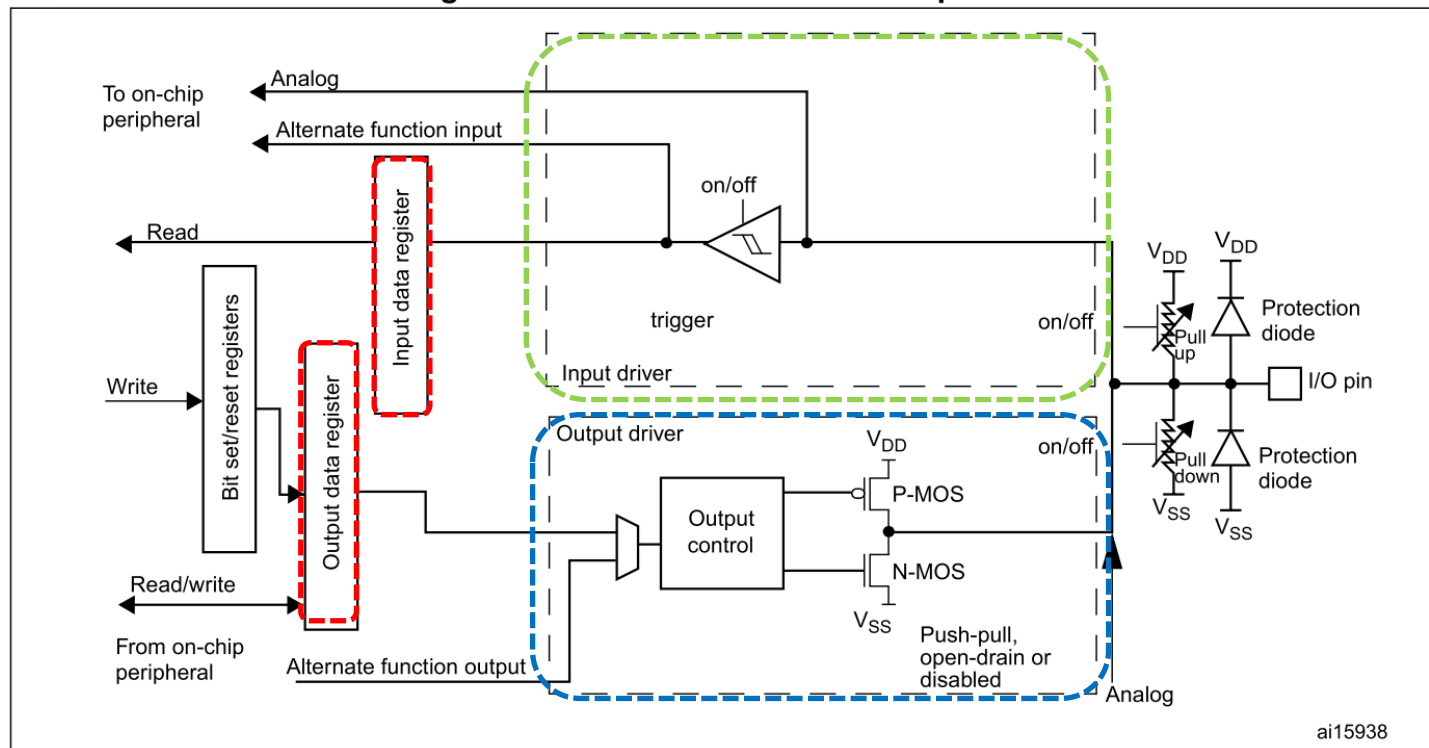
Table 8. Register boundary addresses⁽¹⁾

Boundary address	Peripheral	Bus	Register map
0x58027000 - 0x580273FF	RAMECC3	AHB4 (D3)	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		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



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(b₈=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

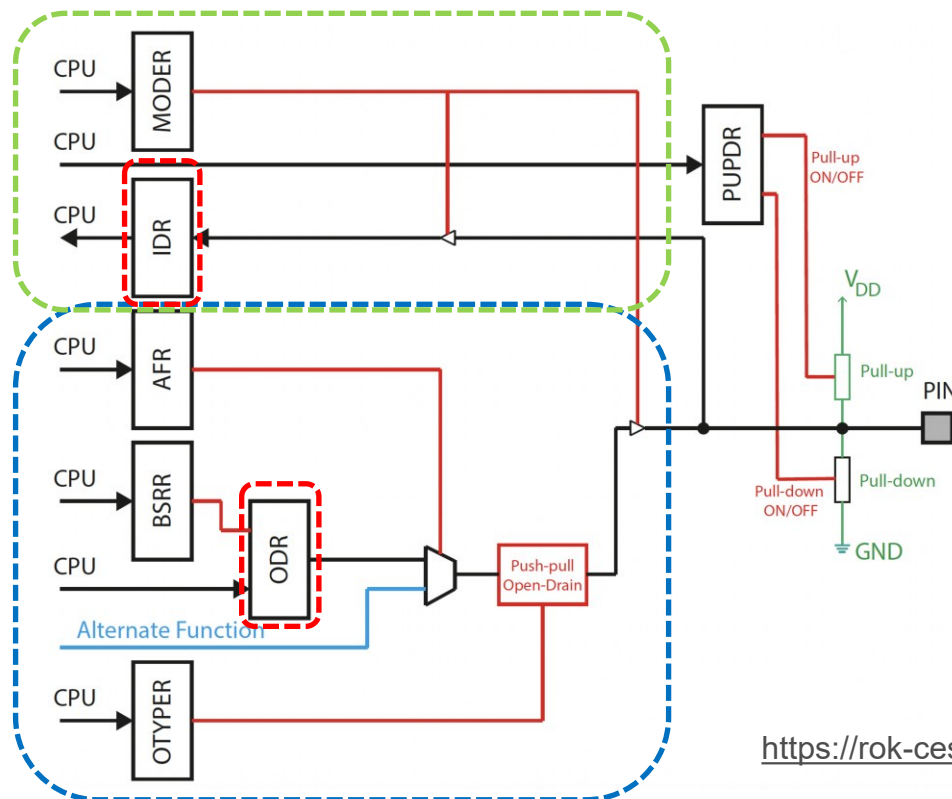
OSPEEDR (Output SPEED Register): **00** – Low speed (reset) .. **11**: Very high speed

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 – *vhod/izhod poenostavljeno*



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

Osnovni registri za GPIO priključke :

RCC_AHBxENR : vklop urinega signala (enote) : Port I: **RCC_AHB4ENR(b₈=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)

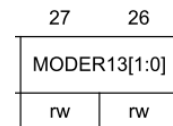
RCC_AHB4ENR(Peripheral Clock Register):

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



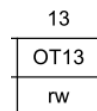
MODER (Mode Register):

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)



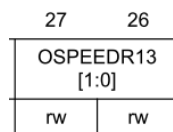
OTYPER (Output TYPE Register):

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



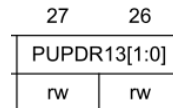
OSPEEDR (Output SPEED Register):

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



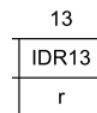
PUPDR (Pull Up/Down Register):

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



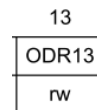
IDR (Input Data Register): **stanje vhoda 1 / 0**

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.

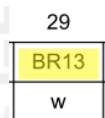


ODR (Output Data Register): **stanje izhoda 1 / 0**

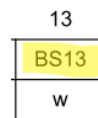
Bits 31:16 Reserved, must be kept at reset value.
 Bits 15:0 **ODR[15:0]**: Port output data I/O pin y (y = 15 to 0)
 These bits can be read and written by software.
 Note: For atomic bit set/reset, the ODR bits can be individually set and/or reset by writing to the GPIOx_BSRR register (x = A..F).



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 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.

BR13
w

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 BSRR 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
```

#define RCC_AHB4ENR ((volatile uint32_t *)0x58024...
= ...
*RCC_AHB4ENR | (1 << 8); // postavimo GPIOI

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 GPIOI BASE address to r6
ldr r5, [r6,#GPIOx_MODER]  // Read GPIOI_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 GPIOI MODER register
```

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

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

LED_ON:

```
// Set GPIOx Pins to 0 (through BSRR register)
ldr r6, =GPIOI_BASE        // Load GPIOI BASE address to r6
mov r5, #LEDs_ON           // Load GPIOI BASE address to r6
str r5, [r6,#GPIOx_BSRR]   // Write to BSRR register
```

LED_OFF:

```
// Set GPIOx Pins to 1 (through BSRR register)
ldr r6, =GPIOI_BASE        // Load GPIOI BASE address to r6
mov r5, #LEDs_OFF          // Load GPIOI BASE address to r6
str r5, [r6,#GPIOx_BSRR]   // Write to BSRR register
```

GPIO krmilnik – krmiljenje izhodov – „C“ + HAL

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)

1.

```
#define __HAL_RCC_GPIOI_CLK_ENABLE() do { \
    __IO uint32_t tmpreg; \
    SET_BIT(RCC->AHB4ENR, RCC_AHB4ENR_GPIOIEN);\
    /* Delay after an RCC peripheral clock enabling */ \
    tmpreg = READ_BIT(RCC->AHB4ENR, RCC_AHB4ENR_GPIOIEN);\
    UNUSED(tmpreg); \
} while(0)
```

2.

4.

```
HAL_GPIO_WritePin(GPIOI, GPIO_PIN_13,1);
```

```
void HAL_GPIO_WritePin(GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin, GPIO_PinState PinState)
{
    ...
    if (PinState != GPIO_PIN_RESET) {
        GPIOx->BSRR = GPIO_Pin;
    } else {
        GPIOx->BSRR = (uint32_t)GPIO_Pin << GPIO_NUMBER;
    }
}
```

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));
```

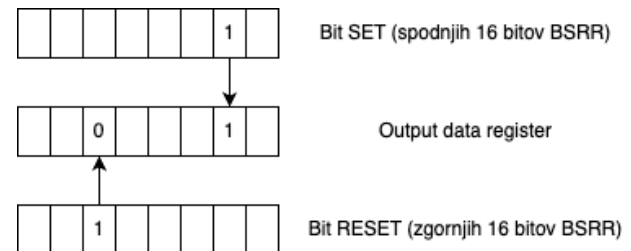
3. **Default vrednosti že ustrezne v registrih : OTYPER, OSPEEDR ,PUPDR**
4. **določi stanje izhoda s pisanjem v BSRR (ali tudi ODR) (nastavljamo na 1/0)**

```
// prizgemo LED
```

```
GPIOD->BSRR = 1 << 3;
```

```
// ugasnimo LED
```

```
GPIOD->BSRR = 1 << (3 + 16);
```



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

GPIO krmilnik – krmiljenje izhodov – „C“ + HAL

Potrebni koraki za krmiljenje izhoda:

1. **RCC_AHB4ENR**(Peripheral Clock Register): **b₈=1 .. Port I Enable**

```
/* GPIO Ports Clock Enable */  
__HAL_RCC_GPIOI_CLK_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)

```
GPIO_InitTypeDef init_structure;  
init_structure.Pin = GPIO_PIN_13;  
init_structure.Mode = GPIO_MODE_OUTPUT_PP;  
init_structure.Pull = GPIO_NOPULL;  
init_structure.Speed = GPIO_SPEED_FREQ_LOW;  
HAL_GPIO_Init(GPIOI, &init_structure);
```

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

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

```
HAL_GPIO_WritePin(GPIOI, GPIO_PIN_13,1);  
HAL_GPIO_WritePin(GPIOI, GPIO_PIN_13,0);
```

```
HAL_GPIO_TogglePin(GPIOI, GPIO_PIN_13);
```

GPIO krmilnik – krmiljenje izhodov – HAL funkcije

Definicije funkcij v HAL knjižnici:

```
HAL_RCC_GPIOI_CLK_ENABLE();
```

```
#define HAL_RCC_GPIOI_CLK_ENABLE() do { \
    __IO uint32_t tmpreg; \
    SET_BIT(RCC->AHB4ENR, RCC_AHB4ENR_GPIOIEN);\
    /* Delay after an RCC peripheral clock enabling */ \
    tmpreg = READ_BIT(RCC->AHB4ENR, RCC_AHB4ENR_GPIOIEN);\
    UNUSED(tmpreg); \
} while(0)
```

```
GPIO_InitTypeDef init_structure;
init_structure.Pin = GPIO_PIN_13;
init_structure.Mode = GPIO_MODE_OUTPUT_PP;
init_structure.Pull = GPIO_NOPULL;
init_structure.Speed = GPIO_SPEED_FREQ_LOW;
HAL_GPIO_Init(GPIOI, &init_structure);
```

```
void HAL_GPIO_Init(GPIO_TypeDef *GPIOx, GPIO_InitTypeDef
*GPIO_Init)
{
    ...
    /* Configure the IO Speed */
    temp = GPIOx->OSPEEDR;
    temp &= ~(GPIO_OSPEEDR_OSPEED0 << (position * 2U));
    temp |= (GPIO_Init->Speed << (position * 2U));
    GPIOx->OSPEEDR = temp;

    /* Configure the IO Output Type */
    temp = GPIOx->OTYPER;
    temp &= ~(GPIO_OTYPER_OT0 << position);
    temp |= ((GPIO_Init->Mode & OUTPUT_TYPE) >>
    OUTPUT_TYPE_Pos) << position);
    GPIOx->OTYPER = temp;
}
```

```
HAL_GPIO_WritePin(GPIOI, GPIO_PIN_13,1);
```

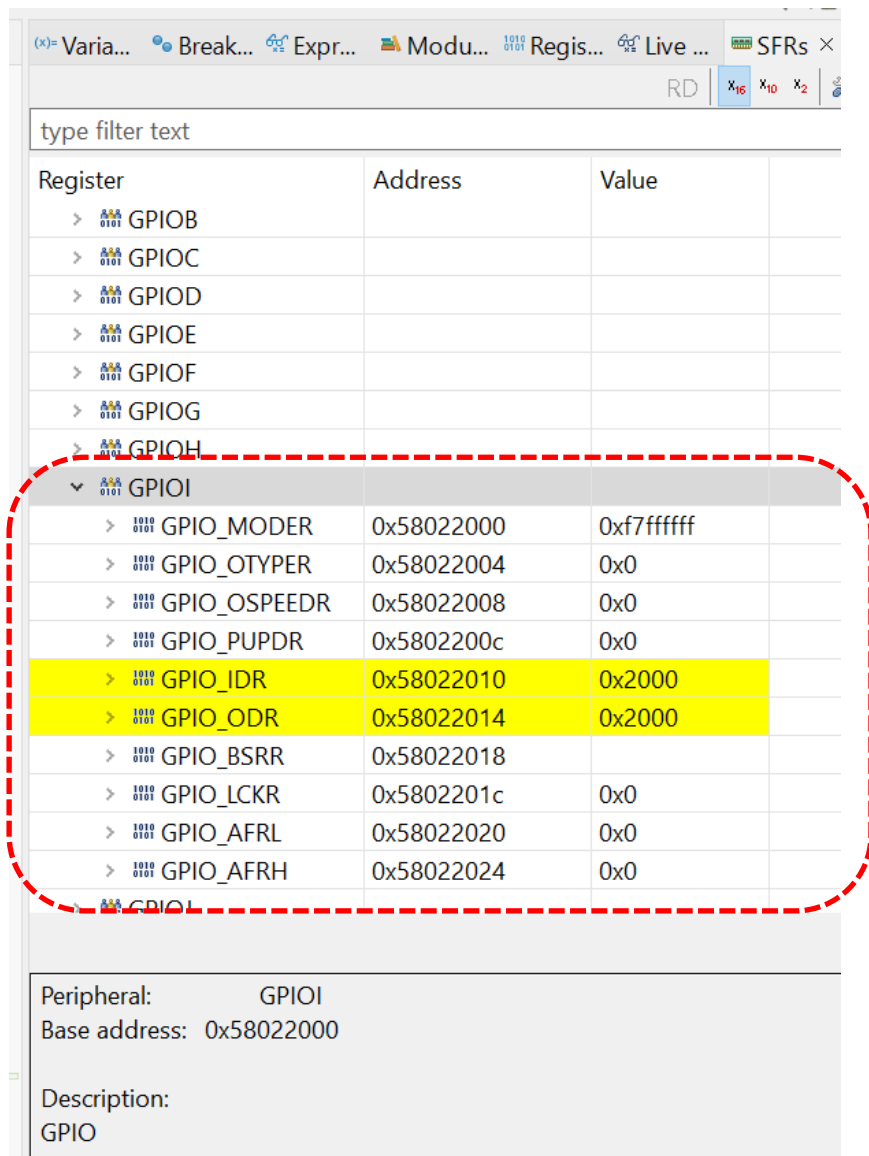
```
void HAL_GPIO_WritePin(GPIO_TypeDef *GPIOx, uint16_t GPIO_Pin,
GPIO_PinState PinState)
{
    ...
    if (PinState != GPIO_PIN_RESET){
        GPIOx->BSRR = GPIO_Pin;
    }else{
        GPIOx->BSRR = (uint32_t)GPIO_Pin <<GPIO_NUMBER;
    }
}
```

```
if ((GPIO_Init->Mode & GPIO_MODE) != MODE_ANALOG)
{
    /* Configure IO Direction mode (Input, Output, Alternate or
    Analog) */
    temp = GPIOx->MODER;
    temp &= ~(GPIO_MODER_MODE0 << (position * 2U));
    temp |= ((GPIO_Init->Mode & GPIO_MODE) << (position * 2U));
    GPIOx->MODER = temp;
}
```

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



The screenshot shows the SFRs window in CubeIDE. The top toolbar includes buttons for 'Varia...', 'Break...', 'Expr...', 'Modu...', 'Regis...', 'Live ...', and 'SFRs'. Below the toolbar is a search bar labeled 'type filter text'. The main table lists SFRs with columns for 'Register', 'Address', and 'Value'. The 'GPIOI' peripheral is expanded, showing its registers. The 'GPIO_IDR' and 'GPIO_ODR' registers are highlighted in yellow. A red dashed rounded rectangle encloses the expanded GPIOI section.

Register	Address	Value
> GPIOB		
> GPIOC		
> GPIOD		
> GPIOE		
> GPIOF		
> GPIOG		
> GPIOH		
▼ GPIOI		
> GPIO_MODER	0x58022000	0xf7ffffff
> GPIO_OTYPER	0x58022004	0x0
> GPIO_OSPEEDR	0x58022008	0x0
> GPIO_PUPDR	0x5802200c	0x0
> GPIO_IDR	0x58022010	0x2000
> GPIO_ODR	0x58022014	0x2000
> GPIO_BSRR	0x58022018	
> GPIO_LCKR	0x5802201c	0x0
> GPIO_AFR1	0x58022020	0x0
> GPIO_AFRH	0x58022024	0x0

Peripheral: GPIOI
Base address: 0x58022000

Description:
GPIO