STM32 mikrovezérlők programozása STM32CubeIDE környezetben

2. HAL GPIO függvények, külső megszakítások

Felhasznált és ajánlott irodalom

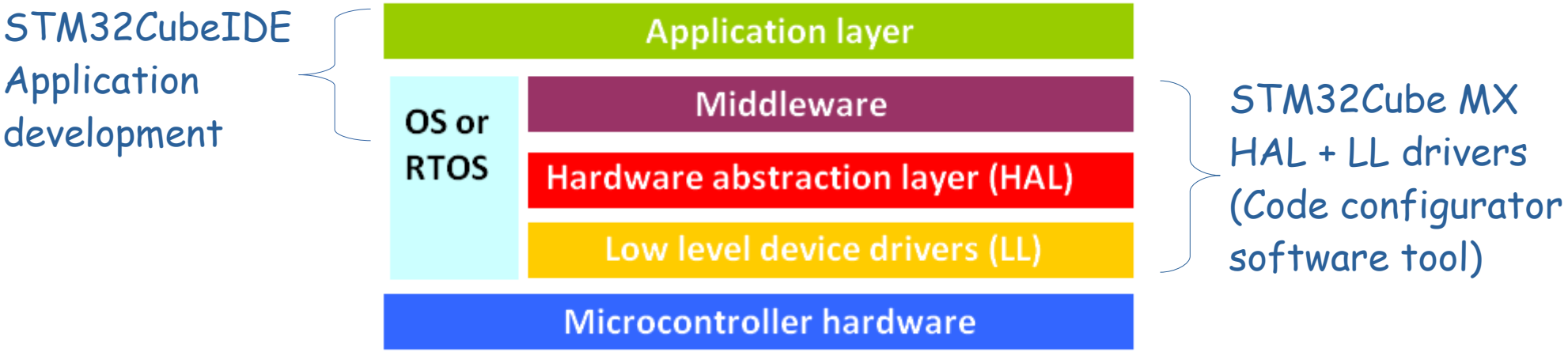
- Joseph Yiu: **The Definitive Guide To The ARM Cortex-M3 and Cortex-M4**
- Muhammad Ali Mazidi, Shujen Chen, Eshragh Ghaemi: STM32 Arm Programming for Embedded Systems
- Carmine Noviello: **Mastering STM32**
- EMCU: **First embedded program for STM32 mcu using STM32CubeIDE**
- DeepBlueMbedded: **STM32 Embedded tutorials**

Adatlapok, kézikönyvek:

- **STM32F103C8 adatlap és termékinfo**
- **STM32F103 Family Reference Manual**
- **STM32F446RE adatlap és termékinfo**
- **STM32F446** Family Reference Manual
- **STmicro: Description of STM32F4 HAL and low-layer drivers**

Az alkalmazásfejlesztés szintjei

- Először az **STM32Cube MX** (Code Configurator Software Tool) és a **HAL + LL** firmware könyvtárak segítségével konfiguráljuk a használni kívánt hardver elemeket
- Majd az **STM32CubeIDE** környezetben elkészítjük az alkalmazást, illetve szükség esetén kiegészítjük a ***middleware*** réteget (utóbbiba tartoznak a kommunikációs protokolok, fájlrendszerek vagy kijelzők kezelése stb.)



STM32 HAL és LL programkönyvtárak

- Az **STM32Cube** hardver absztrakciós programkönyvtára (HAL), egy STM32 absztrakciós szofver réteg, amely maximális hordozhatóságot biztosít az STM32 mikrovezérlő termékcsaládok között, s minden hardver perifériájuk elérését lehetővé teszi
- A **HAL** magas szintű és tulajdonság-orientált API-t biztosít, jó hordozhatósággal, elfedve a felhasználó elől az MCU és a perifériák bonyolult részleteit
- Az alacsony szintű API (LL) gyors és kis helyfoglalású szoftver réteget biztosít, amely hardver-közelibb, mint a HAL.
  Az LL APIs csak korlátozott perifériakészlet elérését teszi lehetővé
- Az **LL** alacsony szintű API-t nyújt, regiszter szintű eléréssel, ami hatékony, de kevésbé hordozható programozást biztosít. Ugyankkor az MCU és a perifériák mélyebb ismeretét kívánja a felhasználótól

STM32 HAL_GPIO függvények

- Inicializáló és de-inicializáló függvények
  - **HAL_GPIO_Init** – port/portkivezetés inicializálása
  - **HAL_GPIO_DeInit** – port/portkivezetés alaphelyzetbe állítása
- I/O műveletek függvényei
  - **HAL_GPIO_ReadPin** – portkivezetés állapotának beolvasása
  - **HAL_GPIO_WritePin** – portkivezetés állapotának írása
  - **HAL_GPIO_TogglePin** – portkivezetés állapotának átbillentése
  - **HAL_GPIO_LockPin** – portkivezetés konfigurálásának rögzítése
  - **HAL_GPIO_EXTI_IRQHandler** – külső megszakítás kiszolgálása
  - **HAL_GPIO_EXTI_Callback** – külső megszakítás visszahívási függvénye

Lekérdezés vagy programmegszakítás?

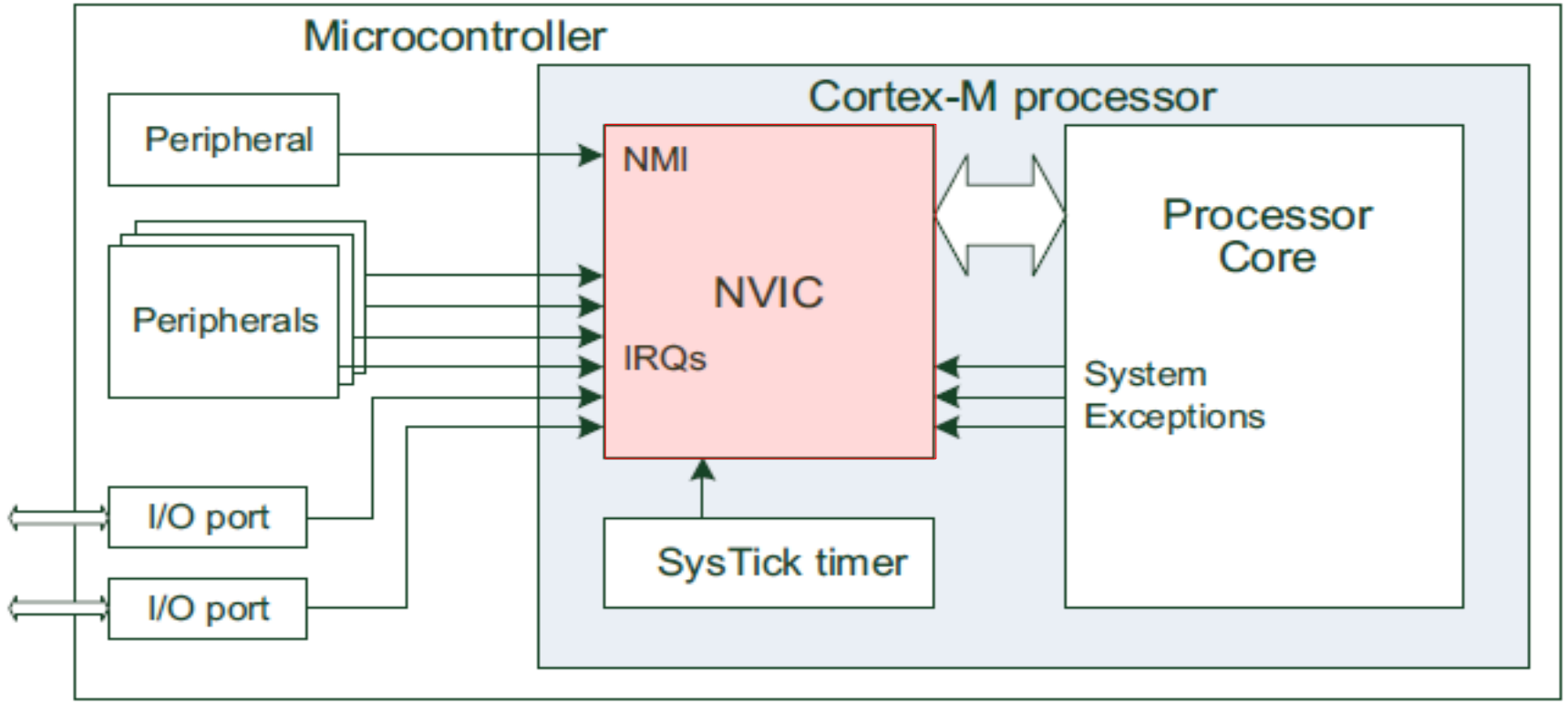
- **A lekérdezéses mód *(polling)*** olyan, mintha a telefonba időnként belehallóznánk, hogy van-e ott valaki. Ez egyszerű módszer, de nem hatékony

- **Programmegszakítás *(interrupt):*** egy hardver esemény bekövetkezte megszakítja a program futását, kiszolgálja az eseményt, majd visszatér
- **A programmegszakítás** olyan, mint amikor a telefon csöng. Abbahagyjuk, amit éppen csináltunk, s felvesszük a kagylót. A telefonálás után ott folytatjuk a korábbi tevékenységet, ahol félbeszakítottuk, amikor a telefont felvettük

- A megszakított tevékenység lehet ”alvás” (energiatakarékos mód) is, ha ébresztési **eseményt *(event)*** keltünk

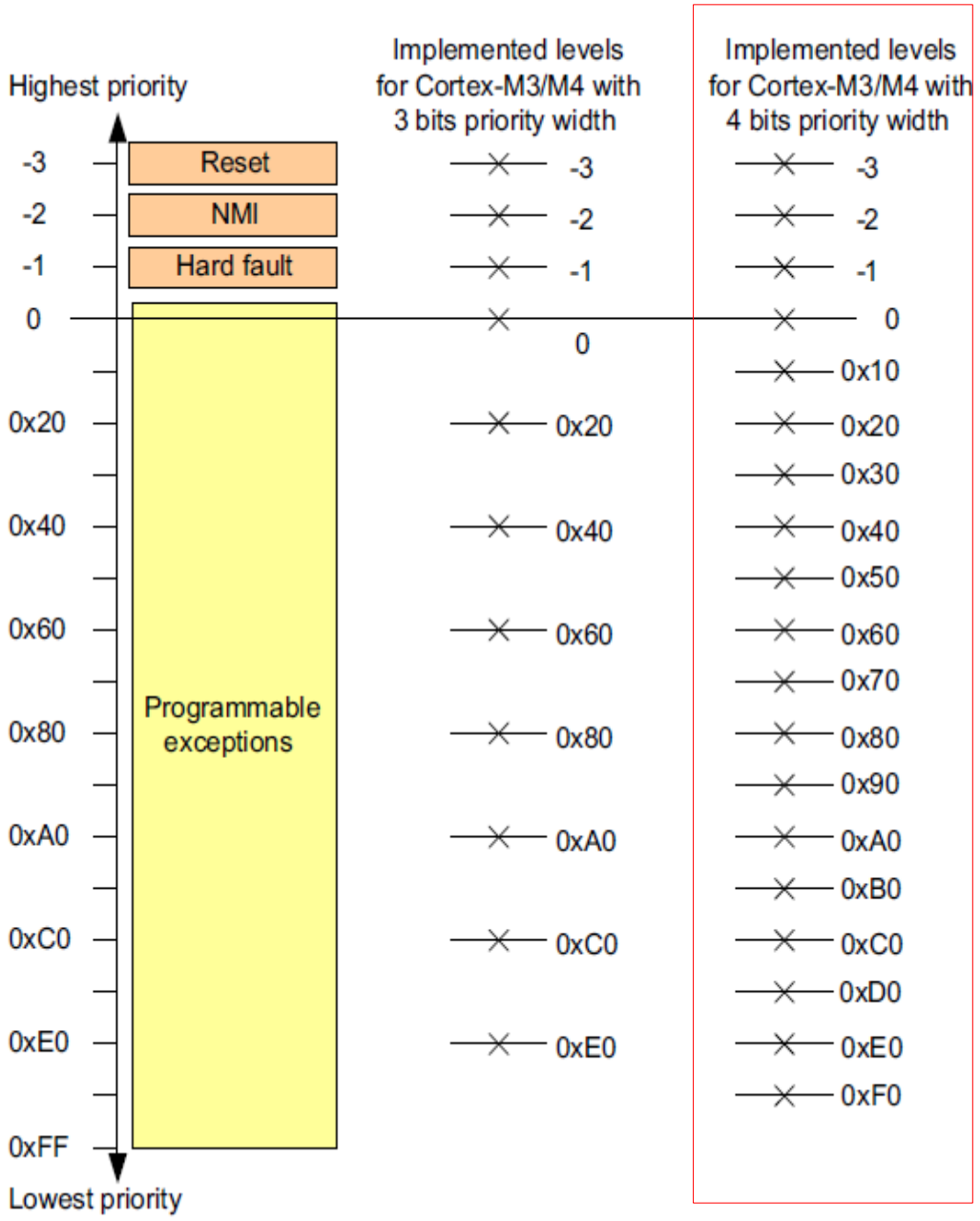
ARM Cortex-M megszakítási rendszer

- Az MCU megszakítás vezérlője (**NVIC** = Nested and Vectored Interrupt Controller) számos forrásból fogad megszakítási kérelmeket
- A megszakítási rendszer külön belépési pontot (vektor) és beállítható prioritást rendel az egyes megszakításokhoz
- A magasabb prioritású megszakítás az alacsonyabb prioritású megszakítás kiszolgálását is megszakíthatja



Megszakítások prioritása

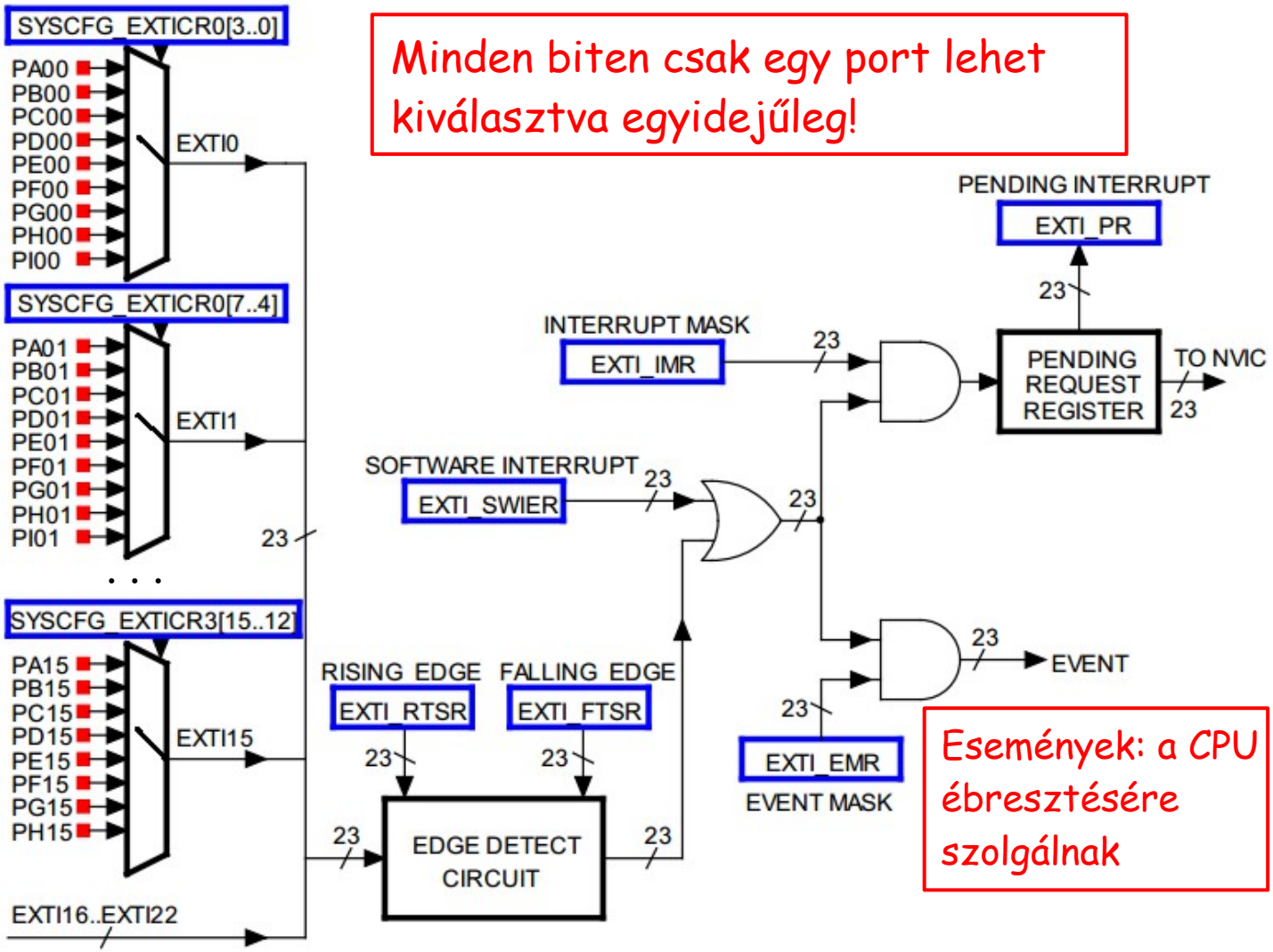
- A Cortex-M3 és Cortex-M4 processzorok felépítése három fix, magas prioritású és legfeljebb 256 szintű programozható prioritási szintet definiál
- Az **STM32F446RE** MCU 4-bites megszakítási prioritás beállítást implementál



Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Implemented				Not Implemented			

Külső megszakítások (EXTI) kezelése

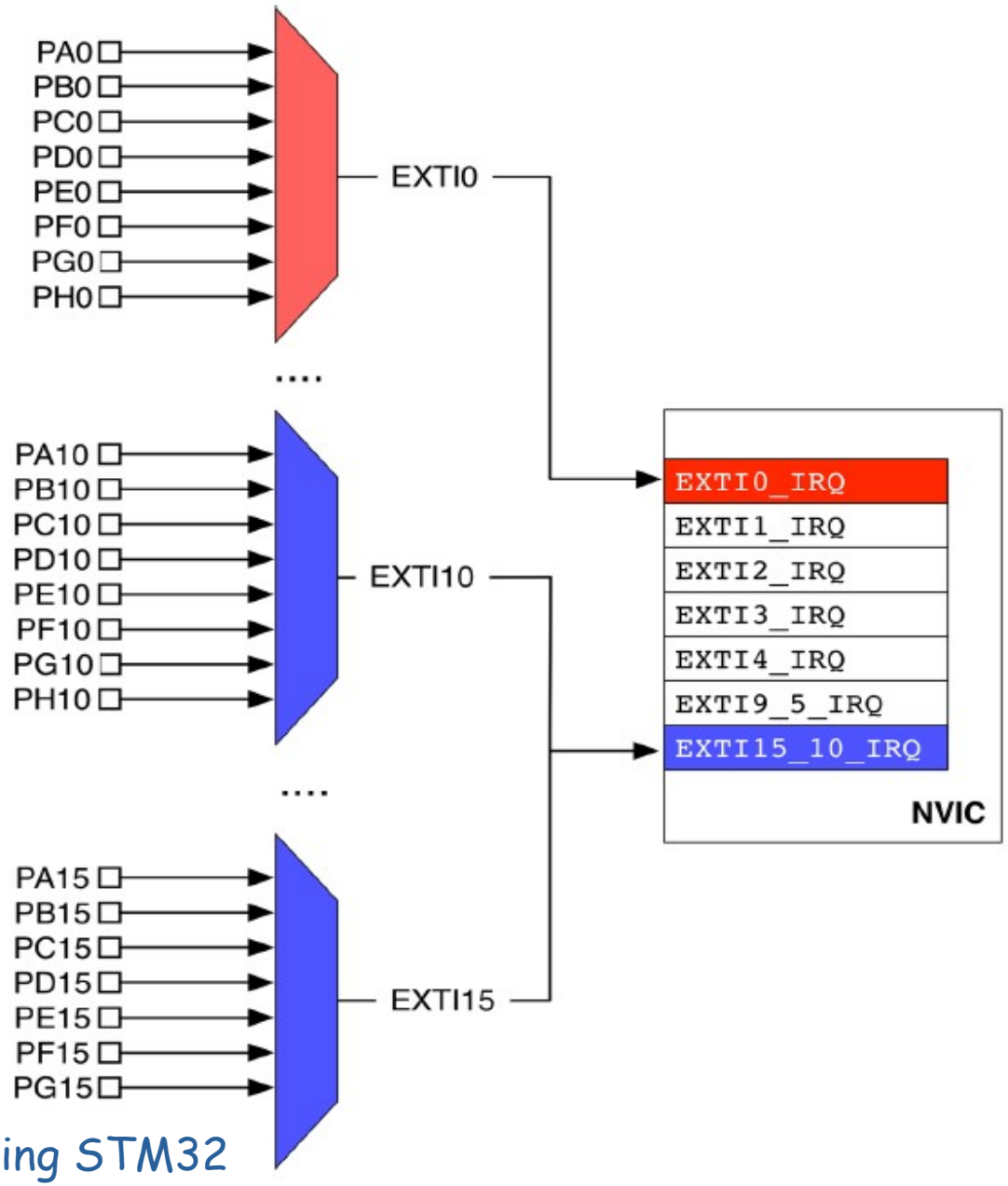
- Az **EXTI** modul a külső megszakítások vezérlője, 23 bemenete közül 16 a GPIO vonalakat fogadja (a portokat multiplexelve)
- Az EXTI16-EXTI22 vonalak egyéb eseményekre reagálnak
- A megszakításokat az élkiválasztást és az ébresztést az EXTI modul regiszterei vezérlik
- A megszakítási jel az **NVIC**-be jut



A kép forrása: https://www.cnblogs.com/shangdawei/p/4723789.html

A megszakítási vektorok kiosztása

- Az **NVIC** megszakításvezérlő csak 7 megszakítási vektort biztosít, az **EXTI5 – EXTI9** és az **EXTI10 – EXTI15** vonalak csak egy-egy vektort kaptak
- Engedélyezett megszakítás esetén az **EXTI_PR** regiszter megfelelő bitje jelzi a megszakítási kérelmet, ennek alapján azonosítható a közös vektorba „beeső” megszakítás eredete



Az ábra forrása: Carmine Noviello, Mastering STM32

Lab02 projektek: GPIO portok kezelése

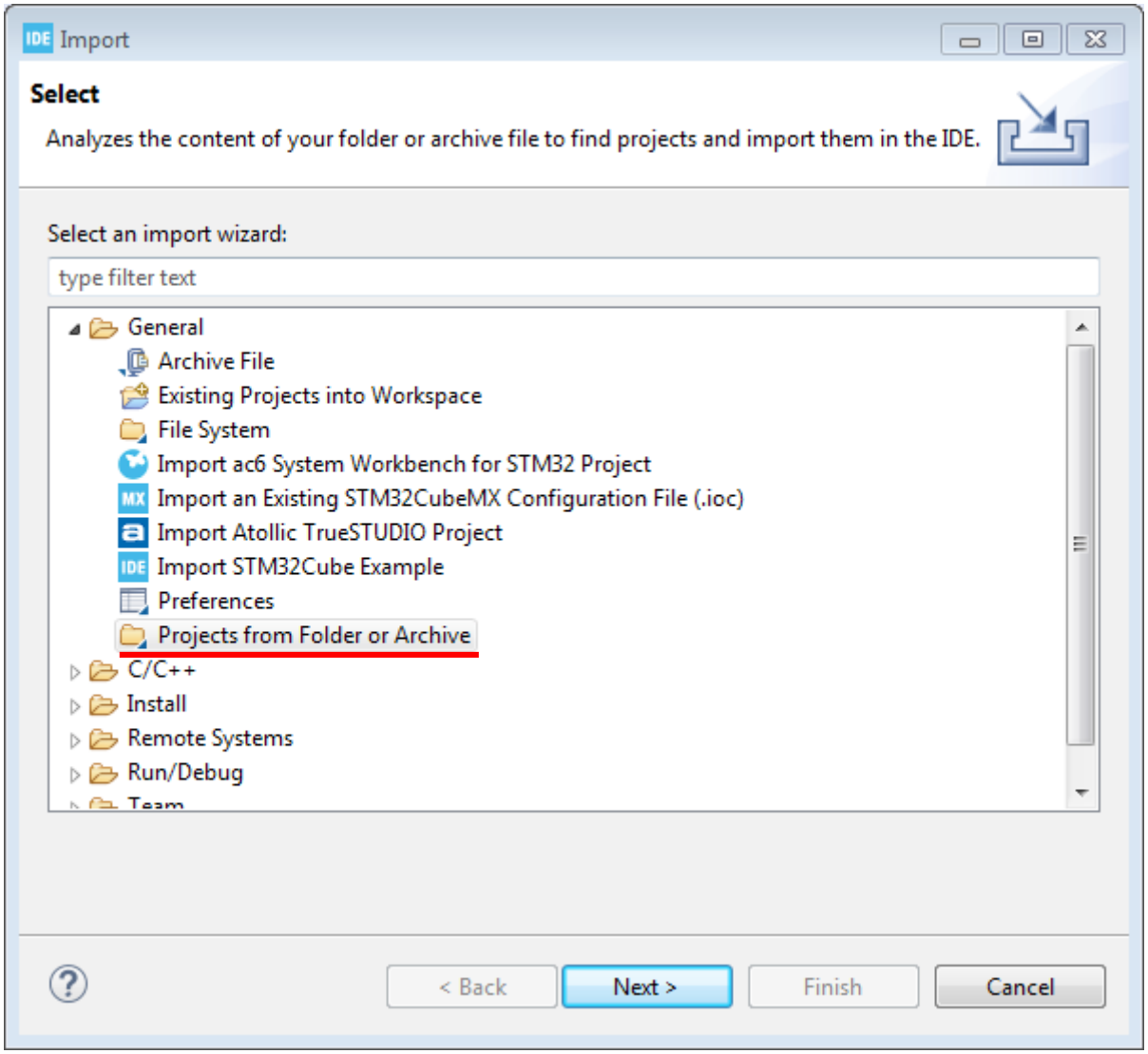
- **Lab02/F446RE_b1int:** LED ki/bekapcsolása nyomógombbal, programmegszakítással a Nucleo-F446RE kártyán (STM32F446RE)
- **Lab02/F446RE_b2int:** LED ki/bekapcsolása két nyomógombbal, programmegszakítással a Nucleo-F446RE kártyán (STM32F446RE)
- **Lab02/F446RE_b3int:** LED villogtatás ki/bekapcsolása két nyomógombbal, programmegszakítással a Nucleo-F446RE kártyán (STM32F446RE)
- **Lab02/F446RE_b4int:** A Nucleo-F446RE kártyára ültetett Arduino-kompatibilis multifunkciós kártya LED4 kijelzőjének vezérlése nyomógombokkal (STM32F446RE)
- Az **STM32F103C8** mikrovezérlőre (Blue pill kártya) is adaptáltuk a fenti projekteket. A hardver eltérések rövid leírása a 40. oldalon található

Meglevő projekt klónozása

- Nem kell mindig a kályhától indulni, klónozhatjuk valamelyik korábbi projektünket, s akkor csak az eltéréseket kell konfigurálni
- A munkaterület mappájában hozzunk létre egy új alkönyvtárat **Lab02** néven, ebben kerülnek majd bele a mai előadás példaprogramjai!
- Másoljuk bele négyszer a múltkori **Lab01/F446RE_button** projektet, rendre **F446RE_b1int, F446RE_b2int, F446RE_b3int,** valamint **F446RE_b4int** névre átnevezve!
- Mind a négy projekt mappájában:
  - A **.project** állományban írjuk át a projekt nevét!
  - Nevezzük át a **F446RE_button.ioc** állományt és módosítsuk benne a projekt nevét!
  - Töröljük az **.mxproject** és a **.launch** kiterjesztésű állományokat!
  - Töröljük a **Debug** mappa tartalmát (ha nem üres)!

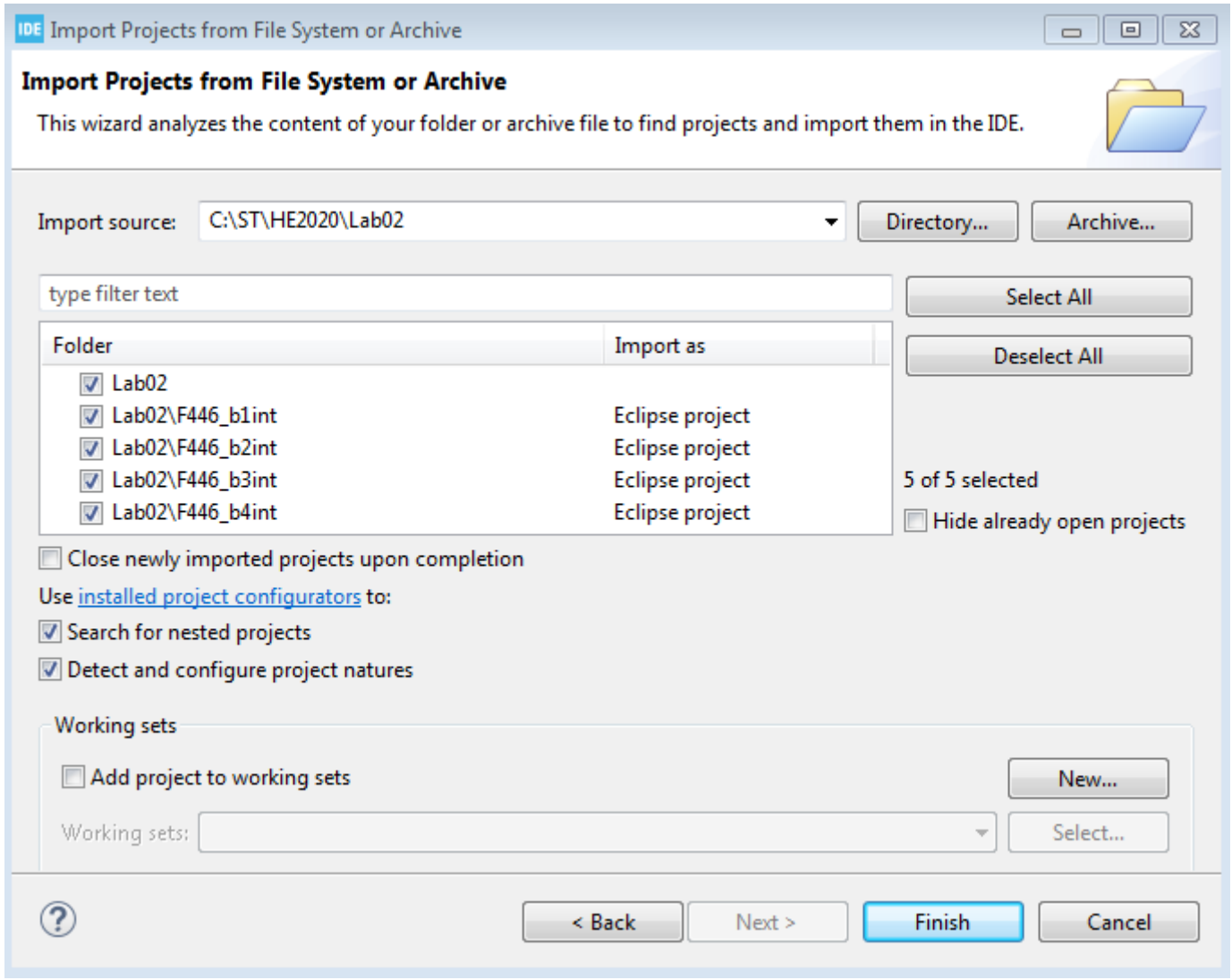
A klónozott projektek importálása 1.

- Az STM32CubeIDE **File** menüjének **Import** menüpontját válasszuk, majd a felugró ablakban a **Projects from Folder or Archive** opciót!
- Ezután kattintsunk a **Next** gombra!



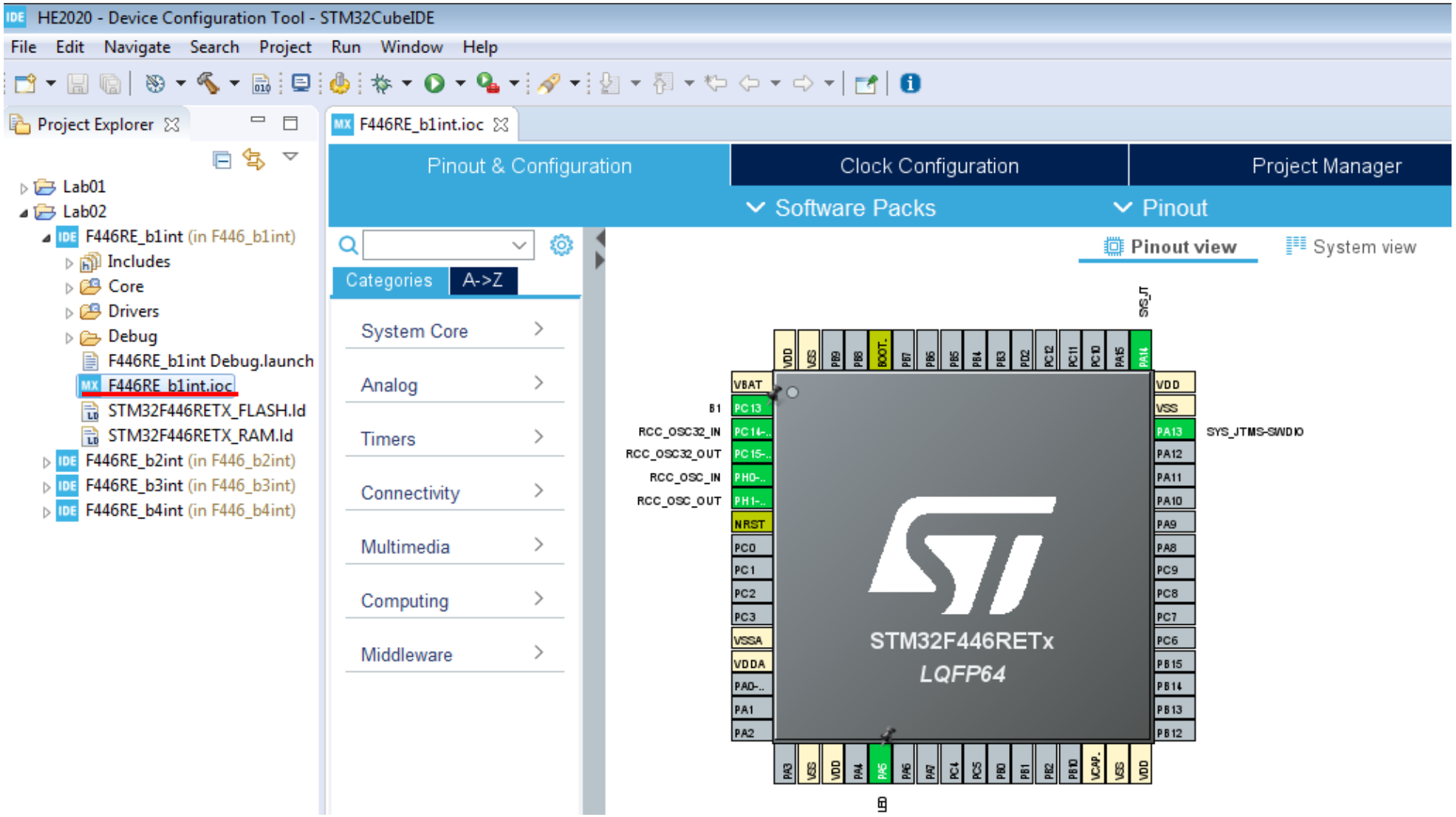
A klónozott projektek importálása 2.

- A **Directory** feliratú gombra kattintva tallózzuk a **Lab02** mappát, majd kattintsunk a **Finish** gombra!



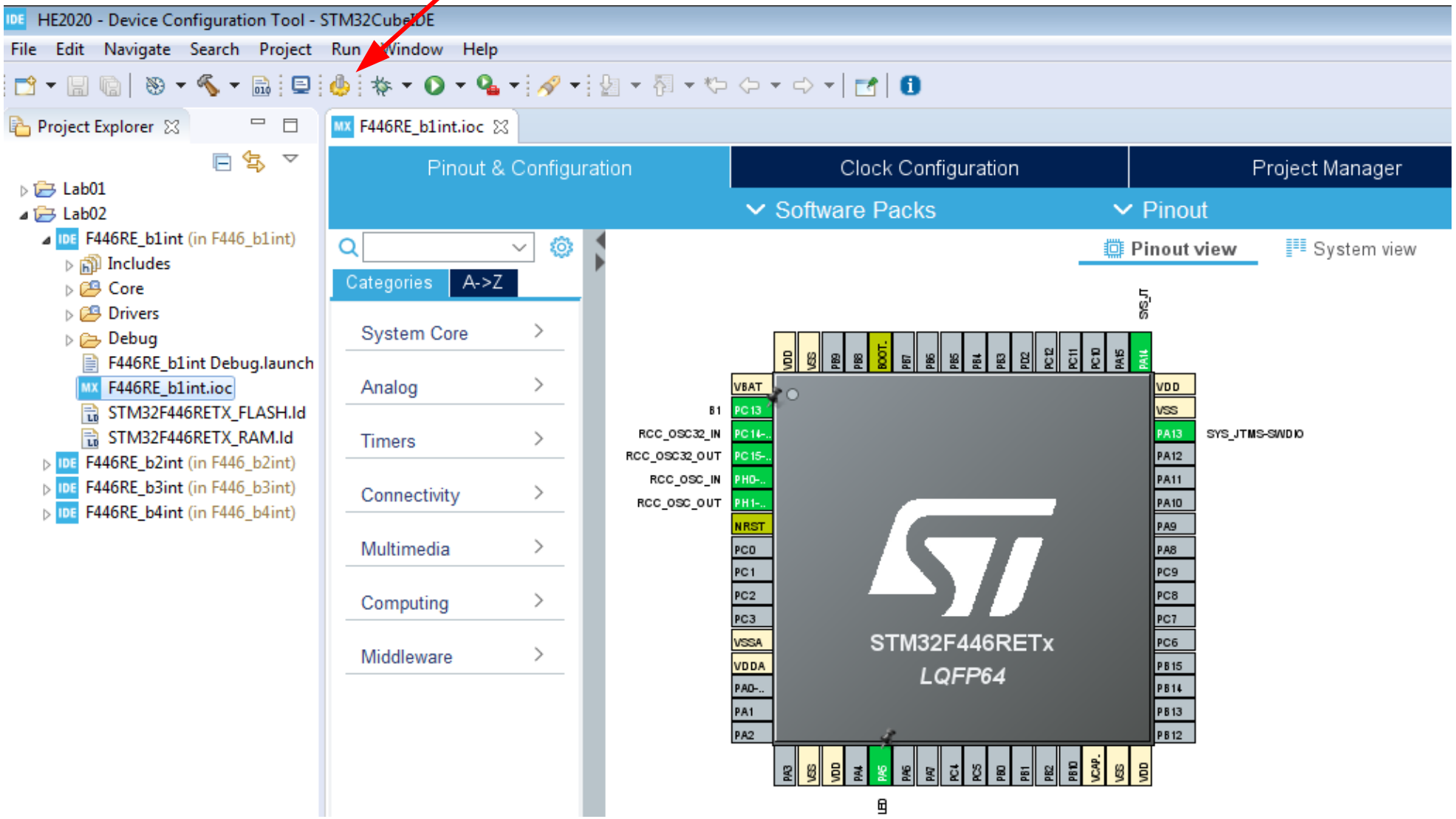
A klónozott projektek módosítása

- A projektben az **.ioc** állományra történő dupla kattintással nyissuk meg a **Device Configuration Tool** ablakot és módosítsuk, amit szükséges



A klónozott projektek újragenerálása

- A kívánt módosítások után generáljuk újra a kódot a fogaskerék ikonra kattintva!

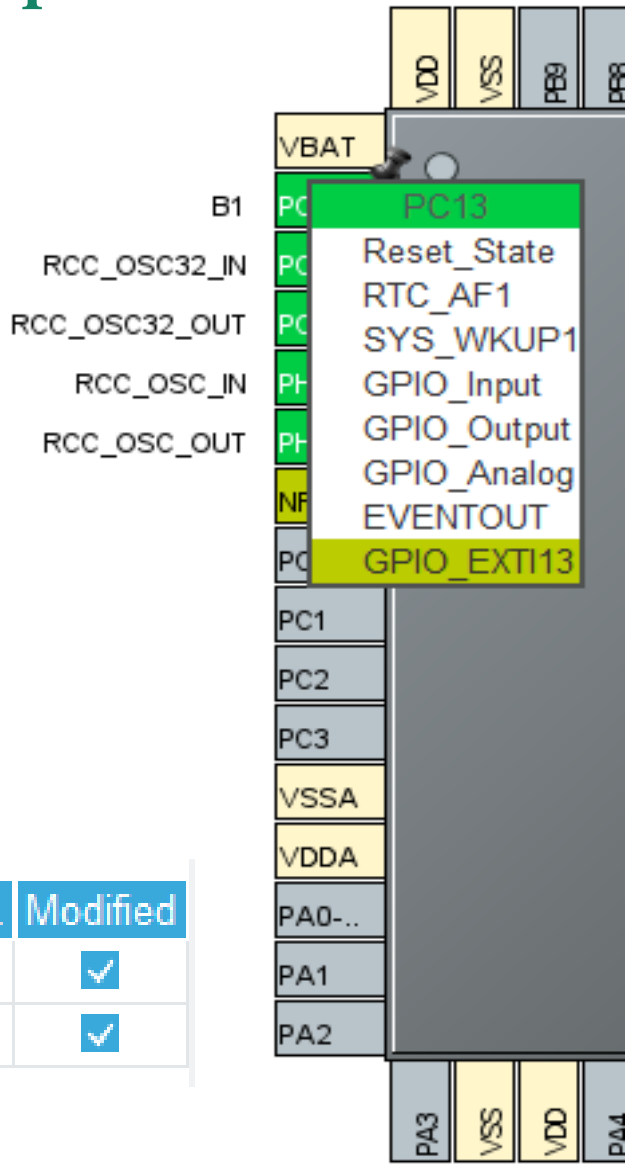


A Lab02/F446RE_b1int projekt

- A **NUCLEO-F446RE** kártya beépített **LD2** LED-jét (**PA5**) kapcsolgatjuk ki-be a beépített **B1** nyomógomb (**PC13**) minden lenyomásakor
- Konfiguráljuk push-pull digitális kimenetként a **PA5** portbitet!
- Konfiguráljuk a **PC13** bemenetet `GPIO_MODE_IT_FALLING` módba, hogy minden lefutó élre (lenyomás) generáljon egy megszakítást!
- Engedélyezzük az **EXTI13**-hoz tartozó `EXTI15_10_IRQn` megszakítási vektort!
- Végül definiáljuk a `void EXTI15_10_IRQHandler()` függvényt, ami az `EXTI15_10_IRQn` megszakítási vektor kiszolgáló függvénye, s ebben minden alkalommal billentsük át a **PA5** kimenetet és ne felejtsük el törölni az EXTI_PR regiszter 13. bitjét
- A HAL programkönyvtár absztrakciós mechanizmusa révén elkerülhetjük a technikai részletekkel való bíbelődést

A GPIO kivezetések konfigurálása

- A **PA5** kivezetést a szokásos módon **GPIO_Output** módba állítjuk
- Opciók: **push-pull** kimeneti mód választás, alaphelyzetben **Low** állapotban legyen, alacsony frekvencia, és a **LED** nevet kapta
- A **PC13** kivezetést külső megszakítást fogadó bemenetnek állítjuk (**GPIO_EXTI13** mód)
- Opciók: lefutó élre érzékeny bemenet belső felhúzással (pull_up), a bemenethez a **B1** nevet rendeltük



Pin Name	...	GPIO ...	GPIO mode	GPIO Pull-u...	Maxi...	User L...	Modified
PA5	n/a	Low	Output Push Pull	No pull-up a...	Low	LED	☑
PC13	n/a	n/a	External Interrup...	Pull-up	n/a	B1	☑

F446RE_b1int main.c : GPIO konfigurálás

```
static void MX_GPIO_Init(void) {
  GPIO_InitTypeDef GPIO_InitStruct = {0};
  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOC_CLK_ENABLE();
  __HAL_RCC_GPIOH_CLK_ENABLE();
  __HAL_RCC_GPIOA_CLK_ENABLE();

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, GPIO_PIN_RESET);

  /*Configure GPIO pin : B1_Pin (PC13) */
  GPIO_InitStruct.Pin = B1_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_IT_FALLING;
  GPIO_InitStruct.Pull = GPIO_PULLUP;
  HAL_GPIO_Init(B1_GPIO_Port, &GPIO_InitStruct);

  /*Configure GPIO pin : LED_Pin (PA5) */
  GPIO_InitStruct.Pin = LED_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(LED_GPIO_Port, &GPIO_InitStruct);
}
```

A GPIO portok órajelének engedélyezése

Az MX_GPIO_Init függvényt az STM32Cube MX eszköz generálta, az előző oldalon leírt beállítások hatására. Változtatás nélkül használjuk.

F446RE_b1int projekt main.c részlet

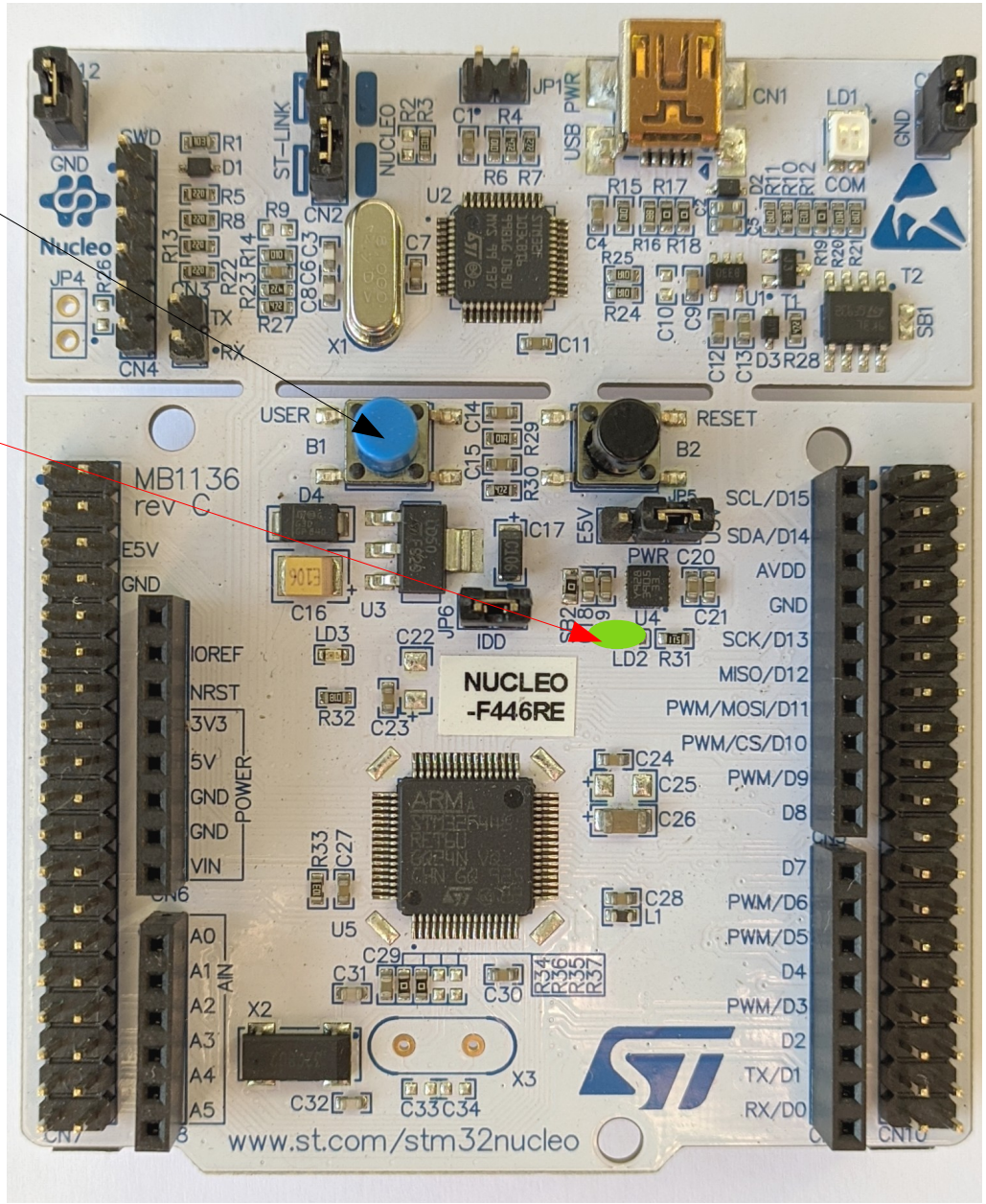
```c
#include "main.h"

void SystemClock_Config(void);
static void MX_GPIO_Init(void);

/* Private user code ------------ */
/* USER CODE BEGIN 0 */
void EXTI15_10_IRQHandler(void) {
  __HAL_GPIO_EXTI_CLEAR_IT(B1_Pin);                 // Clear pending request
  HAL_GPIO_TogglePin(LED_GPIO_Port, LED_Pin);       // Toggle LED state
}
/* USER CODE END 0 */

int main(void) {
  HAL_Init();
  SystemClock_Config();
  MX_GPIO_Init();
  /* USER CODE BEGIN 2 */
  HAL_NVIC_EnableIRQ(EXTI15_10_IRQn);  // EXTI13 interrupt enable
  /* USER CODE END 2 */;
  while (1);
}
```

Az **EXTI13** megszakítás kiszolgálása:
- megszakításjelző törlése
- LED kimenet átbillentése

Az **EXTI13** megszakítás engedélyezése

Lab02/F446RE_b1int projekt futtatás

- A kék nyomógomb minden lenyomásakor az **LD2** LED állapotot vált

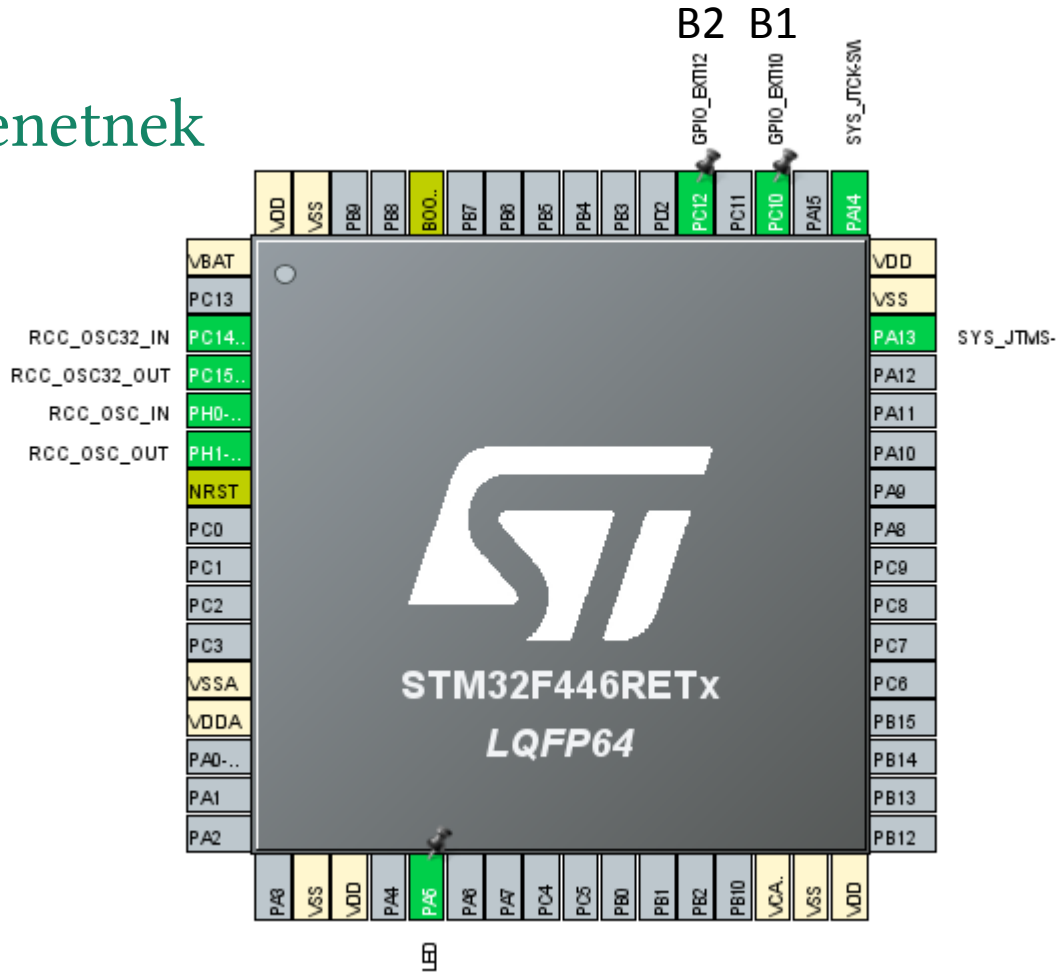


A Lab02/F446RE_b2int projekt

- A **NUCLEO-F446RE** kártya beépített **LD2** LED-jét (**PA5**) kapcsolgatjuk ki-be, két nyomógombbal, megszakításban. A **B1** (**PC10**) nyomógomb bekapcsolja, a **B2** (**PC12**) nyomógomb kikapcsolja (a gomboknak itt **közös megszakítási vektora** van)
- Konfiguráljuk push-pull digitális kimenetként a **PA5** portbitet!
- Konfiguráljuk a **PC10 és PC12** bemenetet `GPIO_MODE_IT_FALLING` módba, így minden lefutó élre (lenyomás) megszakítást generálnak
- Engedélyezzük az `EXTI15_10_IRQn` megszakítási vektort
- Definiáljuk a `void EXTI15_10_IRQHandler()` függvényt, ami az `EXTI15_10_IRQn` megszakítási vektor kiszolgáló függvénye, s ebben hívjuk meg mindkét bemenetre a `HAL_GPIO_EXTI_IRQHandler` függvényt
- Definiáljuk a `HAL_GPIO_EXTI_Callback` függvényt, ami a fenti a `HAL_GPIO_EXTI_IRQHandler` megszakításkezelő visszahívási függvénye, s ebben végezzük el a LED ki- vagy bekapcsolását

A GPIO kivezetések konfigurálása

- A **PA5** kivezetést a szokásos módon **GPIO_Output** módba állítjuk
- Opciók: **push-pull** kimeneti mód választás, alaphelyzetben **Low** állapotban legyen, alacsony frekvencia, és a **LED** nevet kapta
- A **PC10 és PC12** kivezetéseket külső megszakítást fogadó bemenetnek állítjuk (**GPIO_EXTIxx** mód)
- Opciók: lefutó élre érzékeny bemenet, belső felhúzás (pull_up), a bemenetekhez a **B1** és **B2** nevet rendeltük



F446RE_b2int projekt : GPIO konfigurálás

```
static void MX_GPIO_Init(void) {
  GPIO_InitTypeDef GPIO_InitStruct = {0};

  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOC_CLK_ENABLE();
  __HAL_RCC_GPIOH_CLK_ENABLE();
  __HAL_RCC_GPIOA_CLK_ENABLE();
  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, GPIO_PIN_RESET);

  /*Configure GPIO pin : LED_Pin */
  GPIO_InitStruct.Pin = LED_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(LED_GPIO_Port, &GPIO_InitStruct);

  /*Configure GPIO pins : B1_Pin B2_Pin */
  GPIO_InitStruct.Pin = B1_Pin|B2_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_IT_FALLING;
  GPIO_InitStruct.Pull = GPIO_PULLUP;
  HAL_GPIO_Init(GPIOC, &GPIO_InitStruct);
}
```

A GPIO portok órajelének engedélyezése

Az MX_GPIO_Init függvényt az STM32Cube MX eszköz generálta, az előző oldalon leírt beállítások hatására. Változtatás nélkül használjuk.

F446RE_b2int projekt main.c részlet

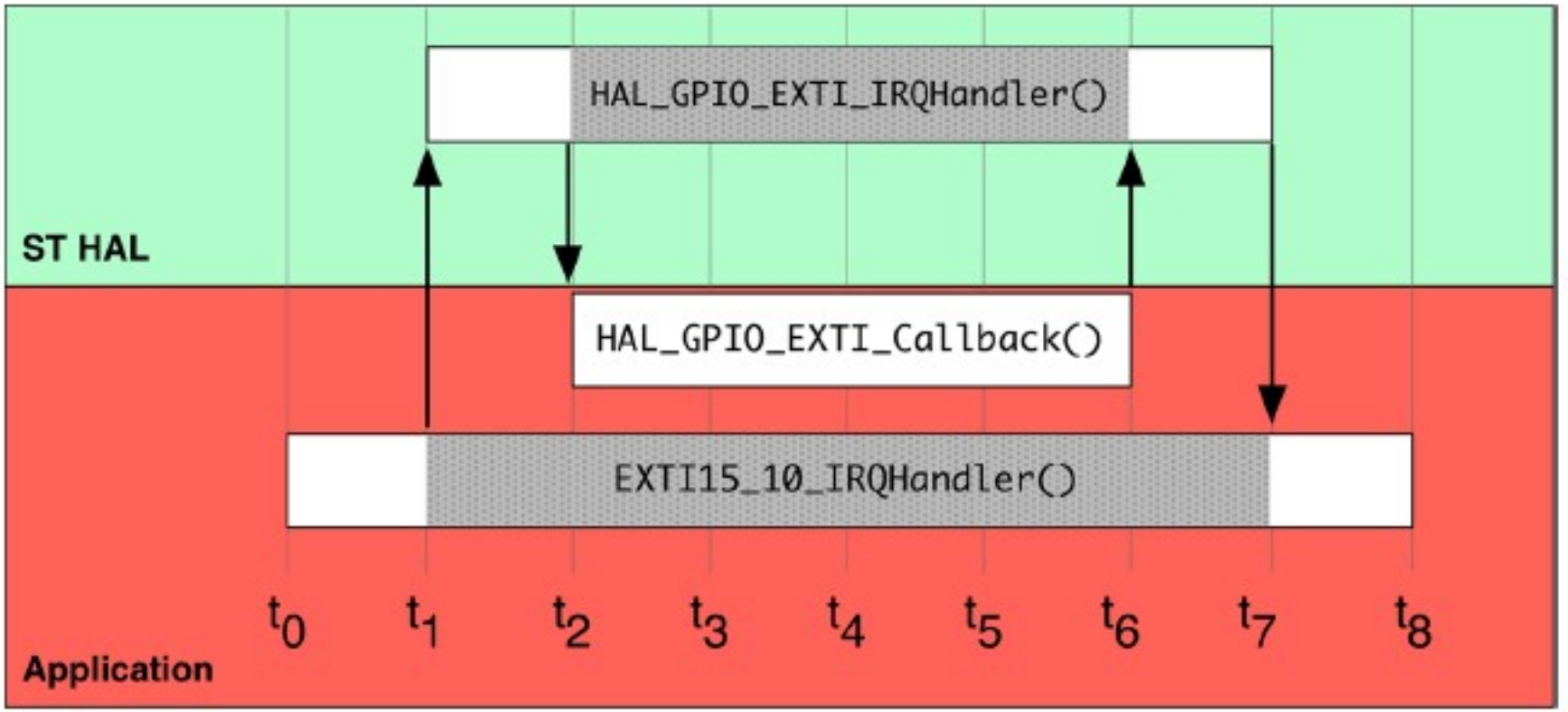
```c
#include "main.h"
void SystemClock_Config(void);
static void MX_GPIO_Init(void);

void EXTI15_10_IRQHandler(void) {
  HAL_GPIO_EXTI_IRQHandler(B1_Pin);
  HAL_GPIO_EXTI_IRQHandler(B2_Pin);
}

void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin) {
  if(GPIO_Pin == B1_Pin)
    HAL_GPIO_WritePin(LED_GPIO_Port,LED_Pin,SET);
  else
    HAL_GPIO_WritePin(LED_GPIO_Port,LED_Pin,RESET);
}

int main(void) {
  HAL_Init();
  SystemClock_Config();
  MX_GPIO_Init();
  HAL_NVIC_EnableIRQ(EXTI15_10_IRQn);  // EXTI13 interrupt enable
  while (1);
}
```

Az **EXTI10/EXTI12** megszakítás kiszolgálása a beépített EXTI kiszolgáló fv. meghívásával, amely a jelzőbitet is törli

Az **EXTI** visszahívási függvénye, amelyben a LED vezérlését elvégezzük

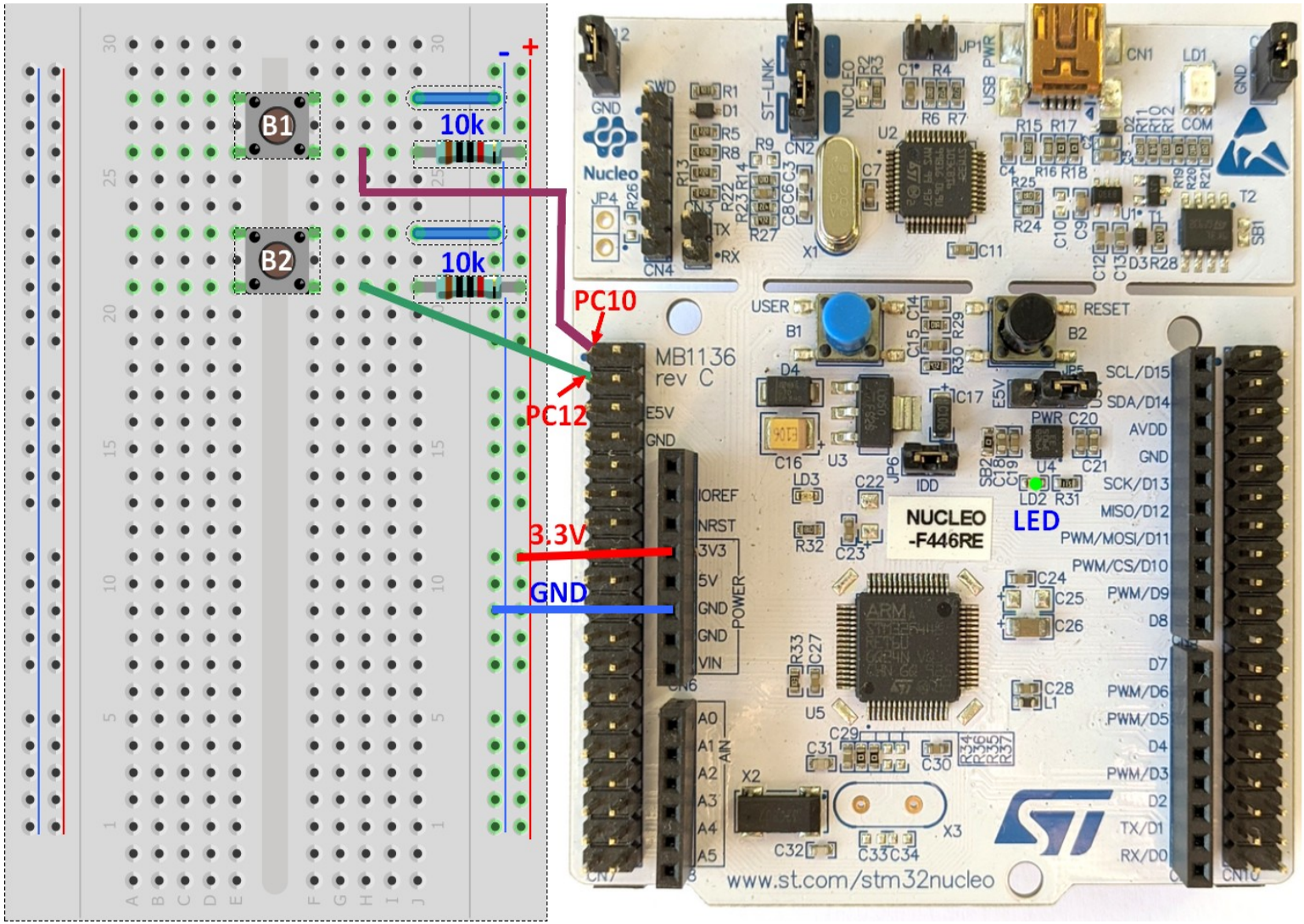
HAL_GPIO_EXTI_IRQHandler()

- A HAL programkönyvtárban definiált `HAL_GPIO_EXTI_IRQHandler` függvény ellenőrzi, hogy a paraméterként megadott sorszámú biten van-e függőben megszakítási kérelem, s ha igen, akkor törli a kérelmet, majd meghívja az alkalmazási rétegben általunk definiált `HAL_GPIO_EXTI_Callback` visszahívási függvényt



Lab02/F446RE_b2int projekt

- Ebben a projektben a beépített **LD2** LED-et (**PA5**) kapcsolgatjuk, két nyomógombbal
- A **B1** (PC10) nyomógomb bekapcsolja
- A **B2** (PC12) nyomógomb kikapcsolja

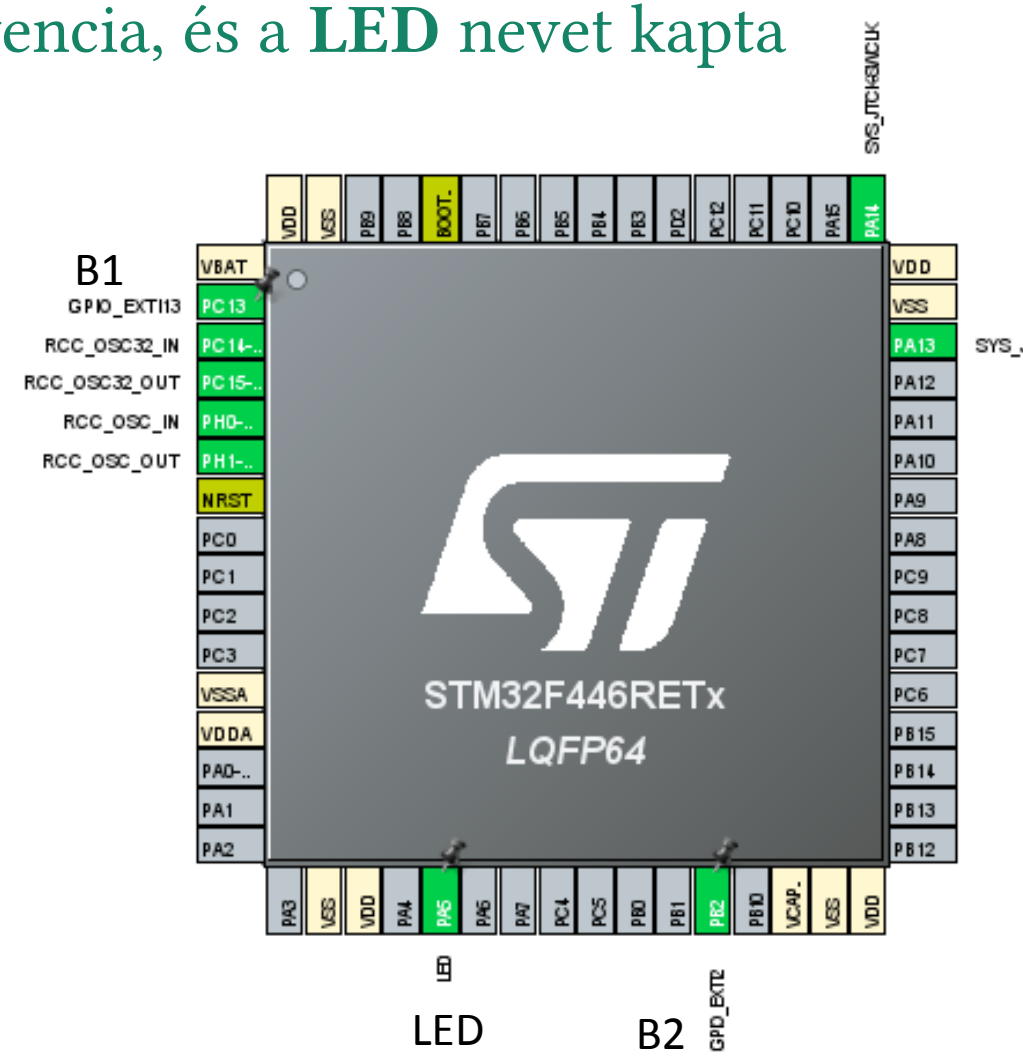


A Lab02/F446RE_b3int projekt

- A **NUCLEO-F446RE** kártya beépített **LD2** LED (**PA5**) villogását kapcsolgatjuk ki-be, két nyomógombbal, prioritásos megszakítással.
  A **B1** (**PC13**) nyomógomb bekapcsolja, a **B2** (**PB2**) nyomógomb kikapcsolja a villogtatást (a gomboknak **saját megszakítási vektora** van)
- Konfiguráljuk push-pull digitális kimenetként a **PA5** portbitet!
- Konfiguráljuk a **PC13 és PB2** bemenetet `GPIO_MODE_IT_FALLING` módba, így minden lefutó élre (lenyomás) megszakítást generálnak
- Engedélyezzük az `EXTI15_10_IRQn` megszakítási vektort 1-es prioritással és az `EXTI2_IRQn` megszakítási vektort 0-ás prioritással
- Definiáljuk az `EXTI2_IRQHandler()` és az `EXTI15_10_IRQHandler()` függvényt, a megszakítási vektorok kiszolgálására, s ezekben hívjuk meg az érintett bemenetre a `HAL_GPIO_EXTI_IRQHandler` függvényt
- Definiáljuk a `HAL_GPIO_EXTI_Callback` függvényt, ami a fenti a `HAL_GPIO_EXTI_IRQHandler` megszakításkezelő visszahívási függvénye, s ebben végezzük el a ***blink*** állapotjelző változó ki- vagy bekapcsolását
- A LED villogatását a főprogramban végezzük, ha ***blink*** értéke nem nulla

A GPIO kivezetések konfigurálása

- A **PA5** kivezetést a szokásos módon **GPIO_Output** módba állítjuk
- Opciók: **push-pull** kimeneti mód választás, alaphelyzetben **Low** állapotban legyen, alacsony frekvencia, és a **LED** nevet kapta
- A **PC13 és PB2** kivezetéseket külső megszakítást fogadó bemenetnek állítjuk (**GPIO_EXTIxx** mód)
- Opciók: lefutó élre érzékeny bemenet, belső felhúzás (pull_up), a bemenetekhez a **B1** és **B2** nevet rendeltük



F446RE_b3int projekt : GPIO konfigurálás

```
static void MX_GPIO_Init(void) {
  GPIO_InitTypeDef GPIO_InitStruct = {0};
  __HAL_RCC_GPIOC_CLK_ENABLE();
  __HAL_RCC_GPIOH_CLK_ENABLE();
  __HAL_RCC_GPIOA_CLK_ENABLE();
  __HAL_RCC_GPIOB_CLK_ENABLE();
  HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, GPIO_PIN_RESET);

  /*Configure GPIO pin : B1_Pin (PC13) */
  GPIO_InitStruct.Pin = B1_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_IT_RISING;
  GPIO_InitStruct.Pull = GPIO_PULLUP;
  HAL_GPIO_Init(B1_GPIO_Port, &GPIO_InitStruct);

  /*Configure GPIO pin : LED_Pin (PA5) */
  GPIO_InitStruct.Pin = LED_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(LED_GPIO_Port, &GPIO_InitStruct);

  /*Configure GPIO pin : B2_Pin (PB2) */
  GPIO_InitStruct.Pin = B2_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_IT_RISING;
  GPIO_InitStruct.Pull = GPIO_PULLUP;
  HAL_GPIO_Init(B2_GPIO_Port, &GPIO_InitStruct);
}
```

A GPIO portok órajelének engedélyezése

Az MX_GPIO_Init függvényt az STM32Cube MX eszköz generálta, az előző oldalon leírt beállítások hatására. Változtatás nélkül használjuk.

F446RE_b3int projekt main.c részlet

```
#include "main.h"
/* USER CODE BEGIN PV */
volatile uint8_t blink = 0;
/* USER CODE END PV */

void SystemClock_Config(void);
static void MX_GPIO_Init(void);

/* USER CODE BEGIN 0 */
void EXTI15_10_IRQHandler(void) {
  HAL_GPIO_EXTI_IRQHandler(B1_Pin);
}

void EXTI2_IRQHandler(void) {
  HAL_GPIO_EXTI_IRQHandler(B2_Pin);
}

void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin) {
  if(GPIO_Pin == B1_Pin)
    blink = 1;
  else
    blink = 0;
}
/* USER CODE END 0 */
```

Az **EXTI13 és az EXTI2** megszakításokat kiszolgáló függvények a HAL-ban definiált közös fv-t hívják meg, paraméterezve

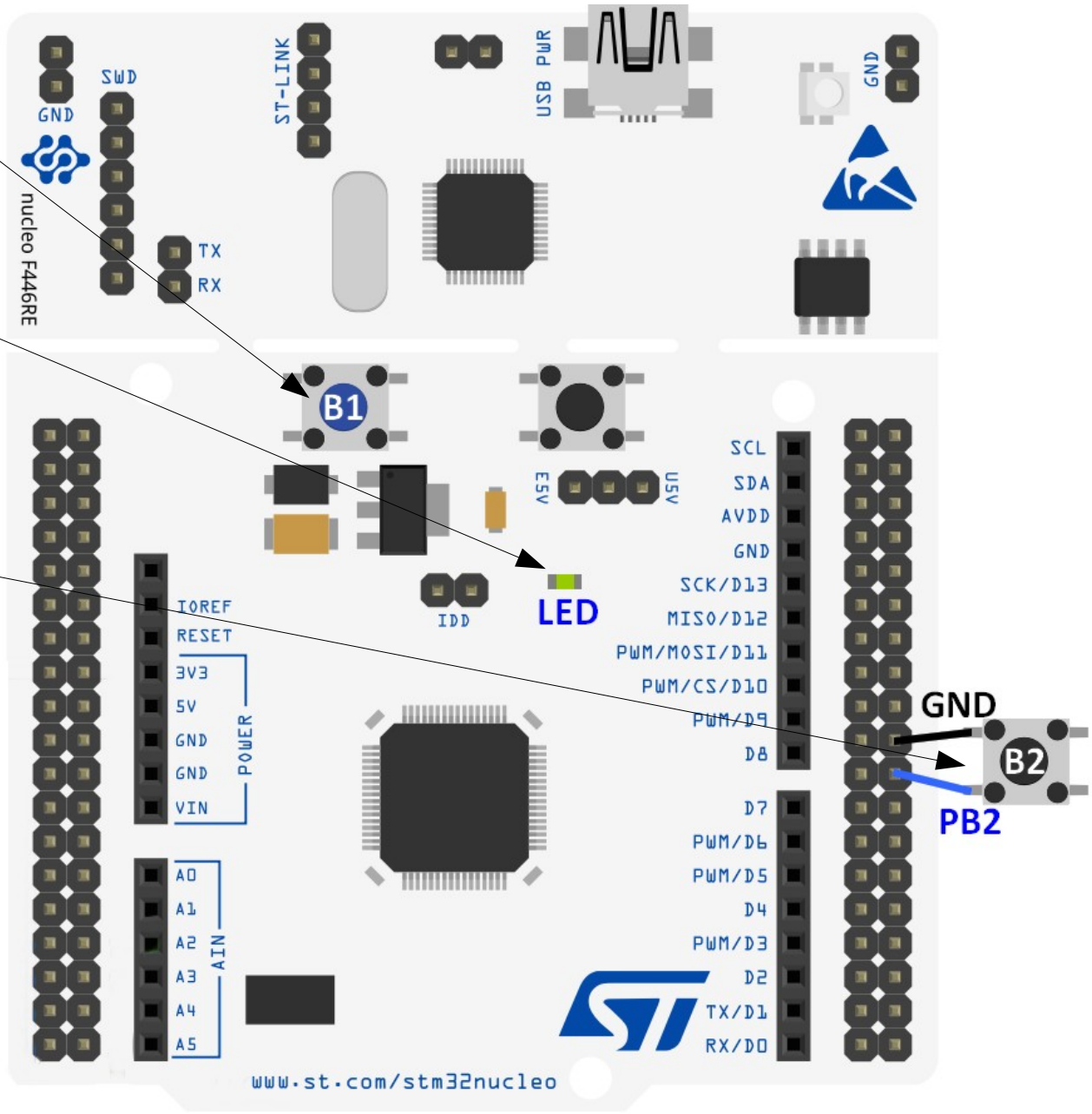
Az **EXTI** visszahívási függvénye, amelyben a LED villogtatás állapotvezérlését végezzük

F446RE_b3int projekt main.c részlet

```
int main(void) {
  HAL_Init();
  SystemClock_Config();
  MX_GPIO_Init();

  /* USER CODE BEGIN 2 */
  HAL_NVIC_SetPriority(EXTI15_10_IRQn, 0x1, 0);
  HAL_NVIC_EnableIRQ(EXTI15_10_IRQn);

  HAL_NVIC_SetPriority(EXTI2_IRQn, 0x0, 0);
  HAL_NVIC_EnableIRQ(EXTI2_IRQn);
  /* USER CODE END 2 */

  while (1) {
      if(blink)
          HAL_GPIO_TogglePin(LED_GPIO_Port, LED_Pin);
      else
          HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, RESET);
      HAL_Delay(200);
  }
}
```

Lab02/F446RE_b3int projekt

- A **B1** nyomógomb lenyomásakor a LED villogni kezd (állapotváltás 200 ms-onként)
- A **B1** nyomógomb lenyomásakor a LED kialszik

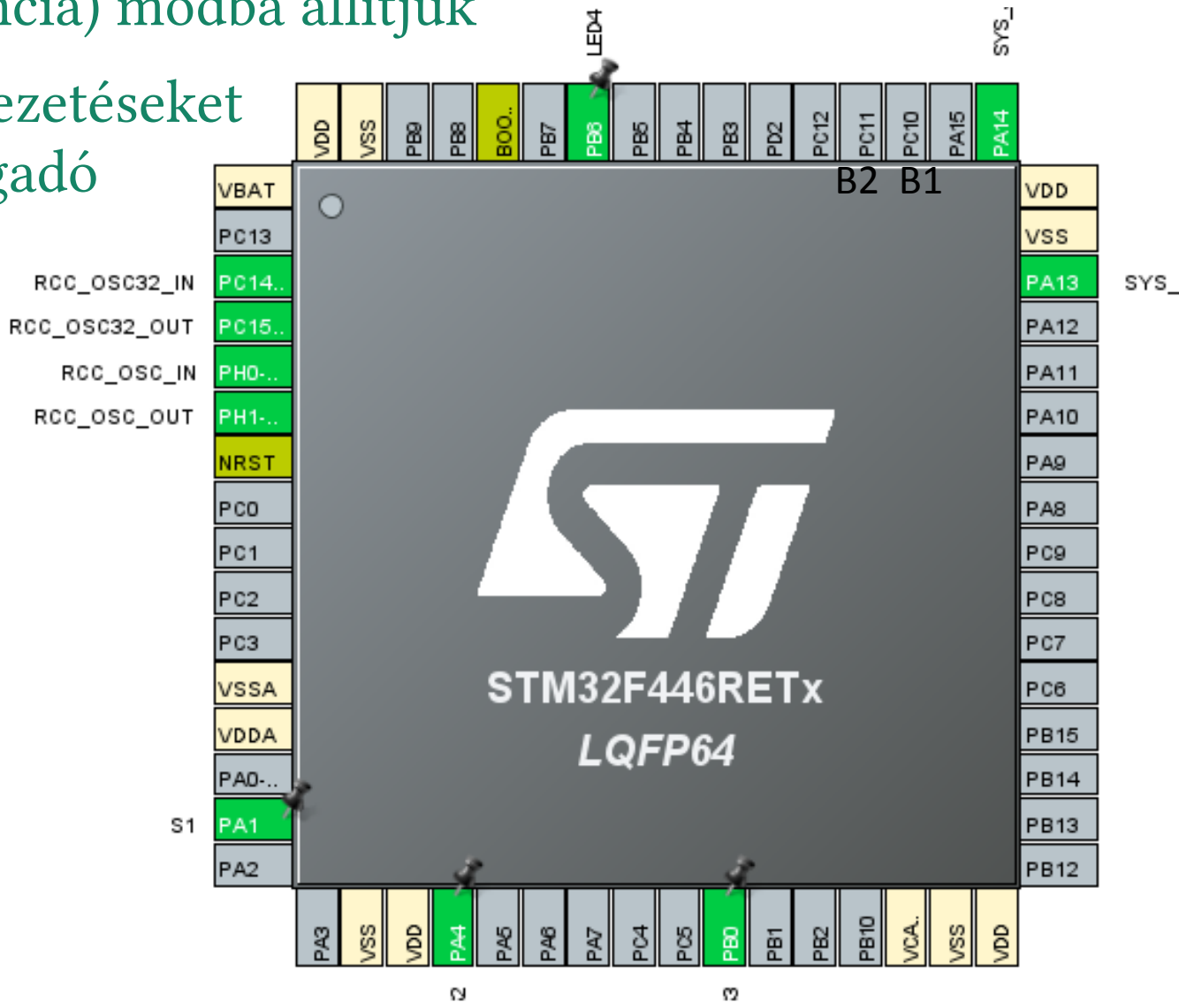


A Lab02/F446RE_b4int projekt

- A **NUCLEO-F446RE** kártyára ültetett multifunkciós bővítőkártya **LED4** kijelzőjét, és nyomógombjait használjuk fel. Az **S1** gomb bekapcsolja, **S2** kikapcsolja, **S3** pedig villogtatja **LED4**-et.
- Hardver bekötés:
  - **S1** (LED bekapcsoló gomb) - **PA1** és GND közé, külső felhúzással
  - **S2** (LED kikapcsoló gomb) - **PA4** és GND közé, külső felhúzással
  - **S3** (LED villogtató gomb) - **PB0** és GND közé, külső felhúzással
  - **LED4** (beépített LED) - **PB6** vezérli a LED katódját
- A nyomógombokat megszakításban vezéreljük, a megszakítások kiszolgáló függvényeit az előző programhoz hasonlóan definiáljuk (`EXTI0_IRQHandler, EXTI1_IRQHandler, EXTI4_IRQHandler`)
- A `HAL_GPIO_EXTI_Callback` visszahívási függvényben a ***ledstate*** változó 0, 1, vagy 2 értéket vehet fel (0: ki, 1: be, 2: villog)

A GPIO kivezetések konfigurálása

- A **PB6** kivezetést **GPIO_Output** (Push-Pull, alaphelyzetben High szint, alacsony frekvencia) módba állítjuk
- A **PA1, PA4, PB0** kivezetéseket külső megszakítást fogadó bemenetnek állítjuk (**GPIO_EXTIxx** mód)
- Opciók: lefutó élre érzékeny bemenet, belső felhúzás nélkül, a bemenetekhez az a **S1, S2** és **S3** nevet rendeltük



F446RE_b4int main.c: GPIO konfigurálás

```
static void MX_GPIO_Init(void) {
  GPIO_InitTypeDef GPIO_InitStruct = {0};
  __HAL_RCC_GPIOC_CLK_ENABLE();                    /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOH_CLK_ENABLE();
  __HAL_RCC_GPIOA_CLK_ENABLE();
  __HAL_RCC_GPIOB_CLK_ENABLE();
  HAL_GPIO_WritePin(LED4_GPIO_Port, LED4_Pin, GPIO_PIN_SET);

  /*Configure PA1, PA4 pins : S1_pin, S2_pin */
  GPIO_InitStruct.Pin = S1_Pin|S2_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_IT_FALLING;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);

  /*Configure PB0 pin : S3_pin */
  GPIO_InitStruct.Pin = S3_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_IT_FALLING;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  HAL_GPIO_Init(S3_GPIO_Port, &GPIO_InitStruct);

  /*Configure PB6 pin : LED4_Pin */
  GPIO_InitStruct.Pin = LED4_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(LED4_GPIO_Port, &GPIO_InitStruct);
}
```

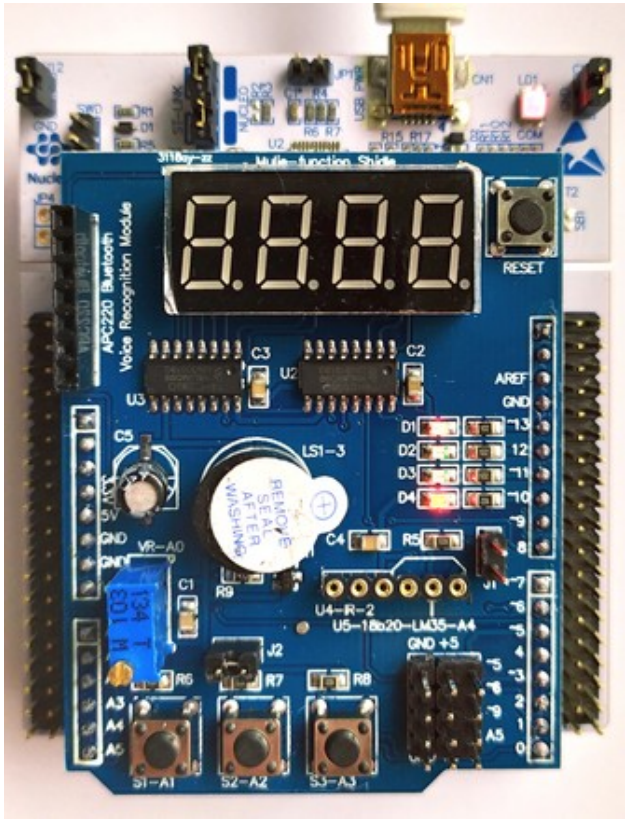
Az MX_GPIO_Init függvényt az STM32Cube MX eszköz generálta, az előző oldalon leírt beállítások hatására. Változtatás nélkül használjuk.

F446RE_b4int projekt main.c részlet

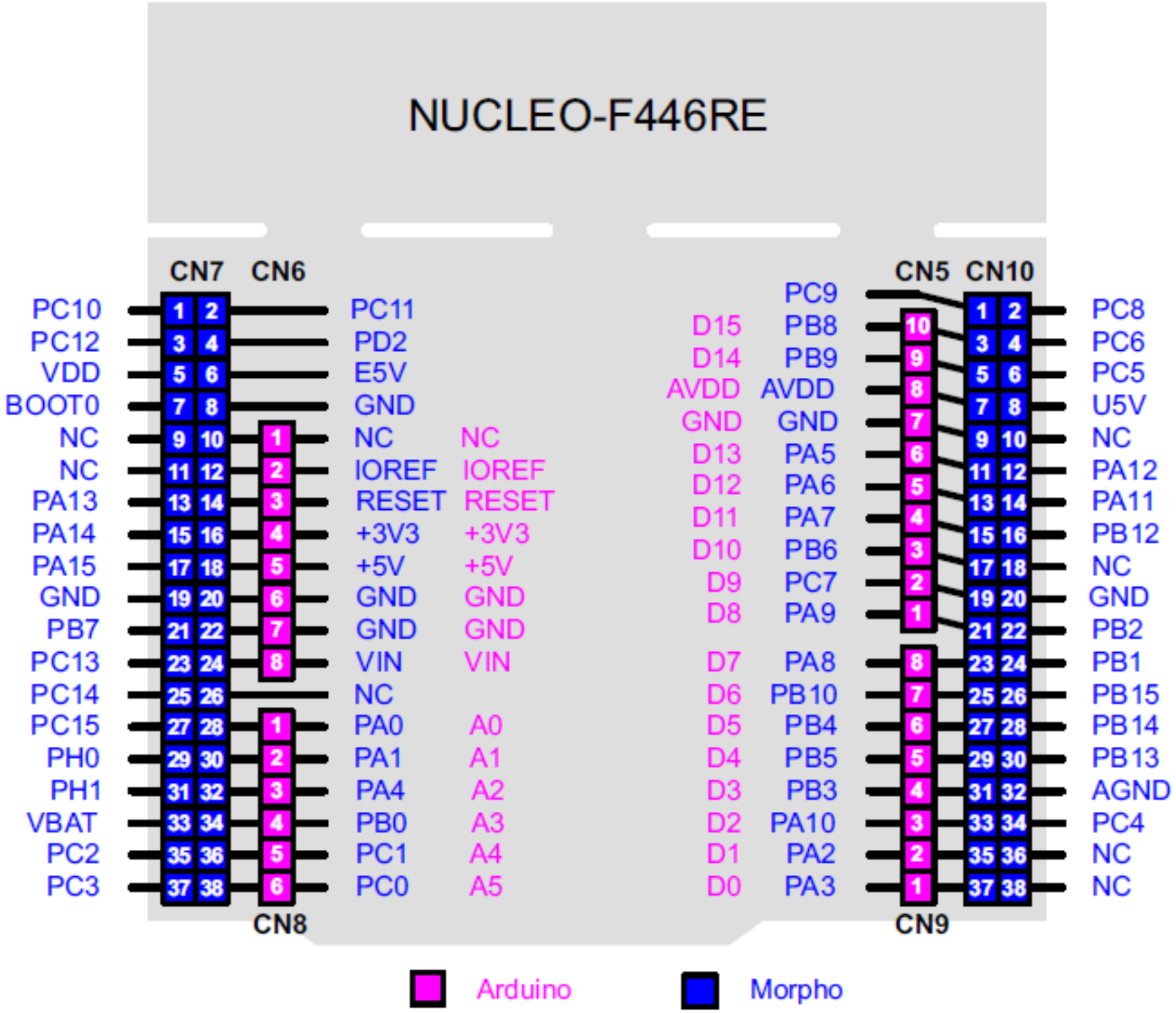
```c
#include "main.h"
volatile uint8_t ledstate = 0;  // 0: off, 1: on, 2: blinking

void SystemClock_Config(void);
static void MX_GPIO_Init(void);

void EXTI0_IRQHandler(void) {
  HAL_GPIO_EXTI_IRQHandler(S3_Pin);
}

void EXTI1_IRQHandler(void) {
  HAL_GPIO_EXTI_IRQHandler(S1_Pin);
}

void EXTI4_IRQHandler(void) {
  HAL_GPIO_EXTI_IRQHandler(S2_Pin);
}

void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin) {
  if(GPIO_Pin == S3_Pin)           // S3 (PB0)
    ledstate = 2;                  // LED blinking
  else if(GPIO_Pin == S1_Pin)      // S1 (PA1)
    ledstate = 1;                  // LED ON
  else                             // S2 (PA4) or other...
    ledstate = 0;                  // LED OFF
}
```

F446RE_b4int projekt main.c részlet

```
int main(void) {
  HAL_Init();
  SystemClock_Config();
  MX_GPIO_Init();

  // S1 - GPIO A1
  HAL_NVIC_SetPriority(EXTI1_IRQn, 0x1, 0);
  HAL_NVIC_EnableIRQ(EXTI1_IRQn);
  // S2 - GPIO A4
  HAL_NVIC_SetPriority(EXTI4_IRQn, 0x0, 0);
  HAL_NVIC_EnableIRQ(EXTI4_IRQn);
  // S3 - GPIO B0
  HAL_NVIC_SetPriority(EXTI0_IRQn, 0x2, 0);
  HAL_NVIC_EnableIRQ(EXTI0_IRQn);

  while (1)
  {
      if (ledstate == 2)
          HAL_GPIO_TogglePin(LED4_GPIO_Port, LED4_Pin);
      else if (ledstate == 1)
          HAL_GPIO_WritePin(LED4_GPIO_Port, LED4_Pin, RESET);
      else
          HAL_GPIO_WritePin(LED4_GPIO_Port, LED4_Pin, SET);
      HAL_Delay(200);
  }
}
```

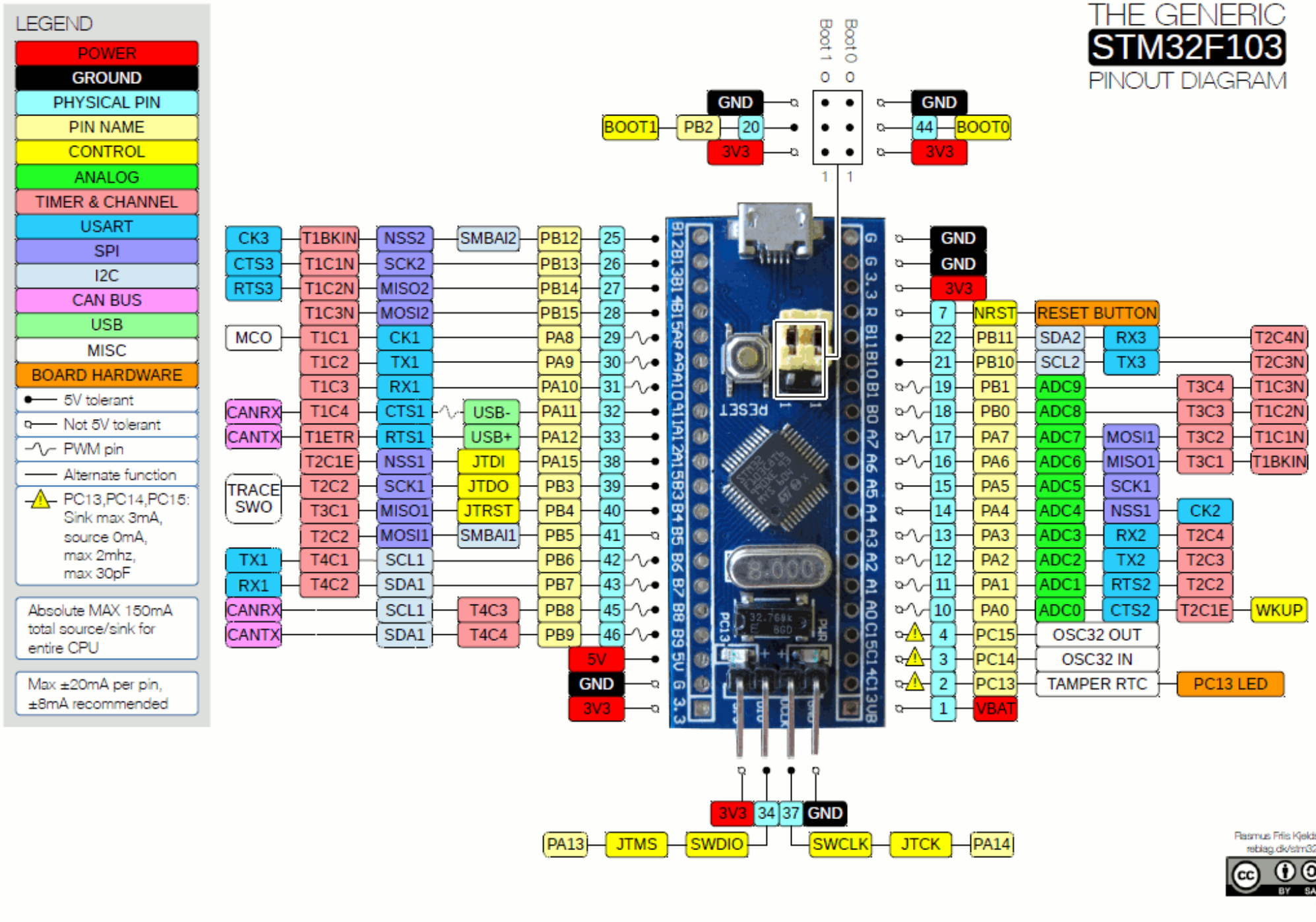


A LED lehúzáskor világít!


NUCLEO-F446RE
CN7
CN6
PC10
PC12
VDD
BOOT0
NC
NC
PA13
PA14
PA15
GND
PB7
PC13
PC14
PC15
PH0
PH1
VBAT
PC2
PC3
PC11
PD2
E5V
GND
NC
IOREF
RESET
+3V3
+5V
GND
GND
VIN
NC
PA0
PA1
PA4
PB0
PC1
PC0
NC
IOREF
RESET
+3V3
+5V
GND
GND
VIN
A0
A1
A2
A3
A4
A5
CN8
CN5
CN10
PC9
D15
D14
AVDD
GND
D13
D12
D11
D10
D9
D8
D7
D6
D5
D4
D3
D2
D1
D0
PB8
PB9
AVDD
GND
PA5
PA6
PA7
PB6
PC7
PA9
PA8
PB10
PB4
PB5
PB3
PA10
PA2
PA3
PC8
PC6
PC5
U5V
NC
PA12
PA11
PB12
NC
GND
PB2
PB1
PB15
PB14
PB13
AGND
PC4
NC
NC
CN9
Arduino
Morpho

Lab02 projektek STM32F103C8-ra

- Az **STM32F103C8** mikrovezérlőre (Blue pill) is adaptáltuk az előadás mintapéldáit. Itt most csak a hardver eltéréseket soroljuk fel. Ne feledjük, hogy a **PC13** kivezetésre kötött beépített LED ezen a kártyán lehúzáskor világít (OD módba konfiguráltuk)!
- **Lab02/F103_b1int:** a beépített LED ki/bekapcsolása egy nyomógombbal, programmegszakítással. **LED**: **PC13, B1** gomb: **PB10**
- **Lab02/F103_b2int:** a beépített LED ki/bekapcsolása két nyomógombbal, programmegszakítással **LED**: **PC13, B1**: **PB10, B2: PB12**
- **Lab02/F103_b3int:** a beépített LED villogtatás ki/bekapcsolása két nyomógombbal, programmegszakítással **LED** katód: **PC13, B1** gomb: **PB10, B2** gomb: **PB0**
- **Lab02/F103_b4int:** a beépített LED vezérlése három nyomógombbal, programmegszakítással **LED** katód: **PC13, S1** gomb: **PB10, S2** gomb: **PB12, S3** gomb: **PB0**


LEGEND
POWER
GROUND
PHYSICAL PIN
PIN NAME
CONTROL
ANALOG
TIMER & CHANNEL
USART
SPI
I2C
CAN BUS
USB
MISC
BOARD HARDWARE
5V tolerant
Not 5V tolerant
PWM pin
Alternate function
PC13,PC14,PC15: Sink max 3mA, source 0mA, max 2mhz, max 30pF
Absolute MAX 150mA total source/sink for entire CPU
Max ±20mA per pin, ±8mA recommended
THE GENERIC
STM32F103
PINOUT DIAGRAM
Boot 1
Boot 0
GND
BOOT1
PB2
20
3V3
GND
44
BOOT0
3V3
CK3
CTS3
RTS3
MCO
CANRX
CANTX
TRACE SWO
TX1
RX1
CANRX
CANTX
T1BKIN
T1C1N
T1C2N
T1C3N
T1C1
T1C2
T1C3
T1C4
T1ETR
T2C1E
T2C2
T3C1
T2C2
T4C1
T4C2
NSS2
SCK2
MISO2
MOSI2
CK1
TX1
RX1
CTS1
RTS1
NSS1
SCK1
MISO1
MOSI1
SCL1
SDA1
SCL1
SDA1
SMBAI2
USB-
USB+
JTDI
JTDO
JTRST
SMBAI1
T4C3
T4C4
PB12
PB13
PB14
PB15
PA8
PA9
PA10
PA11
PA12
PA15
PB3
PB4
PB5
PB6
PB7
PB8
PB9
25
26
27
28
29
30
31
32
33
38
39
40
41
42
43
45
46
5V
GND
3V3
GND
GND
3V3
7
22
21
19
18
17
16
15
14
13
12
11
10
4
3
2
1
NRST
PB11
PB10
PB1
PB0
PA7
PA6
PA5
PA4
PA3
PA2
PA1
PA0
PC15
PC14
PC13
VBAT
RESET BUTTON
SDA2
SCL2
ADC9
ADC8
ADC7
ADC6
ADC5
ADC4
ADC3
ADC2
ADC1
ADC0
OSC32 OUT
OSC32 IN
TAMPER RTC
RX3
TX3
MOSI1
MISO1
SCK1
NSS1
RX2
TX2
RTS2
CTS2
T3C4
T3C3
T3C2
T3C1
CK2
T2C4
T2C3
T2C2
T2C1E
T2C4N
T2C3N
T1C3N
T1C2N
T1C1N
T1BKIN
WKUP
PC13 LED
3V3
34
37
GND
PA13
JTMS
SWDIO
SWCLK
JTCK
PA14
Rasmus Fris Kjeldsen
reblag.dk/stm32
V1.0