VILNIAUS GEDIMINO TECHNIKOS UNIVERSITETAS

ELEKTRONIKOS FAKULTETAS

ELEKTRONINIŲ SISTEMŲ KATEDRA

# LABORATORINIS DARBAS 4

Įterptinių sistemų inžinerija

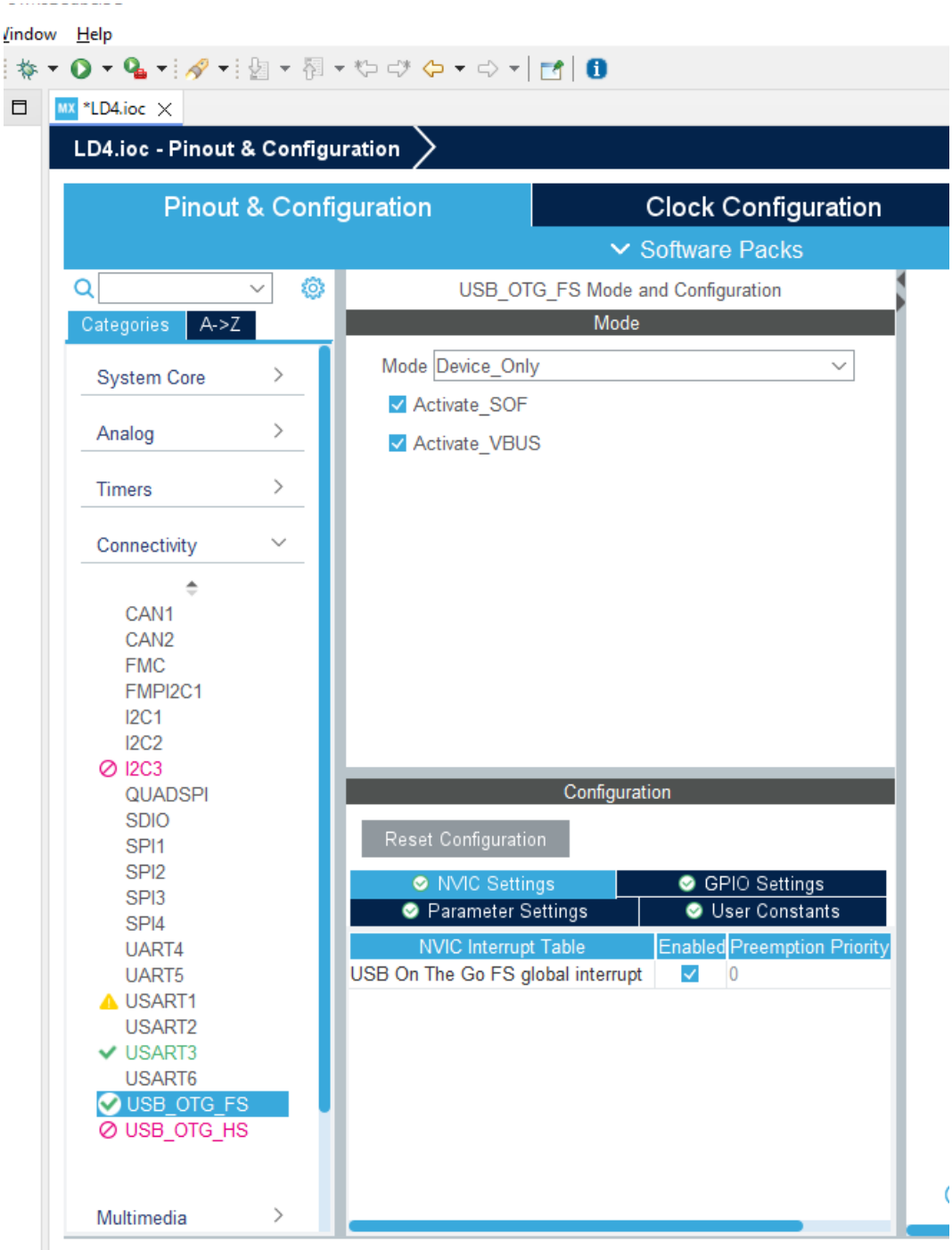
Atliko: EKSfm-24 gr. Ignas Malinauskas

Tikrino: dr. Eldar Šabanovič

VILNIUS 2025

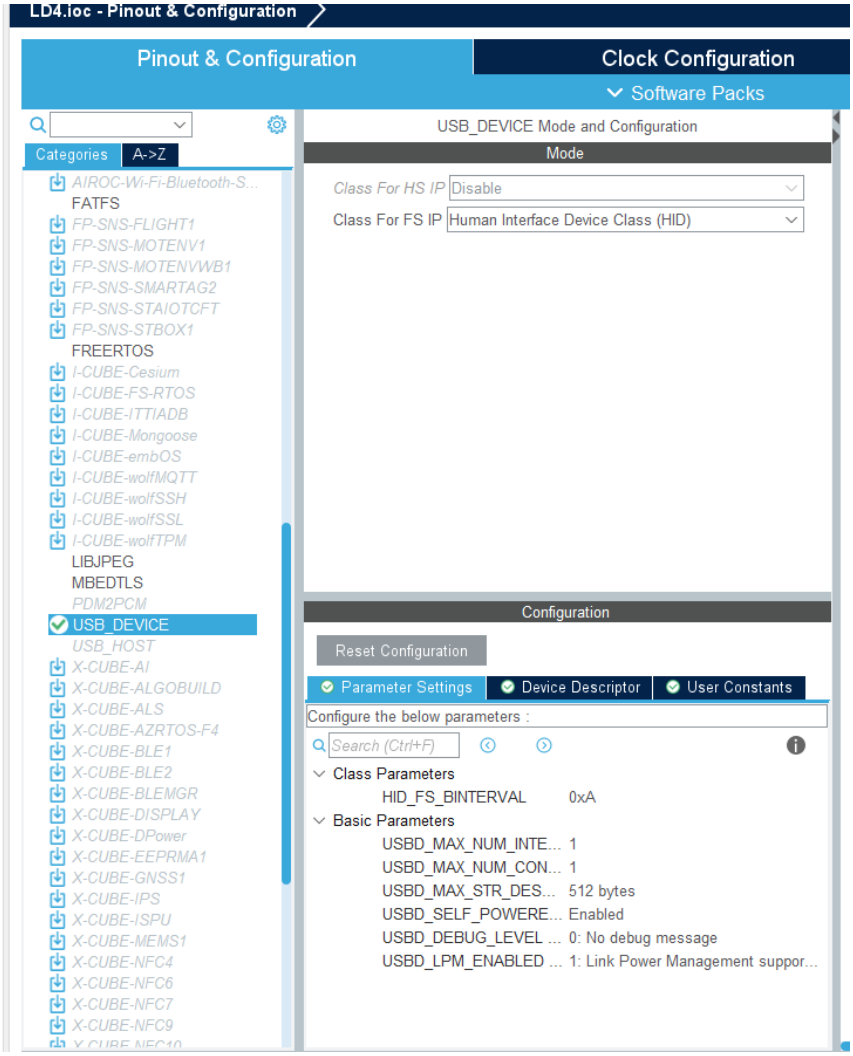
# 1. USB On-the-go nustatymai

Įjungiamas STM32F4 plokštės USB On-The-Go Full Speed parinktis, kad plokštė veiktų kaip įrenginys ir būtų sukonfigūruotos išvestys, įjungiami Start of Frame ir VBUS nustatymai bei globali pertrauktis, kad komunikacija su kompiuteriu būtų įmanoma.

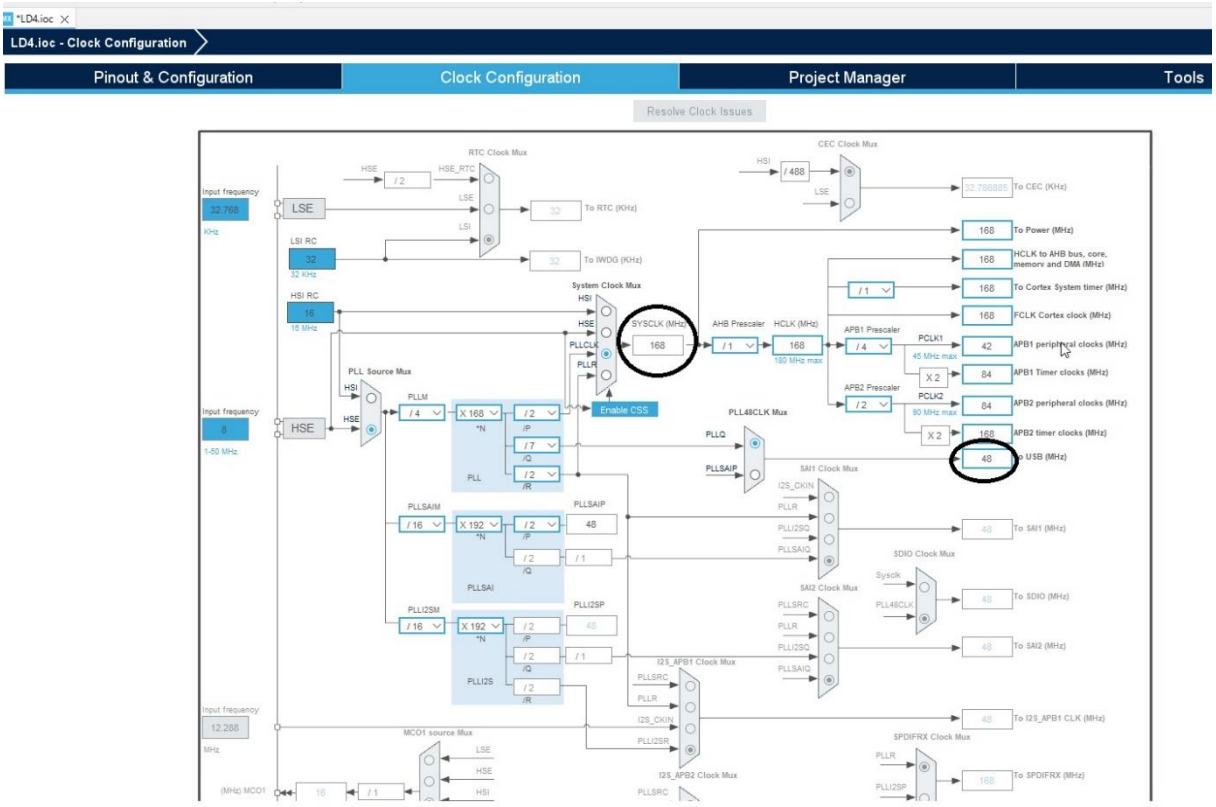


## 2. USB protokolo nustatymai

HID – kompiuteris matys STM32F4 plokštę kaip išorinį įrenginį (pėlytė, klaviatūra), STM32CubeMX sugeneruos HID įrenginio deskriptorių.



## 3. Laikrodžio nustatymai




## 4. HID pėlytės programos kodas

Paspaudus mygtuką pėlytė truputį pajuda žemyn.

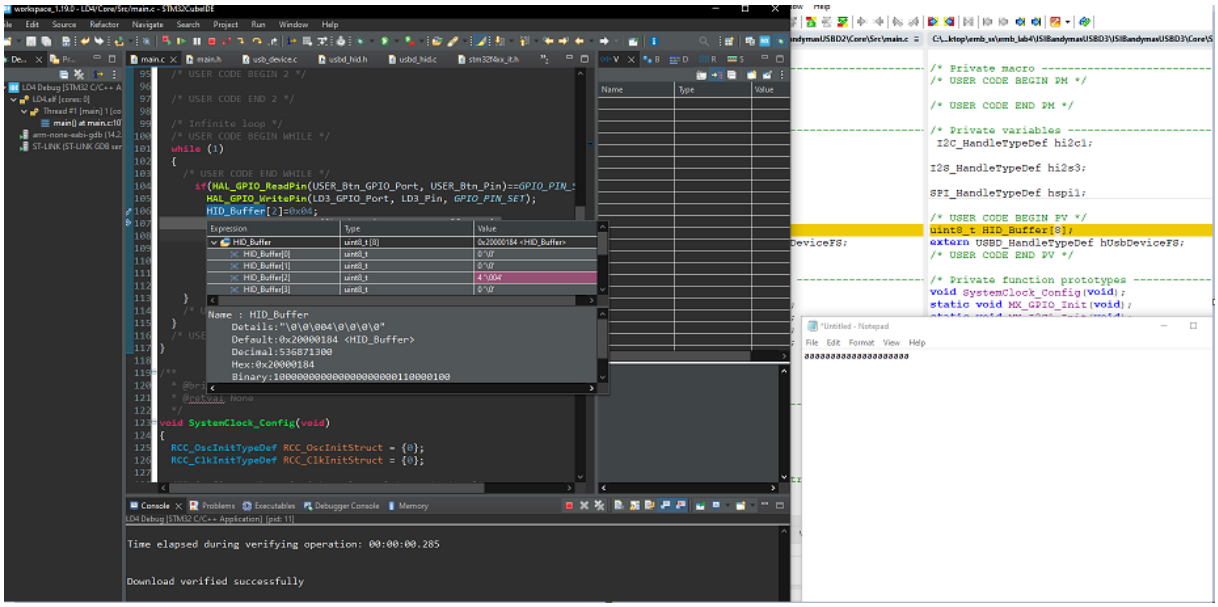
```
UART_HandleTypeDef huart3;

/* USER CODE BEGIN PV */
uint8_t HID_Buffer[4];
extern USBD_HandleTypeDef hUsbDeviceFS;
```

```
while (1)
{
  /* USER CODE END WHILE */
    if(HAL_GPIO_ReadPin(USER_Btn_GPIO_Port, USER_Btn_Pin)==GPIO_PIN_SET){
      HAL_GPIO_WritePin(LD3_GPIO_Port, LD3_Pin, GPIO_PIN_SET);
      HID_Buffer[2]=0x04;
      USBD_HID_SendReport(&hUsbDeviceFS, HID_Buffer, 4);
      HAL_Delay(20);

      HAL_GPIO_WritePin(LD3_GPIO_Port, LD3_Pin, GPIO_PIN_RESET);
  }
  /* USER CODE BEGIN 3 */
}
/* USER CODE END 3 */
```

## 5. HID klaviatūros programos kodas

```
UART_HandleTypeDef huart3;

/* USER CODE BEGIN PV */
uint8_t HID_Buffer[8];
extern USBD_HandleTypeDef hUsbDeviceFS;
/* USER CODE END PV */
```

```
  while (1)
  {
    /* USER CODE END WHILE */
      if(HAL_GPIO_ReadPin(USER_Btn_GPIO_Port, USER_Btn_Pin)==GPIO_PIN_SET){
        HAL_GPIO_WritePin(LD3_GPIO_Port, LD3_Pin, GPIO_PIN_SET);
        HID_Buffer[2]=0x04;
        USBD_HID_SendReport(&hUsbDeviceFS, HID_Buffer, 8);
        HAL_Delay(20a);
        HID_Buffer[2]=0;
        USBD_HID_SendReport(&hUsbDeviceFS, HID_Buffer, 8);

        HAL_GPIO_WritePin(LD3_GPIO_Port, LD3_Pin, GPIO_PIN_RESET);
    }
    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}
```




## 6. HID pėlytės ir klaviatūros deskriptorių ir programos kodo pakeitimai

HID_MOUSE_REPORT_DESC_SIZE ir HID_KEYBOARD_REPORT_DESC_SIZE dydžiai skiriasi, nes pelytė ir klaviatūra yra skirtingi įrenginiai ir jie naudoja skirtingus deskriptoriaus paketų dydžius. Deskriptorius nurodo kaip reikia interpretuoti HID įrenginio

paketus ir suteikia informaciją apie paketo dydį baitais, jų paskirtį ir nurodo kuris baitas atsakingas už kurį mygtuką ar koordinatę. Pelytės deskriptorius didesnis dėl reikiamo didelio skaičiaus nurodyti XY koordinatę, kai klaviatūrai  pagrinde tereikia nusiųsti informaciją ar mygtukas nuspaustas.

Kitas keitimas *nInterfaceProtocol*, kuris nurodo, koks įrenginys prijungiamas kai operacinė sistema yra įjungiama: 0=none, 1 = klaviatūra, 2 = pelytė. Tai svarbu, nes per šitą protokolą BIOS/UEFI gali matyti ir valdyti klaviatūrą, kol tvarkyklė dar nėra paleista.

Taip pat keičiamas pats deskriptorius:

static uint8_t HID_KEYBOARD_ReportDesc[HID_KEYBOARD_REPORT_DESC_SIZE]

Jis sužymi 8 baitų protokolą ir nurodo, kurie mygtukai kuriem baitam priklauso: 1 baitas – modifikuojantys mygtukai (ctrl, alt, shift ir pnš.), 2 – rezervuotas, 3-8 – įprastieji klaviatūros mygtukai, kurių galima nuspausti iki 6 vienu metu.

uint8_t HID_Buffer[8]; Klaviatūros duomenų paketo dydis. Pakeistas iš 4 baitų pelytės raporto dydžio.

## 7. Studento numerio ir SysTick spausdinimas

### Pagrindinio ciklo kodas

```
while (1)
{
  /* USER CODE END WHILE */
    if(HAL_GPIO_ReadPin(USER_Btn_GPIO_Port, USER_Btn_Pin)==GPIO_PIN_SET){
      HAL_GPIO_WritePin(LD3_GPIO_Port, LD3_Pin, GPIO_PIN_SET);

      HID_SendString("Testavimas 20173764  Systick ");

      uint32_t tick = HAL_GetTick();

      HID_SendInteger(tick);
      HID_SendChar('\n');

      HAL_Delay(500);

      HAL_GPIO_WritePin(LD3_GPIO_Port, LD3_Pin, GPIO_PIN_RESET);
  }
  /* USER CODE BEGIN 3 */
}
```

Programa nuspaudus mygtuka įžiebia LED3 lemputę ir išsiunčia norima sakinį į funkcija SendString, tada gaunama SysTick reikšmė, paduodama funkcijai SendInteger, kuri pakeičia INT reikšmę į CHAR.

### SendString ir SendInteger

```
void HID_SendString (const char *str){
    while (*str){
        HID_SendChar(*str);
        str++;
    }

}

void HID_SendInteger(uint32_t num) {
    char buffer[12];
    sprintf(buffer, "%lu", num);
    HID_SendString(buffer);
}
```

Programa išsiunčia sakinį po vieną simbolį į SendChar funkciją. SendInteger daro tą patį prieš tai pakeitus kintamąjį iš INT į CHAR masyvą.

### ASCII į HID lentelė

```
const uint8_t ascii_to_hid[] = {
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, // 0-7
    0x2A, 0x2B, 0x28, 0x00, 0x00, 0x00, 0x00, 0x00, // 8-15 (backspace, tab, enter)
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, // 16-23
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, // 24-31
    0x2C, 0x1E, 0x34, 0x20, 0x21, 0x22, 0x24, 0x34, // 32-39 (space ! " # $ % & ')
    0x26, 0x27, 0x25, 0x2E, 0x36, 0x2D, 0x37, 0x38, // 40-47 ( ) * + , - . /
    0x27, 0x1E, 0x1F, 0x20, 0x21, 0x22, 0x23, 0x24, // 48-55 (0-7)
    0x25, 0x26, 0x33, 0x33, 0x36, 0x2E, 0x37, 0x38, // 56-63 (8-9 : ; < = > ?)
    0x1F, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09, 0x0A, // 64-71 (@ A-G)
    0x0B, 0x0C, 0x0D, 0x0E, 0x0F, 0x10, 0x11, 0x12, // 72-79 (H-O)
    0x13, 0x14, 0x15, 0x16, 0x17, 0x18, 0x19, 0x1A, // 80-87 (P-W)
    0x1B, 0x1C, 0x1D, 0x2F, 0x31, 0x30, 0x23, 0x2D, // 88-95 (X-Z [ \ ] ^ _)
    0x35, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09, 0x0A, // 96-103 (` a-g)
    0x0B, 0x0C, 0x0D, 0x0E, 0x0F, 0x10, 0x11, 0x12, // 104-111 (h-o)
    0x13, 0x14, 0x15, 0x16, 0x17, 0x18, 0x19, 0x1A, // 112-119 (p-w)
    0x1B, 0x1C, 0x1D, 0x2F, 0x31, 0x30, 0x35, 0x00  // 120-127 (x-z { | } ~)
};
/* USER CODE END 0 */
```

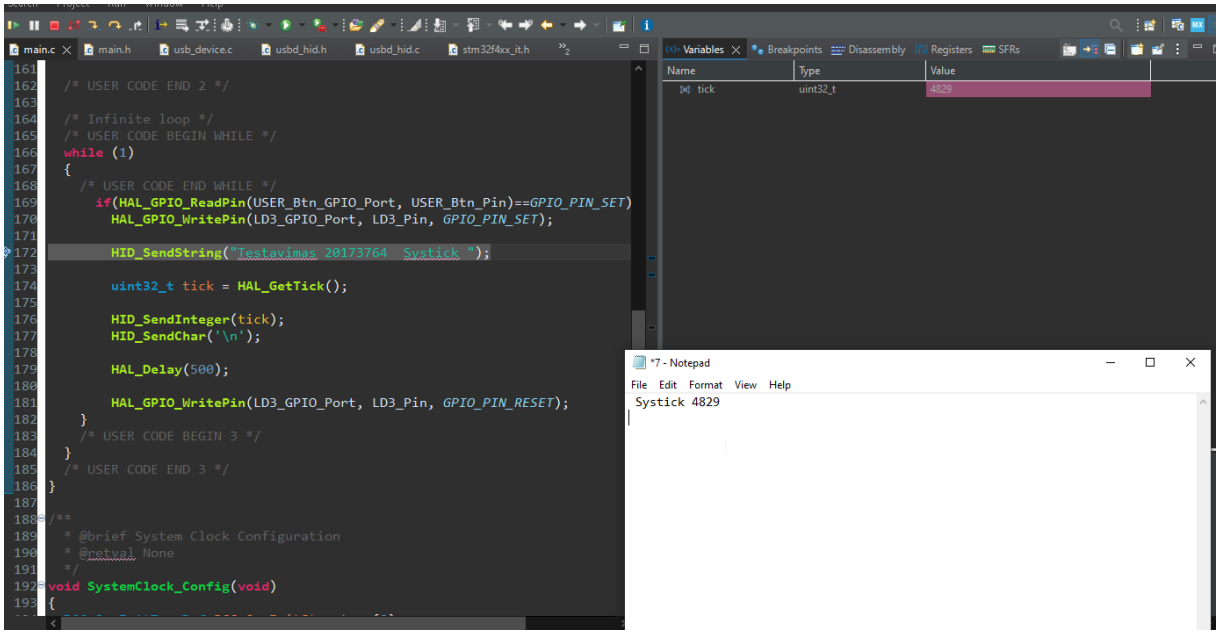
**SendChar**

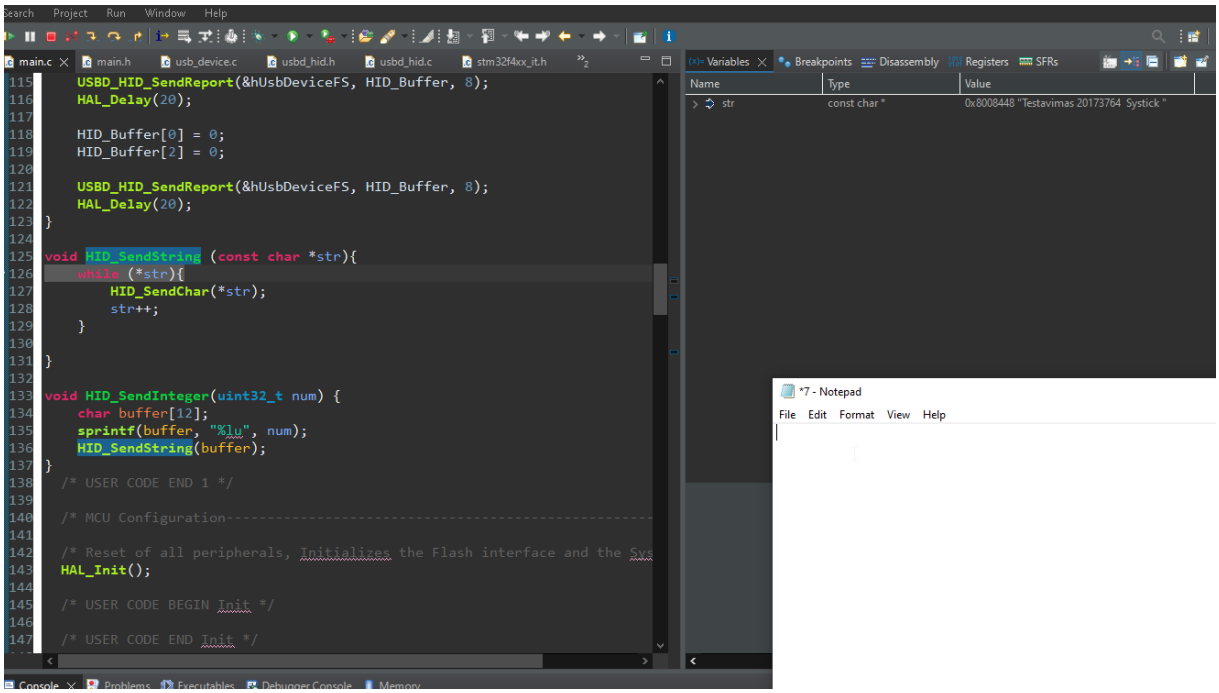
```
void HID_SendChar(char c){
    uint8_t modifier = 0;
    uint8_t keycode = 0;

    if (c >= 'a' && c <= 'z'){
        keycode = ascii_to_hid[(int)c];
    }
    else if (c >= 'A' && c <= 'Z'){
        modifier = 0x02;
        keycode = ascii_to_hid[(int)c];
    }
    else if (c >= '0' && c <= '9') {
        keycode = ascii_to_hid[(int)c];
    }
    else if (c == ' ') {
        keycode = 0x2C; // Space
    }
    else {
        keycode = ascii_to_hid[(int)c];
    }

    HID_Buffer[0] = modifier;
    HID_Buffer[2] = keycode;

    USBD_HID_SendReport(&hUsbDeviceFS, HID_Buffer, 8);
    HAL_Delay(20);

    HID_Buffer[0] = 0;
    HID_Buffer[2] = 0;

    USBD_HID_SendReport(&hUsbDeviceFS, HID_Buffer, 8);
    HAL_Delay(20);
}
```

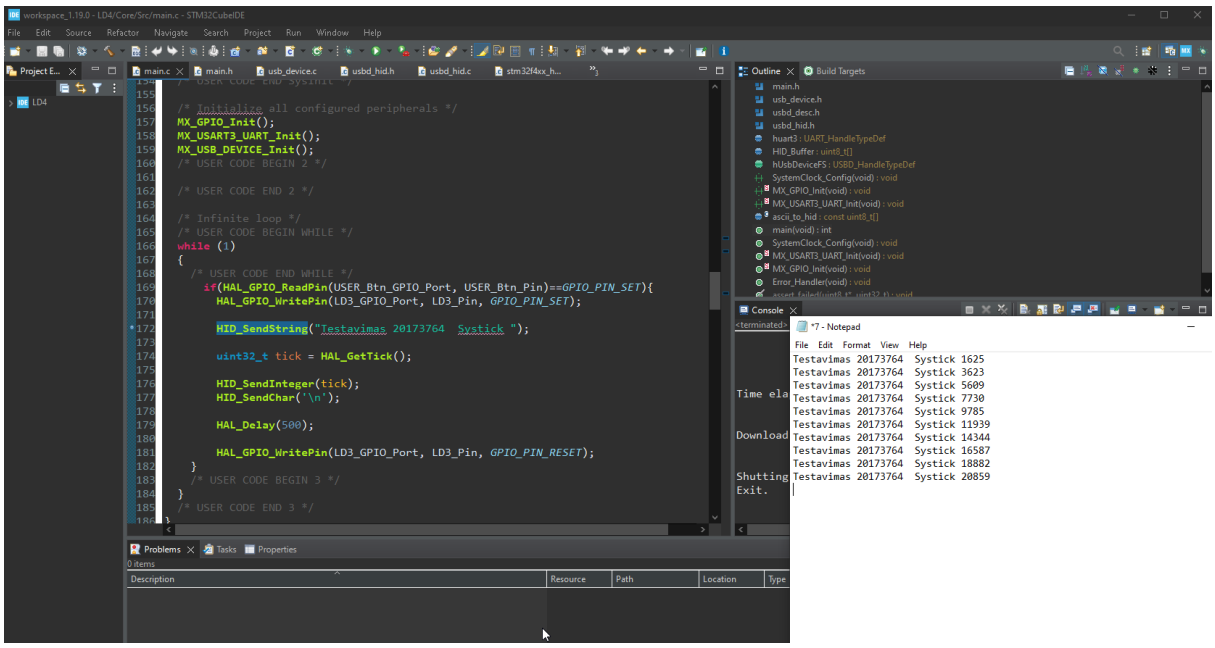
Programa iš apibrėžto *ascii_to_hid* LUT paima simbolio reikšmę HID formatu ir jo reikšmę nurodo *keycode* kintamajam, jei raidė didžioji arba siunčiamas specialusis simbolis, nustatoma modifikuojančio mygtuko bito reikšmė (pvz. SHIFT), visa tai įdedama į HID buferį pagal HID klaviatūros pranešimo šabloną (1 bitas – modifikuojantiems mygtukams, 2 rezervuotas, 3 – nuspaustas klaviatūros mygtukas) ir išsiunčiama į kompiuterį.

## Programos demonstracija

Veikiant derinimui buvo sunku suspėti atidaryti teksto redagavimo programos langą, tai dalis siunčiamo teksto nusikirpdavo.




workspace_1.19.0 - LD4/Core/Src/main.c - STM32CubeIDE
File Edit Source Refactor Navigate Search Project Run Window Help
Project E...
LD4
main.c
main.h
usb_device.c
usbd_hid.h
usbd_hid.c
stm32f4xx_h...
/* USER CODE END SysInit */
/* Initialize all configured peripherals */
MX_GPIO_Init();
MX_USART3_UART_Init();
MX_USB_DEVICE_Init();
/* USER CODE BEGIN 2 */
/* USER CODE END 2 */
/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
/* USER CODE END WHILE */
if(HAL_GPIO_ReadPin(USER_Btn_GPIO_Port, USER_Btn_Pin)==GPIO_PIN_SET){
HAL_GPIO_WritePin(LD3_GPIO_Port, LD3_Pin, GPIO_PIN_SET);
HID_SendString("Testavimas 20173764 Systick ");
uint32_t tick = HAL_GetTick();
HID_SendInteger(tick);
HID_SendChar('\n');
HAL_Delay(500);
HAL_GPIO_WritePin(LD3_GPIO_Port, LD3_Pin, GPIO_PIN_RESET);
}
/* USER CODE BEGIN 3 */
}
/* USER CODE END 3 */
Problems
Tasks
Properties
0 items
Description
Resource
Path
Location
Type
Outline
Build Targets
main.h
usb_device.h
usbd_desc.h
usbd_hid.h
huart3 : UART_HandleTypeDef
HID_Buffer : uint8_t[]
hUsbDeviceFS : USBD_HandleTypeDef
SystemClock_Config(void) : void
MX_GPIO_Init(void) : void
MX_USART3_UART_Init(void) : void
ascii_to_hid : const uint8_t[]
main(void) : int
Error_Handler(void) : void
Console
Time ela
Downloac
Shutting
Exit.
*7 - Notepad
File Edit Format View Help
Testavimas 20173764 Systick 1625
Testavimas 20173764 Systick 3623
Testavimas 20173764 Systick 5609
Testavimas 20173764 Systick 7730
Testavimas 20173764 Systick 9785
Testavimas 20173764 Systick 11939
Testavimas 20173764 Systick 14344
Testavimas 20173764 Systick 16587
Testavimas 20173764 Systick 18882
Testavimas 20173764 Systick 20859