**Lesson 34**

**HAL. USB. HID**

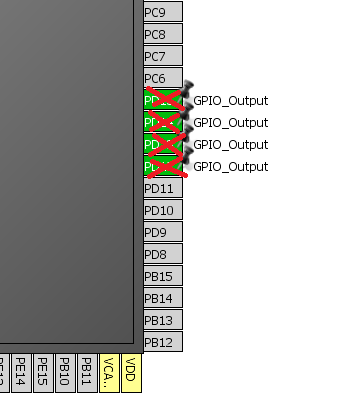
Today we will get acquainted with a new class for USB - this is the Human Interface Device or as the people say HID.

This class is different in that here we send short messages, we have a special probability that our data will all come, but at the moment it is not quite important. Typically, the speed is configured so that the data still reaches. The most important thing is that the data reaches quickly, so such interfaces, such classes are used when programming devices such as a mouse, joystick. keyboard and the like. HID class can be either predefined, that is, it is intended for specific existing types of devices, and arbitrary ( **custom** ). With the second type, we'll work a little on the next activity, and today we'll work with the first class and try to emulate the mouse.

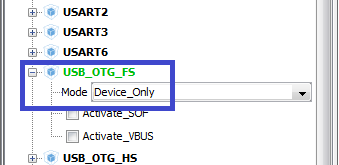
And that with this class a little bit more to understand, we'll try to implement it in our controller **STM32F407VG** .

To do this, we first of all, of course, make a project.

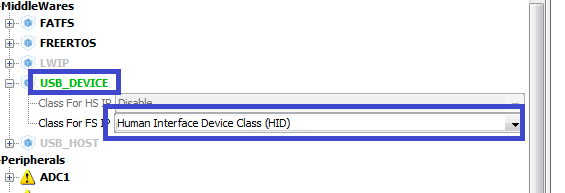
The project is created from the project TEST002. Call it USB\_DEVICE\_HID. Run the project in the Cube, turn off the output LEDs, leave the button



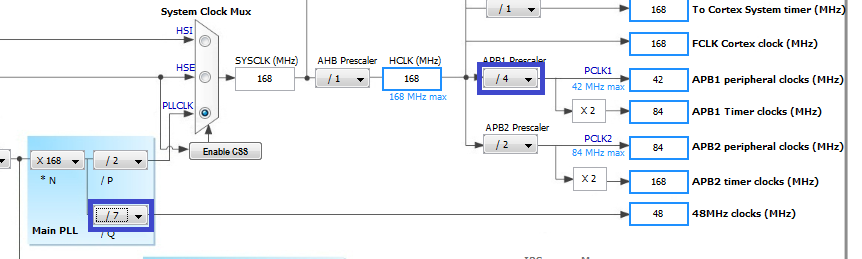
Turn on USB\_OTG\_FS in Device\_Only mode.



In the USB Device, select the Human Interface Device Class (HID)



In Clock Configuration, select the following dividers

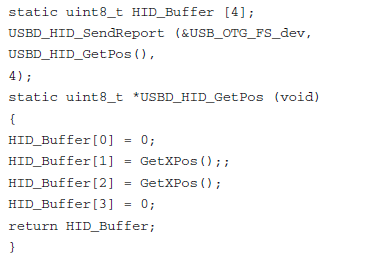


In Configuration, do not touch anything.

Generate and run the project.

We will collect the project and set up the programmer for auto-cutting.

Download the file  **USB Library User manual.pdf from the official site of ST Microelectronics and open it**on page 39. It gives the function to send the report to the device on the **host** . As a host we will have a **PC**



Connect the file to main.c

#include "main.h"

**#include "usbd\_hid.h"**

/ \* USER CODE END Includes \* /

Add variables to the same

/ \* USER CODE BEGIN PV \* /

/ \* Private variables ------------------- \* /

**#define CURSOR\_STEP 1**

**extern USBD\_HandleTypeDef hUsbDeviceFS;**

**uint8\_t HID\_Buffer [4];**

Let's write the function of preparing a buffer for transmission to the bus

/ \* USER CODE BEGIN 0 \* /

**static void GetPointerData (uint8\_t \* pbuf)**

**{**

**static uint8\_t cnt = 0; // loop counter**

**static uint8\_t drct = 0; // the direction of the cursor**

**int8\_t x = 0, y = 0, but = 0;**

**if ((cnt <20) && (drct == 0))**

**{**

**drct = 0;**

**x = CURSOR\_STEP;**

**y = CURSOR\_STEP;**

**cnt ++;**

**}**

**else if (cnt> 0)**

**{**

**drct = 1;**

**x = -CURSOR\_STEP;**

**y = -CURSOR\_STEP;**

**cnt-;**

**}**

**else drct = 0;**

**pbuf [0] = but;**

**pbuf [1] = x;**

**pbuf [2] = y;**

**pbuf [3] = 0;**

**}**

/ \* USER CODE END 0 \* /

In an infinite loop, we will add the buffer transfer code to the bus

  / \* USER CODE BEGIN WHILE \* /

  while (1)

  {

**HAL\_Delay (100);**

**GetPointerData (HID\_Buffer);**

**USBD\_HID\_SendReport (& hUsbDeviceFS, HID\_Buffer, 4);**

  / \* USER CODE END WHILE \* /

We will collect the project. We will sew the controller and see the result.

The cursor is crawling diagonally forward and backward.

To see all this, I recommend watching a video tutorial that will open if you click on the picture at the bottom of the page. Also in the video tutorial I give some explanations about the types of HID class descriptors and much more information. So welcome to the channel on YouTube, there you still will not see.

Now play around with the emulation of the mouse buttons.

We will comment out some of the code in the buffer preparation function and add code for emulating the left mouse button

  int8\_t x = 0, y = 0, but = 0;

// if ((cnt <20) && (drct == 0))

// {

// drct = 0;

// x = CURSOR\_STEP;

// y = CURSOR\_STEP;

// cnt ++;

//}

// else if (cnt> 0)

// {

// drct = 1;

// x = -CURSOR\_STEP;

// y = -CURSOR\_STEP;

// cnt-;

//}

// else drct = 0;

**if (HAL\_GPIO\_ReadPin (GPIOA, GPIO\_PIN\_0) == GPIO\_PIN\_SET) but = 1;**

**else but = 0;**

        pbuf [0] = but;

        pbuf [1] = x;

        pbuf [2] = y;

        pbuf [3] = 0;

}

We collect, sew and try to press the button USER .

Let's replace one by two

// else drct = 0;

        if (HAL\_GPIO\_ReadPin (GPIOA, GPIO\_PIN\_0) == GPIO\_PIN\_SET) but = **2**;

        else but = 0;

Let's see the result.