# Semestral work report

# USB device

## Goal

The goal of the semestral work was to develop an own library for USB initialization and communication for STM32 ARM processor based on given development board STM3240-GEVAL. The device should pass enumeration stage as well as include application layer. For purpose of the semestral work Human-Interface device (HID) - Computer Mouse - was chosen.

## Implementation

The initial setting was done in accordance with manufacturer reference manual. On GET_DESCRIPTOR request the device sends following reply – refer table 1.

| Data | Description |
|---|---|
| 18 | Length of descriptor (in bytes) |
| 1 | Descriptor Type |
| 0x01, 0x10 | USB version (= 1.1) |
| 0x00, 0x00, 0x00 | Device class, subclass, protocol |
| 64 | Max packet size (bytes) |
| 0xDE 0xAD 0xBE 0xEF | Vendor ID, Product ID |
| 0x00, 0x00 | Device version, iManufacturer |
| 0x00, 0x00 | iProduct, iSerialNumber |
| 1 | Number of configurations |

Table 1. The device descriptor

After GET_DESCRIPTOR the host usually prompts to SET_ADRESS request. When the request is acknowledged and zero-length packet sent back, all the communication switches to the given address. Typically, the host asks then for Configuration descriptor and Endpoint descriptor (please refer source code) from which it can define the device type and class and other parameters needed to establish communication. When descriptor is received and it make sense, the host sends SET_CONFIGUTATION and SET_IDLE commands and requests HID Report. When the report is received and acknowledged, the device is deemed as passed enumeration stage. Since that, the host typically communicates through IN endpoint.

## Device configuration

The HID device consists of 2 Endpoints. Endpoint 0 is supposed only for Control data (both IN and OUT). To transmit mouse data, the second Endpoint 1 is used – IN & Interrupt.

The device typically NAKs IN requests in case no data available or sends updated state to a PC. Transmission happens based on timer – approximately every 20ms the device checks its buttons state and send appropriate message.

The manipulation supposes to use on-board buttons and is designed in following way:

**TAMPER BUTTON** – left button (button 0)

**KEY BUTTON** – right button (button 1)

**Joystick – UP DOWN LEFT RIGHT** – to control mouse position

**Joystick – SELECT** – wheel button

Combination of **KEY BUTTON + Joystick UP/DOWN** – Wheel rotation

## LED indication and other aspects

In case the device is switched on, LED4 is turned permanently ON to indicate ON state.

In case USB is connected, LED3 is turned ON to indicate connection. It switches off when connection is lost.

LED2 is toggling with a timer speed, LED1 toggles with data transition speed, so it can be used as transmitting indicator.

The device status is also displayed on the on-board LCD. The mouse is supposed to support multiple disconnects and connects without reset required.

## Conclusions

In this semestral work we get familiar with development of a USB communication device. We get acquainted with principles of USB communication, get into details of transition packets using oscilloscope as well as get practice in programming of embedded device. Despite the USB is a well-known standard, development of a robust communication library is a complex task and requires big amount of time and experience involved.