USB Virtual Reality HID

Functional Requirements List and Performance Specifications

Students:

Weston Taylor

&

Christopher Budzynski

Project Advisor:

Dr. Malinowski

November 25, 2008

Introduction

The purpose of this project is to create a USB (Universal Serial Bus) Virtual Reality HID (Human Interface Device). This USB HID will interface with personal computers and their programs by emulating both a mouse and a keyboard. The HID will translate user movements into on-screen in-game actions, to provide the user with a more lifelike interactive platform for PC games and other virtual environments.

The USB HID will consist of three major subsystems all working together to provide the overall interactive environment. The first subsystem is the Handheld Pointing Device, which will use accelerometer and gyroscope readings, to translate the user's arm movements into on-screen mouse locations. The second subsystem is the Step Pad; this subsystem will provide the commands that allow the user to move throughout the virtual environment. The USB communication board is the final subsystem. This board will receive keyboard and mouse location data from the other two subsystems; it will then translate this data into USB mouse and keyboard commands, which create in-game actions when sent to the host PC.

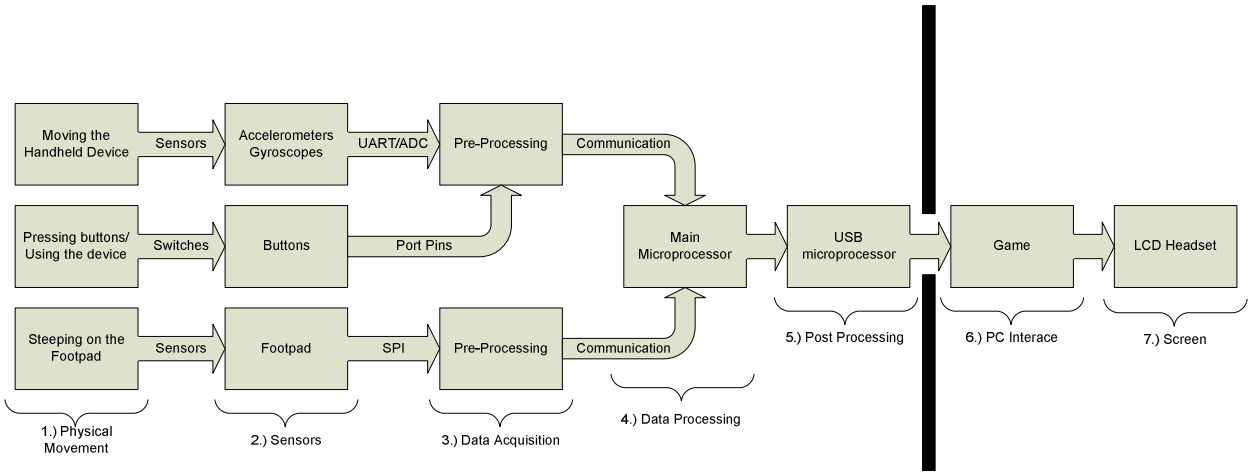
Goals

The overall goal is to translate user movements into on-screen actions to provide a more realistic in-game experience for the user. Here are the detailed objectives that will make this goal attainable:

- USB communication with a PC, emulating both a mouse and keyboard
- Translation of accelerometer and gyroscope readings into on-screen mouse locations
- Processing of data and USB communication using embedded systems
- Development of a Step Pad for in-game movement (or interfacing of an already produced step pad for the embedded application)
- Use of a Headset instead of a monitor to view in-game results
- All embedded programming done using the C programming language, to provide portability and reusability of code
- If time permits, communication between subsystems will use the most appropriate form of wireless communication (ZigBee, Bluetooth, etc.)

System Block Diagram

Figure 1: Over-all System Block Diagram



1.) Physical movement always initiates interaction with the device. Maneuvering the handheld device will determine where the person is looking and pressing its buttons simulate in game actions. Pressure on the footpad sensors indicates movement within the virtual environment.

2.) The sensors will translate a user's movement into a useable form, such as serial data or analog voltages. Accelerometers and gyroscopes will be used to calculate on-screen position, while buttons will be used to simulate functions in the game. Lastly, the footpad will allow a user to step in any direction to make the on-screen character move within the virtual environment.

3.) The data acquisition algorithms will be determined by the sensors' output formats. However, the buttons on the handheld device itself will use port pins, and the step pad will use a modified serial interface. Pre-processing will prepare the data for processing by using software gain to insure that all calculations are done in the same frame of reference and units. Also, depending on the data, offsets may be needed to center the data in the proper area.

4.) Data processing involves taking the cleaned data from pre-processing and converting it to useful data that can be used by the USB communication module. Integrations will be needed for both the accelerometers and gyroscopes in order to get position data from rates of change. In addition, the buttons on the device and in the footpad need to be processed into their corresponding keyboard button presses.

5.) Post processing involves taking the data from data processing and forming packets to be sent across USB to the PC.

6.) The PC will be running a video game, and the data passed over USB will translate into on-screen actions and movements within the virtual environment.

7.) The LCD headset will connect to the PC; this will provide the user freedom to turn round without worrying about a fixed screen.

As previously stated, the primary objective of this project is to translate user motions into on-screen actions; therefore, the project will focus on the blocks to the left of the line in Figure 1 and existing PC software and LCD headset will be used.

Subsystem Functional Requirements and Performance Specifications (Hardware):

- According to preliminary testing, the average person can move a handheld device at a maximum of 350-500 degrees per second (while holding a stopwatch). Since humans can move a handheld device up to 500 degrees/sec, the accelerometers and gyroscopes must be able to measure this movement without saturation. The tests were run with the individual holding an object that weighs less than a pound and only the arm was moving. However, when this system is used, the handheld device will weigh between 5-10 lbs and upper body movement will be required; both of which will slow the user's movement. Using this information, the sensors only need to be able to measure up to approximately 250-300 degrees/sec. Also, since the USB protocol sends packets every 1ms, the sensors' readings must update at least this fast or faster to reduce latency.

- In regards to the footpad, most users will weigh less than 300 lbs; therefore, the footpad must function effortlessly up to this body weight threshold.

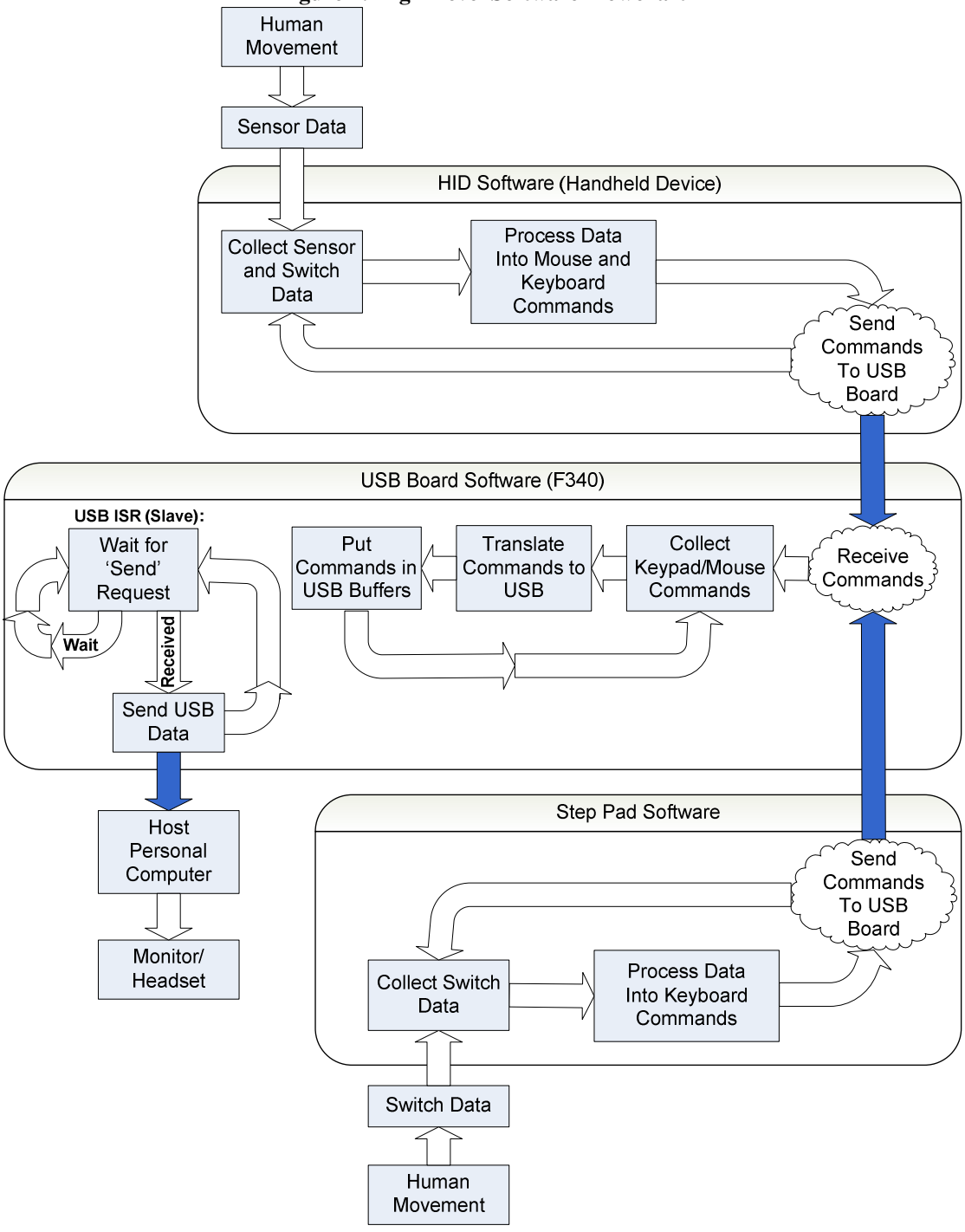
- USB 2.0 compatible devices will be used in Full Speed mode (12 Mbits/s) with frames being sent every 1ms (as specified by the USB protocol).

- All hardware devices must operate from 0°C to 40°C; this is not a large temperature range because this will be a consumer electronic product used inside at room temperature.

High-Level Software Flowchart

This project will use multiple embedded systems to provide data acquisition, data processing, intersystem communication, and USB communication; this means that a major portion of the project will be software development. The overall high-level software flowchart can be seen in Figure 2 (below).

Figure 2: High-Level Software Flowchart



Handheld Device Software (top of Figure 2)

This software block has a simple task; it must collect the sensor and button data, translate that data into keyboard and mouse movement commands, and send those commands to the USB communication board. The "Collect Sensor and Switch Data" block will use the Analog-to-Digital converter, the SPI, or the UART to collect sensor readings depending on the sensor hardware that is purchased. The next block will interpret that sensor data into keyboard button presses and mouse movement/position information. The final block sends the keyboard and mouse information to the USB board, using a wireless or serial communication method, so that the data can be forwarded to the PC.

Step Pad Software (bottom of Figure 2)

This software block performs the same functionality as the Handheld Device Software except that the Step Pad will not influence the mouse movement at all, so only keyboard button press information will be sent to the USB board.

USB Board Software (middle of Figure 2)

The USB board software's main function is to collect all keyboard and mouse movement information from the other devices and send it to the PC over USB. Its first software block, "Receive Commands", does just that; it receives the information from the other boards using either wireless or serial communication. Next, the two data streams are merged together so that all keyboard presses and mouse movements are recorded. The "Translate Commands to USB" block translates the collected mouse and keyboard information into USB bytes that can be sent to the PC. Finally, the last block puts the commands into the USB API buffers so that the USB ISR can send the data to the Host PC when requested.

Software Functional Requirements and Performance Specifications (Software):

- Since the USB protocol sends out data packets every 1 ms, the major requirement for the software is that all data collection, processing, and translation into the USB protocol be completed within this 1ms period (including the intersystem communication). This will ensure that updated information is always transferred to the PC and that the worst case latency will be limited to 1 ms. However, since human reaction time is considerably greater than 1 ms and standard monitor refresh rates are approximately 60 Hz (14.7 ms), an update period of up to 5 ms will be acceptable for computationally intensive data.

- Assuming that all calculations can be completed within 0.5 ms, this leaves 0.5 ms for the data transmission between systems. The initial estimations are that at least 2 bytes per system will need to be sent every 0.5 ms meaning that both the Handheld Device and the Step Pad subsystems must transfer their data in 0.25 ms. Initially, assuming no protocol overhead, the data throughput would need to be at least 64 Kbits/s (2 bytes/.25 ms). In reality, protocol overhead will be present and the throughput will need to be much larger.

- Initially, we will implement our system by emulating a USB mouse and keyboard composite device because they are easier to implement due to the many design examples provided on the internet. However, if time permits, functionality will be consolidated into a single USB gamepad device which will better interact with existing PC games due to the gamepad's inherent in-game functionalities (such as dedicated crouch and jump commands).

Conclusion

The most difficult part of the project will be the translation of accelerometer and gyroscope readings into mouse movement and on-screen position, while reducing the influence of noise, saturation, and drift due to accumulated error. Also, the device must emulate both a mouse and a keyboard from the same device; this could potentially introduce some USB coding difficulties. Once these hurdles are overcome, this project will provide an exhilarating interactive platform for many personal computer software environments.

References

[1] Silicon Labs, C8051F34x Data Sheet,
https://www.silabs.com/products/mcu/usb/Pages/C8051F34x.aspx

[2] Wikipedia, Inertial Navigation System,
http://en.wikipedia.org/wiki/Inertial_navigation_system

[3] D. Schertz, EE565 Fall 07 Lectures Notes 20 – 24, Bradley University

[4] Device Class Definition for Human Interface Device (HID) Version 1.11
http://www.usb.org/developers/hidpage/

[5] HID Usage Tables Version 1.12
http://www.usb.org/developers/hidpage/