**USB\_KeyBoard**

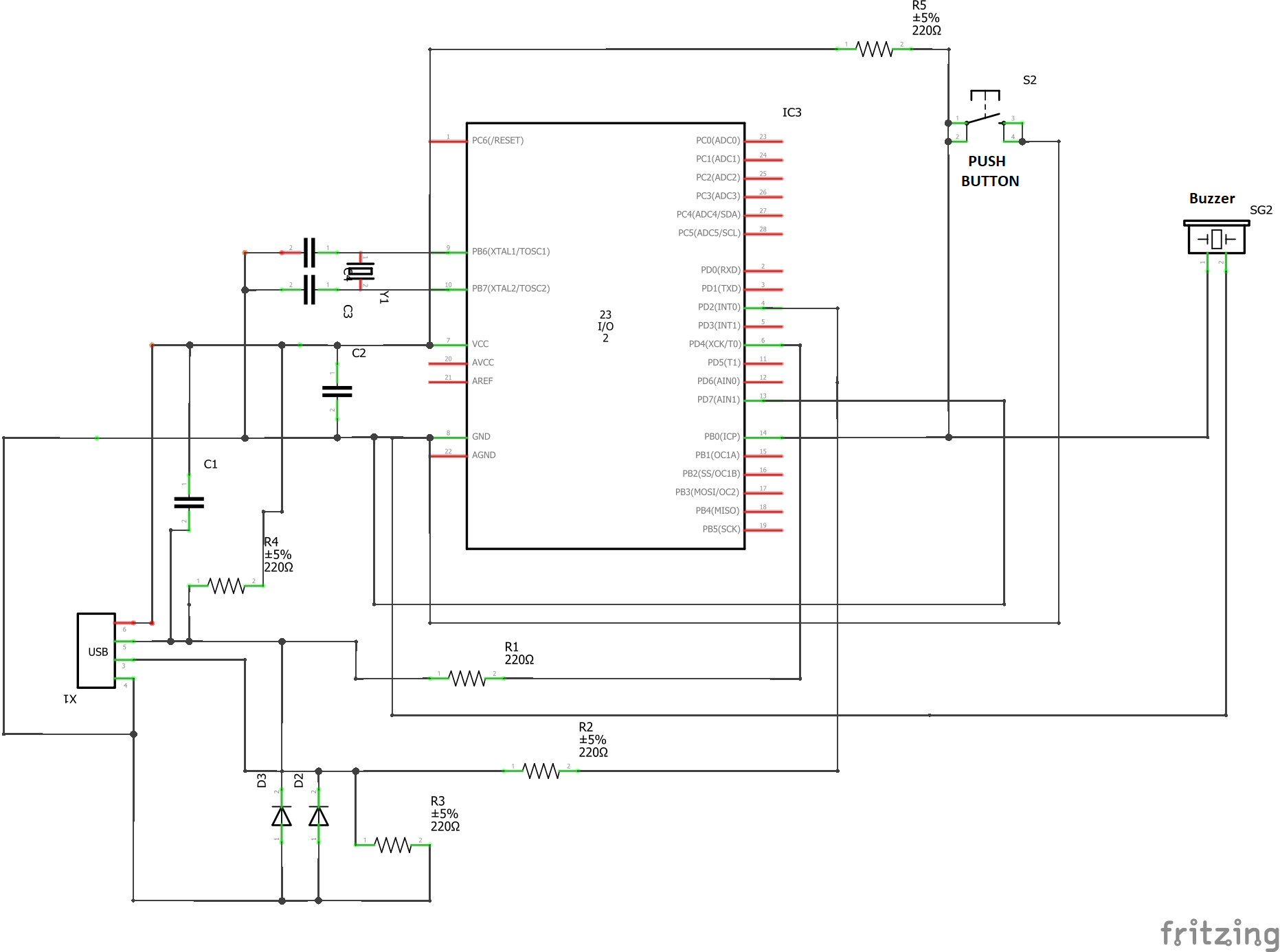
Project submission for Device Drivers course.

### Summary

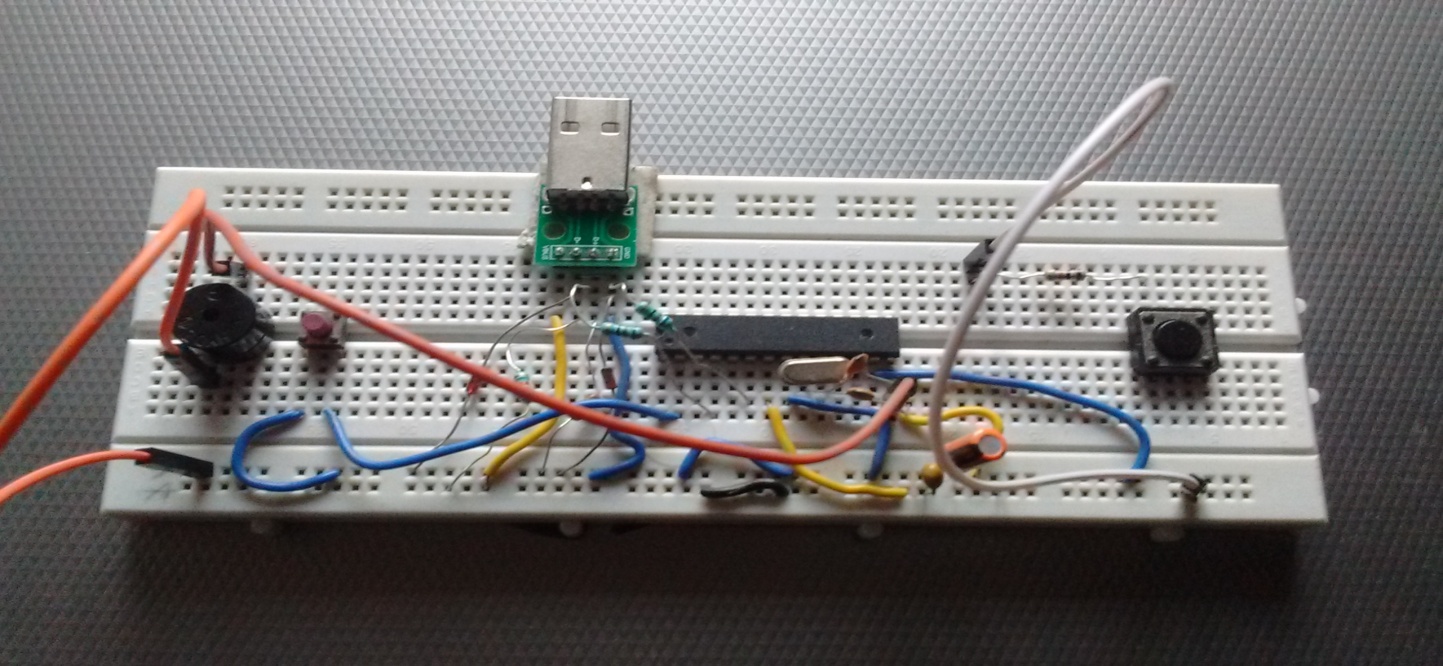
In this project we are making a keyboard which is interfaced via USB. The hardware part includes

ATMEGA8 microcontroller and various components to be interfaced to USB. The firmware part includes a C program that controls PORTB output depending on the USB control request received from the USB core. The User-space program reads the event file generated by Driver code and key press is detected. Driver code polls the hardware by sending control messages using usb\_control\_msg(). The driver is automatically plugged in (registered) on detection of an allowed device and unregistered on disconnection.

**HARDWARE SCHEMATIC**

****

**HARDWARE SNAPSHOTS**

****

**FIRMWARE DESCRIPTION-**

This firmware closely follows the V-USB implementation by Starkjohann. "make hex" builds main.hex file and "make flash" flashes hex file to the hardware but requires hardware to be reset once to enter USBasp mode.

**Steps To be Followed**

**Step1: This project directory contains keyboard\_firmware folder and it contains main.hex file which is needed to be uploaded to the usb hw device which we have prepared using ATMEGA 8 microcontroller.**

**Note:- Generation and flashing of Hex file is done by opening a terminal in keyboard\_firmware folder and typing the following commands**

**a. make hex**

**b. sudo avrdude -c usbasp -p m8 -U flash:w:main.hex**

**Step2: This project directory contains usb\_keyboard.c file which is the host side driver written to poll the hw device with usb control messages in keypress\_pollfn() function which is inturn called from keypress\_probe() function. Probe function basically calls whenever usb hw device is attached. Whenever usb control messg is received by usb hw device, status of portB0(pin 14) to which push button is connected is taken and sent to driver. Note that status is “bit 1” if button is pressed or “bit 0” if button isn’t pressed.**

**Note:- use following commands to compile and insert module in kernel**

**a. make clean**

**b. make all**

**c. sudo insmod usb\_keyboard.ko**

**(for checking kernel log use dmesg -wH)**

**Incase if you are done with using the driver module it can be removed from kernel by giving following command**

**sudo rmmod usb\_keyboard.ko**

**Step3: User Space Application present in same project folder with the name “input\_sub\_app.c” will read the event file from “dev/input/event5”. Different fields of “struct input\_event” is read to get complete information of the event such as type, value and event code.**

**Note :- use following commands to compile and run user space application**

**Compile:- gcc -o input\_sub\_app input\_sub\_app.c**

**Run:- ./input\_sub\_app**

### Authors

G.V.S.Manoj Kumar and Himanshu Singh ME Embedded Systems, BITS Pilani

## License

* This project is licensed under the GNU-GPL License.