University of Colorado - Boulder

ASEN 5067 - Microavionics for Aerospace

02 November, 2021

Final Project Proposal

Author:
William Watkins[a]

[a]SID:106338366

Professor:
Trudy Schwarz

Ann and H.J. Smead
Aerospace Engineering Sciences
UNIVERSITY OF COLORADO **BOULDER**

I. Overview

This proposal, submitted to meet the requirements of Microavionics (ASEN5067), describes the technical and functional elements of a media control center for a desktop personal computer. This media control center will eventually be a stand-alone device consisting of a volume control, a mute/unmute control, a play/pause control, and a next/previous track control. The controls will be assembled on a breadboard and the commands will be set via UART on the EasyPIC Pro.

Initial success will be achieved if movement of the rotary encoder successfully changes the volume setting on the PC. Further levels of success for this project include additional functionality of the media control center. For example, a second level of success would include the media control center having a functioning mute button. The final levels of success include having the play/pause and next/previous track controls functional. My primary off ramp would be purchasing a PIC18 MCU that supports native USB communication instead of USB-UART.

Table 1 Levels of Success.

Level	Objectives
1	Rotary Encoder successfully changes the volume of the host PC.
2	Mute button successfully mutes the host PC.
3	Play/pause button successfully starts/stops currently playing track.
4	Next/previous track buttons skip to the next track/go to the previous track.

Originally, an LCD was going to be used to display the current volume level of the PC and if it was muted or not. However, there is not a way for a external device to request the current volume level or current track playing without writing a separate program to automatically send that data. If the first three levels of success are met before the project is due, this will be a stretch goal, but will not be incorporated into this project proposal.

II. Software Flowchart

The base functionality of the project is straightforward. Upon power up, the code will first identify itself to the Host PC as a peripheral device. Then, it will continuously poll Port D to determine if a button has been pressed or the rotary encoder has been rotated. If either of these happens, a subroutine will be called to determine which occurred, which button was pressed if necessary, and then send the appropriate command via USB-UART to the host PC. The commands to be sent are determined by the USB HID Usage Tables. Fig. 1 illustrates how the program will run.

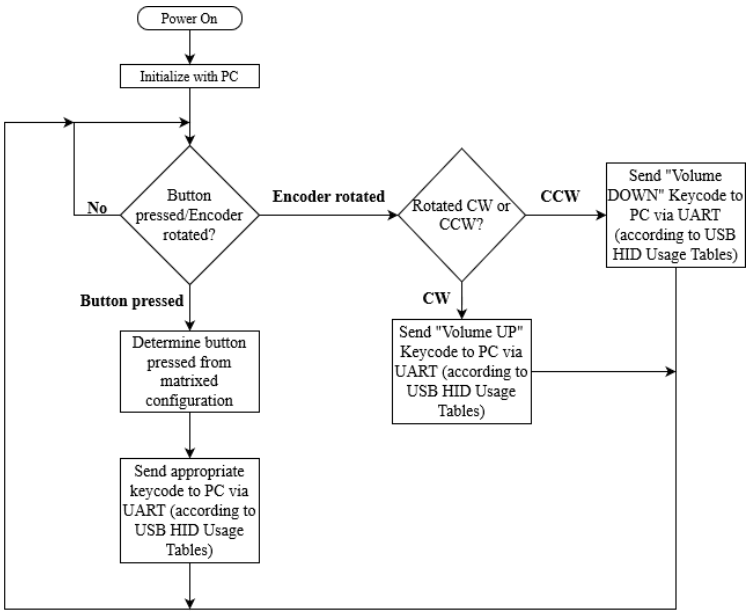

Fig. 1 Media Control Center Software Flowchart.

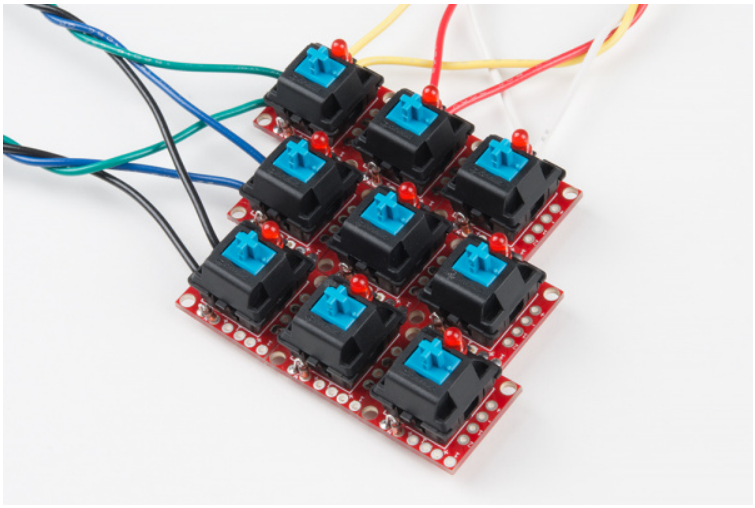

Fig. 2 A matrixed keyboard configuration.

A "matrixed" configuration for the buttons is one that does not have wires running to each button. Instead, the buttons are arranged in a row-column (matrix) format, and have one wire per row/column. An example can be seen in fig. 2. When the appropriate row/column inputs are driven high simultaneously, that indicates which button has been pressed.

III. Functional Block Diagram

The functional block diagram for this project is seen below. The matrixed keyboard buttons and rotary encoder will be connected to Port D for ease of use. Port D is then accessed by the PIC18F87K22. When the program determines a

signal needs to be sent, it will be done using the FTDI chip in order to use the USB-UART protocol. This data will be sent to the host PC using a B-Male to A-Male USB cable.

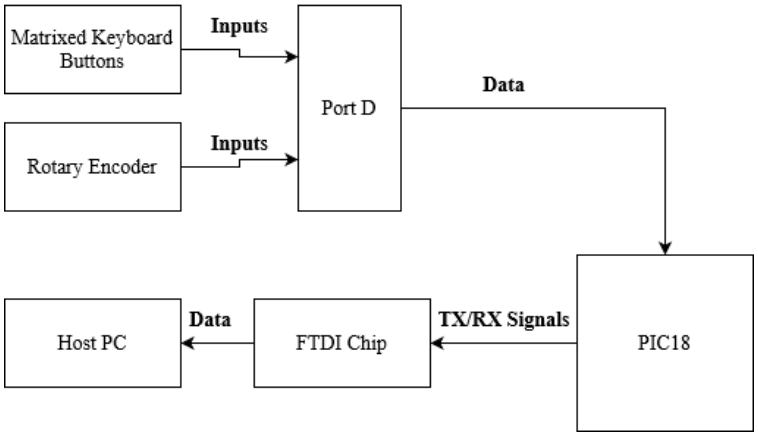

Fig. 3 Media Control Center Functional Block Diagram.

IV. Wiring Diagram

The wiring diagram for the media control center is below. Since the EasyPIC Pro board will be used for most of the data handling and sending, the only items that require additional power and ground handling are the keyboard buttons set up in the matrixed array. This will be done by having common 5V and Ground rails on the breadboard, with the power supplied by the EasyPIC Pro board.

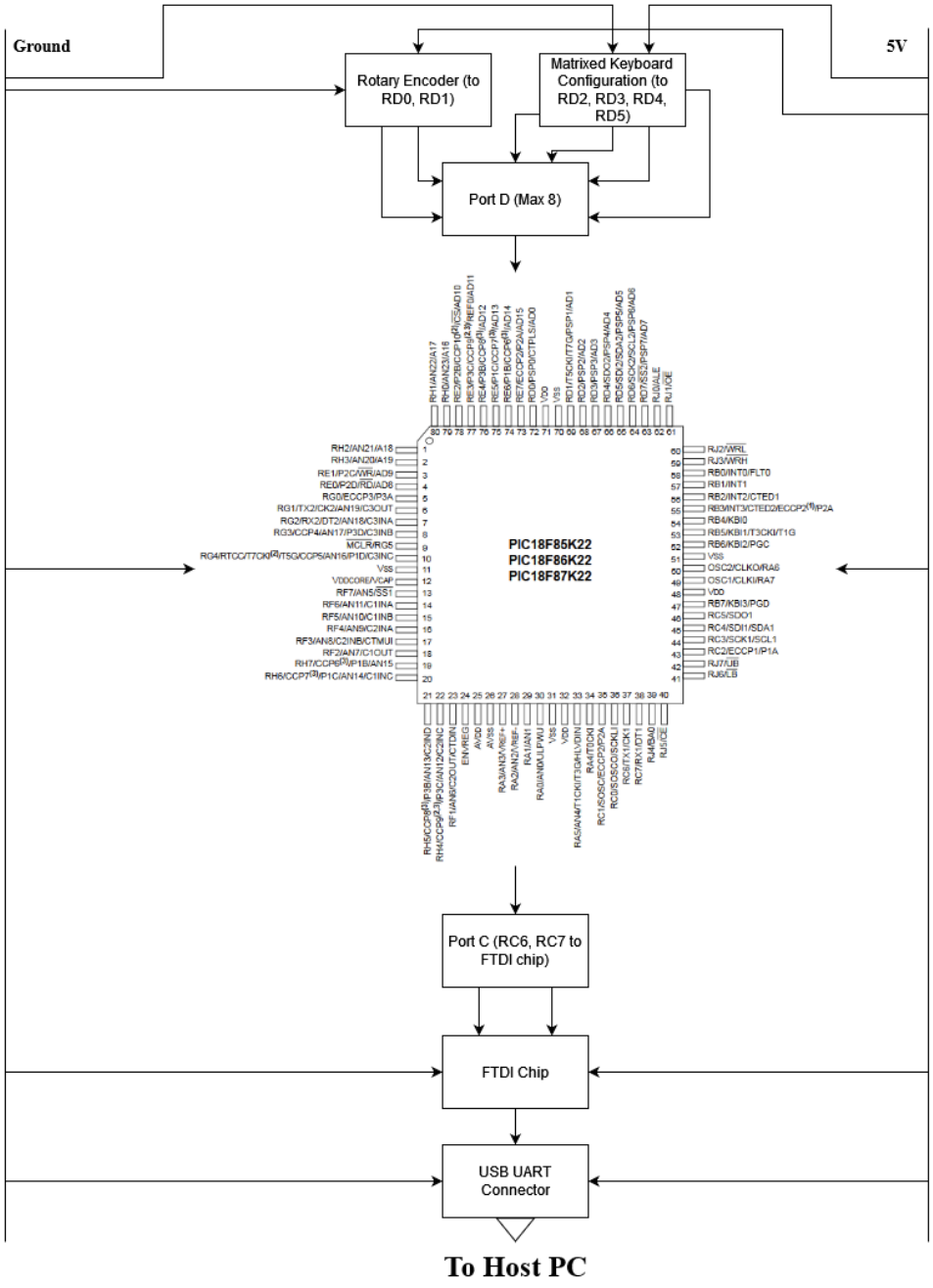

Fig. 4 Media Control Center Wiring Diagram.

V. Bill of Materials

The Bill of Materials is located below.

Table 2 Bill of Materials.

Part	Price	Order Status	Manufacturer	URL Link
Cherry MX Switch Breakout x4	$1.95 ea	Ordered	SparkFun	https://www.sparkfun.com/products/13773
Cherry MX Switch x4	$0.95 ea	Ordered	SparkFun	https://www.sparkfun.com/products/13834
Jumper Wires Standard 7" M/M - 30 AWG (30 Pack)	$2.25/pack	Ordered	SparkFun	https://www.sparkfun.com/products/11026
100EP514330R 330 Ohm Resistors, 1/4 W, 5% (Pack of 100)	$6.17/pack	Ordered	E-Projects	https://www.amazon.com/Projects-Resistors-Watt-Choose-Quantity/dp/B0185FGN98?th=1
Diode Small Signal - 1N4148, x4	$0.15 ea	Ordered	SparkFun	https://www.sparkfun.com/products/8588

VI. Status

Currently, the parts have been ordered and the psuedocode for the project is about halfway done. One of the biggest early roadblocks was identifying the correct way to interface with the computer to send commands. As I mentioned earlier in the proposal, the appropriate way to do that was identified in the USB HID Usage Tables document. However, the next roadblock will be correctly implementing those commands via the USB-UART interface. If I cannot get the commands to work via USB-UART, my plan is to purchase a PIC18 microcontroller that is able to use the native USB interface on the EasyPIC Pro, such as the PIC18F87J50. In order to get in front of this risk, however, I am going to be developing the code to communicate with the host PC first, as the polling of Port D should be relatively straightforward. In addition, there are a large number of open-source keyboard firmwares available on github and other code repository websites. The communities for these firmwares are extremely active and happy to help people who are developing their own firmware. These have already been beneficial in creating the psuedocode for this project.