Introduction

Parker

Developing a user interface is often one of the most time consuming and expensive aspects of building a device, and can take up to 50% of the project's time and budget. We envisioned a device that solves this problem by making the user interface development process more accessible and approachable from the ground up. By prioritizing an easy setup and universal compatibility, we can revolutionize the way engineers and entrepreneurs interact with their prototypes

I am the project sponsor, so instead of a company assigning us a project, We’ve been developing an idea that I came up with. After pitching it to the Senior design professors, I put together the dream team. We've created something I'm super excited about! This is Kelly Mae, our team leader.

KM

ProtoControl is a user interface development platform, and we've built an engineering tool to make building a User interface easier. Protocontrol is a one stop shop for user interface design, with physical components and on-screen inputs that can be setup in under 15 minutes. For example, this is a protocontrol unit controlling this 3d printer turned claw machine![rest of the team introduces themselves]

(KM → Nida→Abisha→Parker→Will)

* + 

Begin

Intro

Will

* Website Overview (Angular, NoSQL, Flask)
  + Visuals: QR code for website, tech stack visuals
* How to use website
  + Flow chart
* How we identify data (DB schema)
  + Deployment diagram
* Build layout for msp430 demo

306 demo

Parker

* String parsing
* Challenge: Arbitrary digital components / oop
  + I created a comprehensive header defining classes for each of the digital compintents, so no matter how the profile is layed out, each component handles inputs correctly and outputs data correctly
  + Draw and touch functionality based on changing location
* Looks for physical components
  + Scans for i2c adress
* Sends serial data to user system
  + Sends message as human readable characters
* Outputs setup information on start, and echos Serial output to monitor

Abisha

* Users also have the choice to include hardware peripherals
* So far we have integrated Rotary Encoder, Keypad, Buttons, Joysticks and a Slider with ProtoControl
* Important to also have tactile components so the users have as many options as possible as they go through the iterative process (interchangeable)
* User experimenting with different layouts →Try different configurations easily
* Challenge: how can the user know where the data came from? Two solutions, one for digital components and one for physical components
* Why I2C? In order to have hot-swappable components (location and type), we need a way to distinguish between multiple components. We decided to utilize I2Cs addressability. Each hardware peripheral has a unique address
* Look for addresses within a certain range (on startup and every time we query the website) Embedded System
* Digital uses PCF8574 (gpio pin expander) and a custom PCB (up to 8 unique addresses)
  + Rotary encoder
  + Keypad
  + buttons
* Analog uses ADS1115 (ADC with I2C output) on a breakout board (up to 4 unique addresses)
  + Joystick
  + slider
* Challenge: finding libraries that integrate with pcf8574 chip (fail fast)

Nida

* Focuses on two products: main board and the component pcb
* Main board: acts as a shield between the microcontroller and the touchscreen, provides multiple options to be powered such as barrel jack and Vin ports (in addition to the microcontroller), UART port, two switches to switch between user profiles and most importantly the I2C ports and we had many iterations of this board before we finalized the design as you can see here on the table
* Component pcb: provides GPIO pins for I2C components and another port for these components to interact with the main board, 3 resistors to be able to assign unique address values to each component
* Challenges: being new to pcb design and learning how to use KiCad, learning how to integrate vias to get rid of inner layers, learning how to flood planes on the board to minimize the number of traces
* The total current draw of protocontrol even with components connected is about 200 mA which means the total power usage is approximately 1W

Alarm Demo

* Now that you have an idea of both the software and hardware aspect of our device, we would like to present an example application of protocontrol: a security system for your room. Let’s say you want a heads up before someone walks into your bedroom, protocontrol can help you control a system consisting of an ultrasonic sensor and a buzzer that goes off when it detects motion outside. Abisha will walk you through how you can utilize this system with the protocontrol.
* [While demonstrating the system] The system is initially disarmed. In order to arm the system, you press the toggle . At this point the sensors are On. When you get close to the sensor the rgb led turns red and the buzzer starts sounding. If you move outside of the radius, the led and buzzer turns off. However the system is still armed. In order to disarm the system you must enter a pin. If the pin is invalid, the system remains armed. If the pin is valid, the sensors turn off and motion isn’t detected. Move the keypad arround
* Introduce the claw machine: going from a hobbyist type project to a more complicated system. Protocontrol is equally effective

KM

Claw machine demo

* Inputs are joystick and toggle
* String is sent to pi and parsed into gcode command or servo state