CyWear

Project Brief & Specifications

Revision 1.2

Non-Confidential

CyWear Project Brief & Specifications

Document History

Revision	Date	Confidentiality	Changes
1.0	Aug 07, 2024	Confidential	Initial Release
1.1	Sept 22, 2024	Confidential	More detailed project structure
1.2	Oct 05, 2024	Non-Confidential	Revised external links, added watermark.

Disclaimer

THE PROJECT IS DESIGNED, DEVELOPED AND TESTED AS A LEARNING EXERCISE, SOLELY BY ONE ENGINEER. IT SHOULD NOT BE USED IN ANY CRITICAL INFRASTRUCTURE AS RELIABILITY IS NOT GUARANTEED.

Having said that, the project will be developed with the highest quality achievable by a single developer, following best known practices. Additionally, all design considerations, as well as all development decisions will be documented in documents of this kind, giving full insight on the project's inner workings, covering all decisions made; allowing external observers to learn about the project, and potentially contribute to or build upon the project.

Project Status

The project is in its very early stages of development. A brief of the whole system design has been drafted but, at the moment, has NOT BEEN MADE PUBLIC.

Web Address

`Https://github.com/CyWear`

Contents

1 Project Objective

CyWear, short for Cyber Wearable Gadget, is a smart glove designed to wirelessly trigger customisable actions on various devices. The main goal is to create a modular platform that allows for easy and intuitive control of different digital devices.

The system is designed to be flexible and can be expanded as needed. This means that custom actions can be created, different devices and services can be integrated, and new features can be added when required. By focusing on a clear structure and modular design, CyWear ensures that the system is easy to modify and scale without needing major changes to the core design.

The idea is to have a lightweight wearable gadget that, when in range of the central hub, transmits sensor data to the hub, which then parses it to trigger actions on either of the connected devices. These actions and devices can be configured through the system's desktop program. Eventually, a mobile app may be developed to improve usability, but this is something that is well beyond the current project scope.

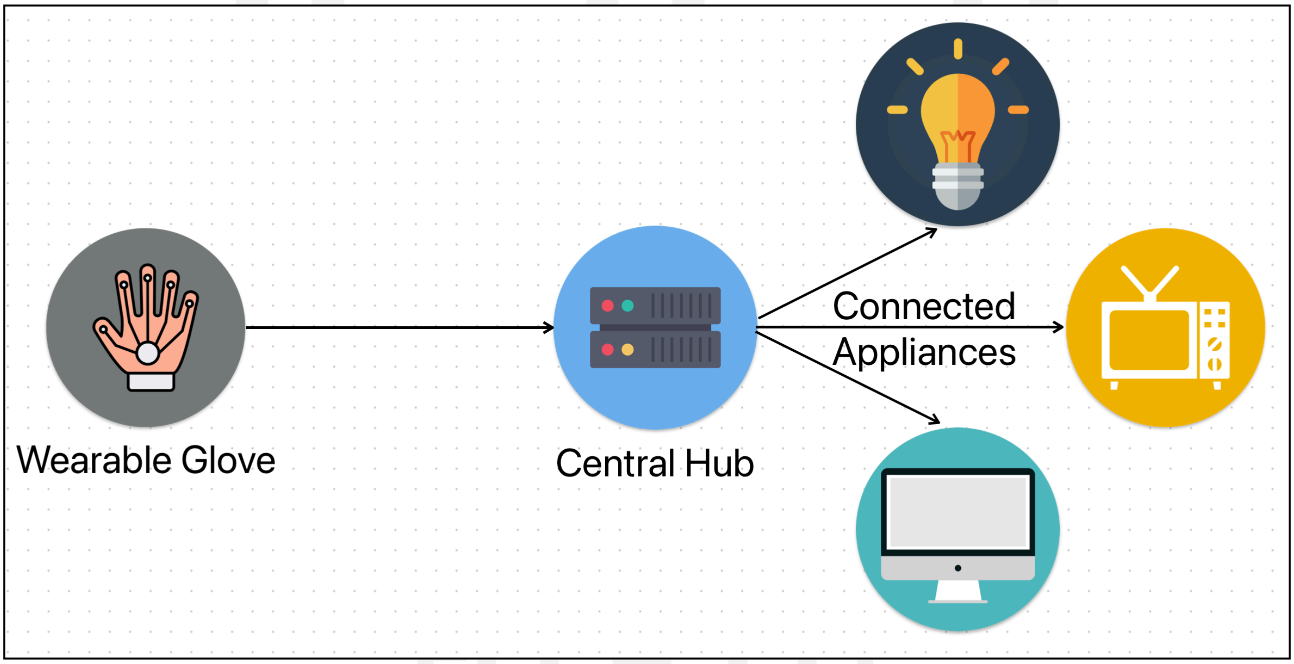

Image 1: System Structure

Eventually, the Central Hub is will be able to connect multiple gloves for each of their custom

The system consists of multiple components, each addressing different aspects of the project, each maintained in its own repository, all hosted on the GitHub organisation: `https://github.com/CyWear`

*note not all repositories are public.

2 Scope & Deliverables

The scope of CyWear includes the design, development, and testing of a smart glove system that can wirelessly trigger customisable actions on various devices.

The system will consist of multiple components, including:

- A wearable glove with sensors and a microcontroller
- A Raspberry Pi based "central hub" for processing and communication
- A software framework for integrating with various devices and services
- A user interface for customising actions and settings

The scope of the project includes the following key features:

- Gesture recognition and detection using sensors and machine learning algorithms
- Wireless communication between the glove and the hub
- Integration with various devices and services, such as computers, smartphones, and smart home devices
- Customisable actions and settings using a user-friendly interface
- Modular design and architecture to allow for easy expansion and modification

The scope of the project does not include:

- Development of a commercial-grade product with guaranteed reliability and support
- Integration with critical infrastructure or systems that require high levels of security and reliability
- Development of a user interface for non-technical users

3 Components

3.1 Central Hub

The Central Hub will manage high-level operations using a custom lightweight General OS. The primary responsibilities of this unit include:

- Communicating with connected devices (e.g., computers, smart home devices).
- Receiving and executing action commands sent from the microcontroller, which deciphers gesture data from sensors.
- Triggering appropriate actions (such as automation commands or system control actions).
- Managing configuration settings and providing a user interface for customisation and system feedback.
- Integration with platforms like “Home Assistant” or “IFTTT” to enable smart home automation and expanded service interaction.

The Central Hub receives pre-processed commands from the glove's microcontroller-based Data Acquisition and Processing Unit, and executes them accordingly. This division of labour streamlines the system, reducing the computational load on the Pi and minimising both real-time latency issues and communication overhead.

For this purpose, a Raspberry Pi 3B+ will be utilised, solely for the reason that I already own it, and that it provides a suitable platform for the task.

3.2 Data Acquisition and Processing Unit

This unit, embedded in the glove, is responsible for acquiring data from the sensors and processing and interpreting that data in real time. It will:

- Read data from multiple sensors, such as gesture sensors, IMUs, flex sensors, etc.
- Decipher gesture signals and perform basic signal processing (e.g., filtering, normalisation) to reduce noise and improve accuracy.
- Identify gestures and commands by using custom algorithms, possibly employing basic machine learning models or threshold-based detection.
- Communicate wirelessly with the Central Hub (via MQTT) to transmit processed action commands instead of raw data.
- Manage the system’s power consumption, ensuring efficient energy use through scheduling and sleep states (leveraging low-power modes like deep sleep when idle).

The Raspberry Pi Pico 2 will be used for this purpose, as it offers adequate processing power, efficient sensor interfacing, and a real-time operating system (RTOS) to handle real-time tasks like gesture recognition. The addition of its RISC-V core and improved power management makes it a suitable microcontroller for this role.

3.3 2D Gesture Sensor

A 2D gesture recognition sensor that can detect hand gestures, providing a basic level of gesture recognition. This sensor will be used to detect:

- Hand movements (up, down, left, right, etc.)
- Hand gestures (e.g. waving, pointing, etc.)
- Gesture patterns (e.g. clockwise, counterclockwise, etc.)

PAJ7620U2 sensor would be used for this purpose, since I already own it. It uses a combination of infrared LEDs and photodiodes to detect the movement of a hand or finger, and can recognise gestures based on the movement patterns.

3.4 Flex Sensor

An advance extension to the system, once the basic gesture recognition is developed, the functionality will be expanded with additional flex sensors that will read finger movement, providing a more detailed level of control. More on this later.

3.5 Inertial Measurement Unit

To increase the kind of gestures, expanding upon basic PAJ7620U2 coverage, this sensor will help read the pace, orientation, and movement of the glove, enabling more complex gestures and actions. More on this later.

3.6 Haptic Feedback Motors

The glove will primarily act as an input device to multiple devices, but to make the experience more immersive and interactive, it will consist of a haptic feedback motor, which would be primarily used for alerts, and to confirm if/when the requested action was triggered. This, as well, will be implemented in l

4 Power Management

A more detailed insight is provided in Power Documentation.

4.1 Low-Power Sleep Modes

To ensure CyWear is truly portable, it is essential to optimise power usage. Implementing low-power modes for sensors and microcontrollers when the glove is inactive will help achieve this. The Pico or other microcontrollers can be configured to remain in a deep sleep mode, waking up only when a specific gesture is detected using an interrupt-based system.

The INA219 sensor can be utilised to monitor real-time power consumption, facilitating debugging and optimisation of energy usage.

4.2 Battery Management

A LiPo battery with a battery management system (BMS) should be integrated to handle charging, power distribution, and safe operation. The BMS can also provide battery status data, which can be used to display a low-power warning or disable high-power components when necessary.