**Design: Tensile Tester Control**

**Revisions**

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Author** | **Changes** | **Date** |
| 001 | Tait Richards | Initial Release | 2021-08-12 |
| 002 | Tait Richards | Implementing suggested changes | 2021-08-18 |

**Table of Contents**

[1. References 3](#_Toc96609588)

[1 Purpose 5](#_Toc96609589)

[2 Concept of Operation 5](#_Toc96609590)

[2.1 Use Case 5](#_Toc96609591)

[3 System Design 5](#_Toc96609592)

[3.1 System Architecture 5](#_Toc96609593)

[3.1.1 Hardware Components 6](#_Toc96609594)

[3.1.2 Software Components 7](#_Toc96609595)

[4 High-level Hardware Design 9](#_Toc96609596)

[5 High-level Software/Firmware Design 10](#_Toc96609597)

**Acronyms**

|  |  |
| --- | --- |
| RPi | Raspberry Pi |
| SPR | Steps Per Revolution |
| GPIO | General Purpose Input-Output |
| GUI | Graphical User Interface |
| IP | Internet Protocol |
| PWM | Pulse Width Modulation |

# References

|  |  |
| --- | --- |
| [1] | Raspberry Pi Foundation, "Raspberry Pi 3 B+," 14 March 2017. [Online]. Available: https://static.raspberrypi.org/files/product-briefs/Raspberry-Pi-Model-Bplus-Product-Brief.pdf. [Accessed 17 August 2021]. |
| [2] | Progressive Automations, "PA-17 POT Data Sheet," [Online]. Available: https://cdn.shopify.com/s/files/1/2134/8037/files/PA-17\_POT\_datasheet\_v1.01.pdf. [Accessed 12 August 2021]. |
| [3] | Dimension Engineering, "SyRen 10 / SyRen 25 motor driver user’s guide," July 2007. [Online]. Available: https://www.robotshop.com/media/files/pdf2/dim-06-manual.pdf. [Accessed 12 August 2021]. |
| [4] | Sherline Products, "CNC-Ready Rotary Table," [Online]. Available: https://sherline.com/wp-content/uploads/2015/12/3700cncinst.pdf. [Accessed 12 August 2021]. |
| [5] | Bulk Man 3D, "TB6600 Stepper Motor Driver," [Online]. Available: https://bulkman3d.com/wp-content/uploads/2019/06/TB6600-Stepper-Motor-Driver-BM3D-v1.1.pdf. [Accessed 12 August 2021]. |
| [6] | Omega, "LCHD SERIES Low Profile Load Cells," [Online]. Available: https://assets.omega.com/manuals/test-and-measurement-equipment/load-and-force/load-cells/M4086.pdf. [Accessed 12 August 2021]. |
| [7] | Sparkfun Electronics, "OpenScale Applications and Hookup Guide," [Online]. Available: https://learn.sparkfun.com/tutorials/openscale-applications-and-hookup-guide. [Accessed 12 August 2021]. |
| [8] | "node-red-node-pi-gpio," 22 November 2020. [Online]. Available: https://flows.nodered.org/node/node-red-node-pi-gpio. [Accessed 11 August 2021]. |
| [9] | "node-red-dashboard," 7 July 2021. [Online]. Available: https://flows.nodered.org/node/node-red-dashboard. [Accessed 11 August 2021]. |
| [10] | "pigpiod C Interface," 28 March 2021. [Online]. Available: https://abyz.me.uk/rpi/pigpio/pdif2.html. [Accessed 11 August 2021]. |
| [11] | P. King, "How Do I Set a Static IP Address on Raspberry Pi?," 22 May 2021. [Online]. Available: https://www.makeuseof.com/raspberry-pi-set-static-ip/. [Accessed 18 August 2021]. |

# Purpose

This document describes the design for the tensile tester control system.

# Concept of Operation

The high-level operation of the tensile test frame is illustrated by the following use case.

## Use Case

The use case for this machine is very specific. The system is designed to put material samples under tension to the point of failure, while also being able to rotate the samples while under tension. Furthermore, the samples are being probed using x-rays to analyze and model their internal structure. The rotation is required because to create a 3d profile of the sample x-ray images need to be taken from all angles and combined. This also requires the machine to have an open “window” around the test sample containing no metal that will obstruct the x-ray probe.

# System Design

The Tensile Test Frame control system comprises three major components; the web interface that runs on the Raspberry Pi 3 [1] (RPi) accessed by the user’s computer or other device, the motor control code running on the RPi, and the motion hardware (motors and sensors). Each subsystem is detailed below.

## System Architecture

The control system is based on an RPi, this runs the code that reads and processes the sensor data, instructs the motor drivers on how to move, and accepts inputs from the web interface. Users control the system using a computer connect ed to the RPi using the web interface provided with this system . From there the RPi processes the inputs from the user and instructs the machine to move accordingly. The interactions between the inputs on the web interface and the motion hardware can be seen in Figure 1 below.

Diagram

Description automatically generated

*Figure 1: Control Block Diagram*

### Hardware Components

|  |  |  |
| --- | --- | --- |
| Hardware Component | Purpose | Figure 1 Block |
| Stepper Motors | Rotate sample | Motion Hardware |
| Stepper Motor Driver | Controls motion of stepper motors | Motion Hardware |
| Homing switch | Locates zero position of rotational axis | Motion Hardware |
| DC motor driver | Controls the motion of the linear actuator | Motion Hardware |
| Linear Actuator | Applies tension to the sample | Motion Hardware |
| OpenScale | Reads load cell | Motion Hardware |
| Load cell | Sensor that measures the sample tension | Motion Hardware |
| Power supply | Provides power to motor drivers | Motion Hardware |
| Raspberry Pi | Runs program code and hosts web interface | Node-Red web interface |
| User’s Computer | Interacts with web interface | Node-Red web interface |

Table 1: System Hardware Components

The main hardware components are the RPi, the user’s computer, and the motion hardware. While the user’s computer can be any computer on the same network as the RPi, the other hardware components require more in-depth explanation. The motion hardware is composed of two major sections, drivers / interface hardware, and motors / sensors. The motors and sensors are what are interacting with the machine, where as the interface hardware allows the RPi to communicate with and control the motors and sensors.

The system is composed of three motors and two sensors along with their respective interface devices. One motor controls the linear actuator [2] which applies tension to the sample. This motor is driven using a DC motor driver [3] which is controlled by the RPi using PWM for analog voltage (there is a smoothing capacitor and resistor to prevent jerky motion). The other two motors [4] are attached to rotary tables that axially rotate the sample . The motors are originally configured as 400 SPR (steps per revolution) unipolar (six-wire) motors. However, they have been modified for use as 200 SPR bipolar (4-wire) steppers that are half-stepped to 400 SPR A motor mounted gear head provides 72:1 gear reduction giving us 28800 SPR for the rotational assembly. The two rotary tables sit on opposite ends of the tensile tester and provide a coordinated dual drive to prevent torsion on the sample. The rotary tables are driven in opposite directions so that they rotate the sample the same way. Th e two motors must be synchronized to avoid sample torsion. They use the same driver [5] as that guarantees they are always synchronized. Motor direction is flipped by swapping the connections of one of the motor coils.

The two sensors are the microswitch used to zero the rotational axis, and the load cell [6] used to measure the tension on the sample. The microswitch is connected directly to the RPi via GPIO. The load cell cannot be read directly by the RPi as the GPIO (general purpose input-output) pins cannot read its analog voltage signal, which is ~30mV and must be amplified to be effectively read. Because of these limitations we use an interface board known as the OpenScale [7] to read the values of the load cell.

### Software Components

The major software components are the Node-Red code controlling the web interface and performing much of the logic, the C programs used to interface the Node-Red code with the motor drivers, and the Linux operating system the RPi runs.

Node-Red is JavaScript based and supports flow-based programming where nodes are connected into a logic “flow”. The RPi is programmed using a web interface accessed on <IP\_OF\_RPI>:1880 where <IP\_OF\_RPI> is the IP address of the RPi on the network. The Node-Red code utilizes modules (notably node-red-node-pi-gpio [8] and node-red-dashboard [9]) that allow for the creation of a GUI (graphical user interface) and access to the GPIO pin on the RPi. The Node-Red code keeps track of the location of the rotational axis and stores the current position (in case of power loss) in a file located at /home/pi/ tensile-tester/log/position on the RPi’s filesystem. The system also allows for data logging where, on command, the system will record the position of the axis and the tension in the sample multiple times per second. These files are stored in /home/pi/tensile-tester/log/<USER\_CHOSEN>.csv The Node-Red code reads the tension values from the OpenScale over 8n1 9600 baud on serial port /dev/ttyUSB0. The Node-Red code runs locally on the RPi and is accessed over the network by the user’s computer for control.

The C code, which implements the GPIO interfaces and manages their signaling, is run locally on the RPi. It provides precise control of the GPIO pins required to operate the stepper motors smoothly. The C programs utilize the pigpiod C library [10] which is what offers the precise timing of the GPIO pins. If changes to the code are required, this library must be included in the compilation process. The compiled programs and source code are in /home/pi/tensile-tester/src on the RPi’s filesystem. The programs, their functions, and input parameters are listed in table 2 below. The Node-Red code runs the C programs using bash commands which provide the parameters required by the C programs.

|  |  |  |
| --- | --- | --- |
| Program | Function | Input Parameters |
| moves.c | General motion of the stepper motors moving the motors a specific number of steps while ramping up to the maximum rotational speed. | Number of steps, motor driver pin number |
| zero.c | Resets the zero of the rotational axis, this is done by rotating the axis until a microswitch is pressed, causing the current position to be set as zero. | PWM frequency |
| pwm.c | Creates a PWM signal to be sent to any GPIO pin on the RPi; it is used by the Node-Red code to create an analog voltage to send to the DC motor driver. | GPIO pin number |

Table 2: C Programs

The final component of the software is the Linux operating system on the RPi. The RPi is running standard Raspbian 10 Buster . Node-Red must be installed and requires its daemon to be running for the web interfaces to be active. The pigpiod C library also requires a daemon to be running for the programs to work properly. Both daemons are configured to run on startup. The final important aspect of the operating system is the network access. The RPi must be configured to connect to the same network as the user computer. This could be accomplished using Wi-Fi or Ethernet. In both cases a static IP address [11]would be preferable so that the RPi has the same address on every startup. This configuration will likely have to be done on site for each network.

# High-level Hardware Design

The RPi is connected to the other hardware components in three different ways, see figure 2 below. The stepper motor drivers are connected using GPIO 17 for the step signal, GPIO 22 for the direction signal, and GPIO 27 for the enable signal. The DC motor driver is controlled using PWM over GPIO 20 . The homing switch is read from GPIO 26. The motor drivers are also hooked up to the 24v power supply. The OpenScale [7]is connected to the RPi on the serial port /dev/ttyUSB0 using 8n1 9600 baud serial. The user’s computer is connected through the RPi’s web interface found on “<IP\_OF\_RPI>:1880/ui” where the “<IP\_OF\_RPI>” is whatever IP it receives on the network [12] (though it will request a static IP at 192.168.0.200).

Chart

Description automatically generated with medium confidence

Figure 2: Main Power and Signal Connections

# High-level Software/Firmware Design

The RPi runs Raspbian Linux which is used as the base for the other software components, being the Node-Red code and the C motor control programs, see figure 3 below. All the code is run locally on the RPi, though, the GUI is accessible through a web interface on any other computer in the network. The C programs and the Node-Red code interact through the bash terminal provided by Linux. The Node-Red code performs much of the functionality, it takes input from the GUI and load cell and processes that data to instruct the C code how to move the motors. It tracks the position of the rotational axis and will remember where it was left. The C code essentially only provides the necessary signals to control the motor drivers as requested by the Node-Red code.

Diagram

Description automatically generated

Figure 3: Software Hierarchy