Irrigation System µPiHAT

ANTNOA001, CNNBEN002, RYXCAM002

Due: 11 June 2021

Executive Summary

This report encompasses the design and development of an irrigation µPiHAT-compliant PCB, that is used for the irrigation of small gardens/vegetable patches via soil moisture level sensing and water dispensing. The µPiHAT was designed with plant-lovers, scientists, and pool/pond-owners as the diverse array of intended users. Of note is that this device is designed to be portable and cost-efficient.

The report introduces and outlines the use case and functioning of the irrigation PiHAT, and thereafter provides block diagrams and a more detailed explanation of the design process. Thereafter, the report submodules – namely, the Power Supply, Status LEDs, and Amplifier submodules - and their respective specifications and functioning are defined.

A reference manual to the PiHAT's PCB is provided, and all final PCB design files are available in a GitHub repository. The report culminates with a conclusion and reflection of the design process.

Table of Contents

Table of Figures

Introduction

The irrigation PiHAT can be used for the irrigation of small gardens/vegetable patches via soil moisture level sensing and water dispensing. It may also be used in lab environments where scientists need to hold the moisture level of the soil at a constant level. An optional configuration, which shows the modular ability of the HAT, is to replace the moisture sensors with a float switch or humidity sensor to either fill up pools; ponds; or water tanks or to operate ventilation systems if an enclosed space reaches a maximum humidity level. The PiHAT will allow more advanced users to continuously perform measurements or monitor the soil moisture levels by sending its input from the sensors into the main Pi GPIO header.

The moisture sensors provide analogue signals to the PiHAT via input pins. Watering levels are controlled by the user. Solenoid valves are connected to the PiHAT for water distribution. The higher voltage power supply provides power to the PiHAT separately from the main Pi board as the valves draw far more current and voltage than the Pi can supply. Each moisture sensor will have an LED level indicator to show soil moisture levels.

Project Subsystems Block Diagram

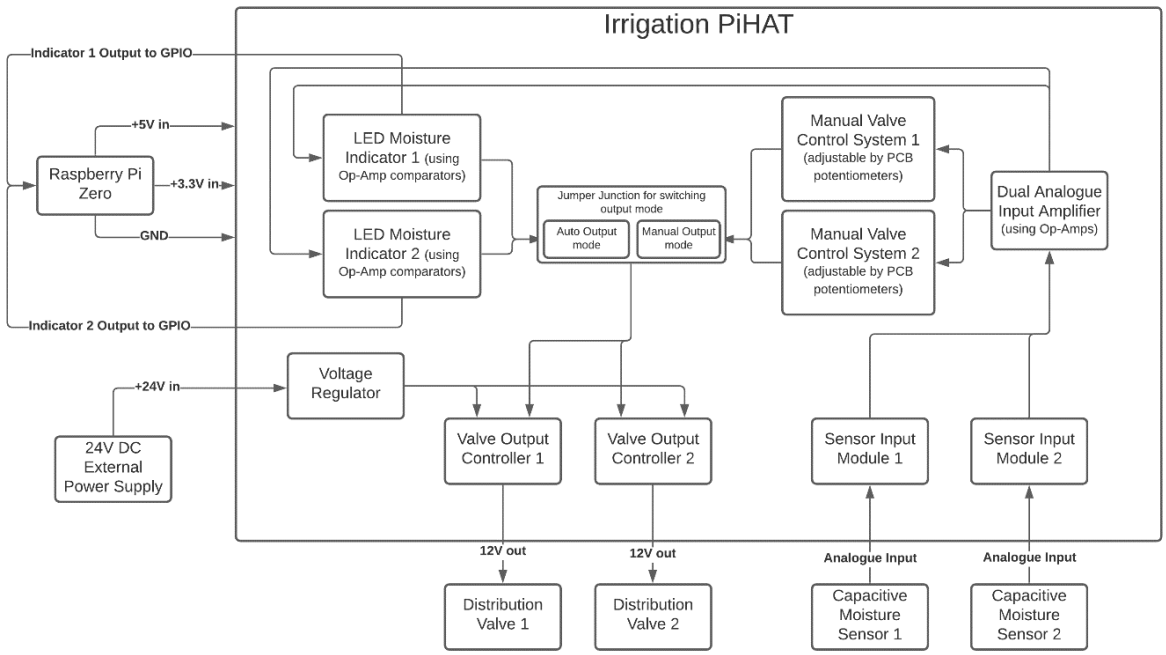

Figure 1 – Complete Irrigation System µPiHAT Block Diagram. The diagram shows all interfacing between submodules.

Design process

Each section of the process was divided up amongst each team member in a fair weighting to ensure that a final product could be produced in the most time efficient manner. Each team member worked on a section of the design process that they had a high proficiency in. Initially, the project topic was decided on as a team after some research was completed. The choice was unanimous in the Irrigation System µPiHAT as it proved to be an especially useful application of the Raspberry Pi for hobbyists and researchers.

The next stage was to define the possible requirements a client could provide in this context. These requirements are the high-level thought processes from the client's perspective, showing the expected or desired outcomes of the system's design. From these requirements, more detailed specifications were obtained. The full list of Requirements and Specifications of the design can be found here.

Almost immediately after the requirements and specifications were finalized, a set of rough schematics were drawn up and marked as the V0.1 schematics. Initially, a linear voltage regulator was used and has since been replaced in the V1.0 schematics onward by the LTC2895 Switching Step-Down Voltage Regulator. The final **Amplifier** submodule schematic can be found here. While the **LED and Valve Control** subsystem schematic can be found here. Finally, the **Power Supply** submodule is linked here. All three subsystems were combined to form a **Final System Design** in KiCAD and the schematic of this is found here.

Simulations were performed in LTSpice shortly after the final schematics were checked. The simulations were a key point in the design process to ensure each submodule worked fully. The collection of LTSpice simulation files can be found here.

Lastly, an overall PCB was generated in the KiCAD tool. It was decided that the standard Raspberry Pi Zero HAT template built into KiCAD would be used. The raw PCB file can be found here. There is no pdf version generated for this file and all Gerber files have been added to the gitignore as they are generated files. All dependencies and external libraries were included in the GitHub version control.

Power Supply Submodule

The package chose is the LTC3895 150V Low IQ, Synchronous Step-Down DC/DC Controller. This regulator in its typical application structure provides a highly efficient and power saving component. It requires several external resistors, capacitors, inductors and MOSFETs in its typical form. External cooling systems, such as heat-sinks, are not required for this IC as it is very efficient and produces extraordinarily little heat unlike a linear regulator.

The output voltage will be a constant 12V DC which will be sent to the Amplifier subsystems to be used in the powering of the solenoid valves. The voltage out does have a small delay time which is negligible as it is less than 1 millisecond. This is due to the internal configuration of the LTC3895 IC that will be used.

The switching voltage regulator can provide a stable 12V DC output even if the input voltage drops to 15V or exceeds 48V for example. The lowest rated voltage the switching regulator can accept is 7V and the highest rated voltage the switching regulator can accept is 140V, although both limits are not recommended to be exceeded.

The maximum current produced by the regulator is 5A at the V_{out} terminal. This can be adjusted by putting an appropriately chosen resister in parallel with the load to prevent high current draw straight to the load. This 5A threshold suits the project well for expansion as an individual solenoid valves draw a maximum of 710mA during operation and the combined current draw in parallel is 1.4A. This is much lower than the 5A rating and allows for future output expansion.

The link below demonstrates how the power module should be connected to the mains and how to troubleshoot the connection if necessary. GitHub Link for the power submodule HOW-TO file: https://github.com/ryxcam002/IrrigationPiHat/blob/master/How-Tos/DesignProcess.md#power-module

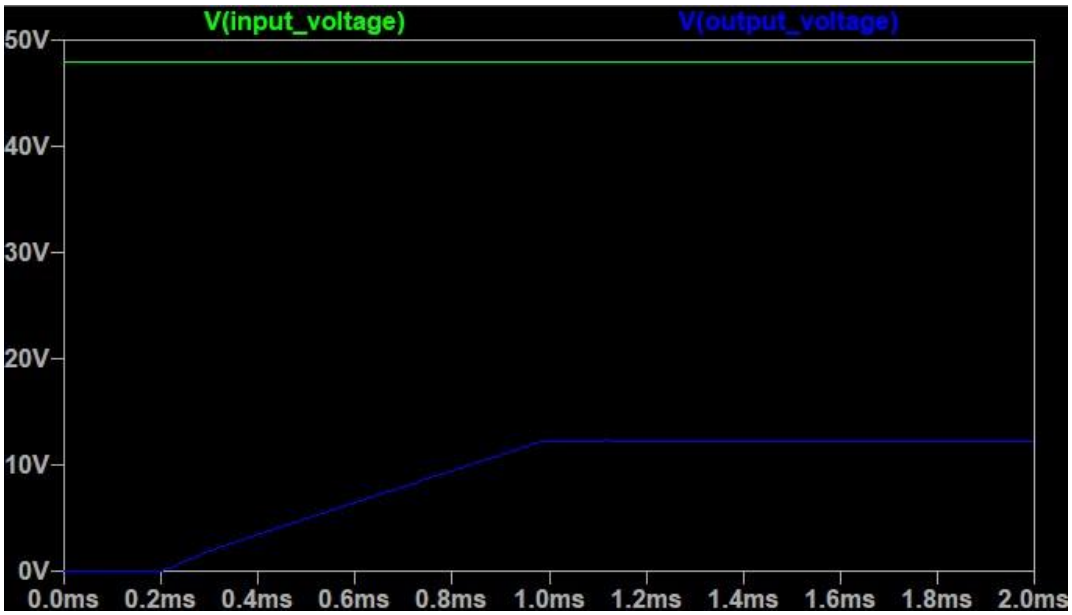

Figure 2 – High end protection demonstration of the switching voltage regulator. Input voltage shown at 48V and output is constant at 12V.

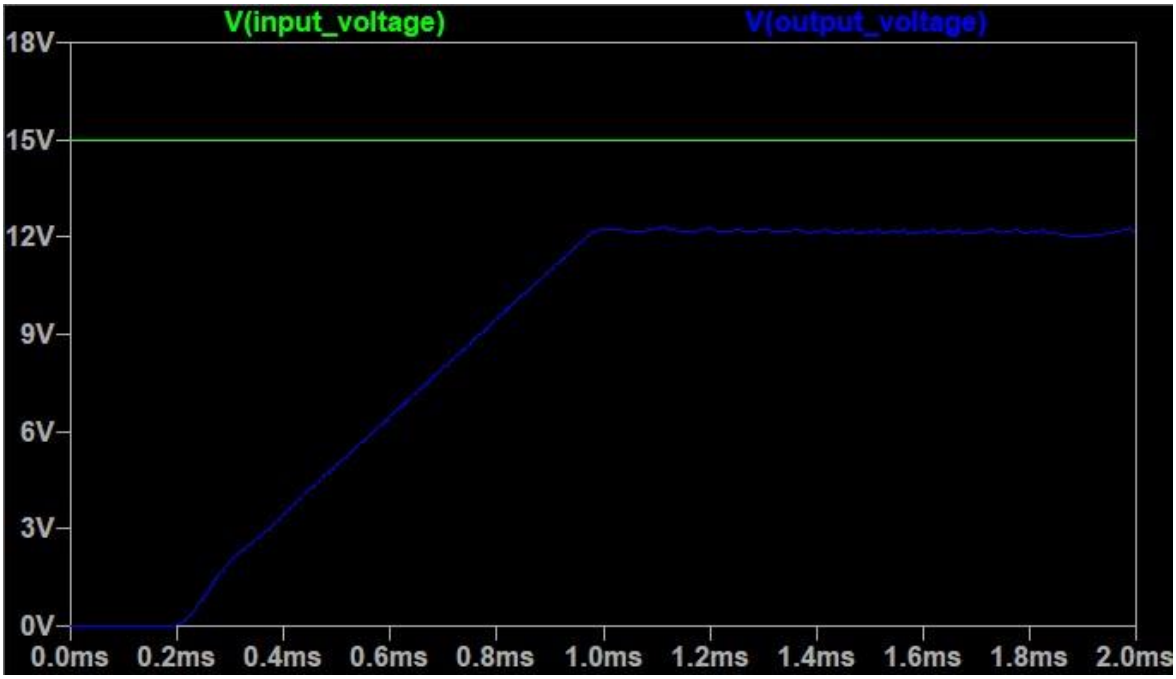

Figure 3 - Low end protection demonstration of the switching voltage regulator. Input voltage shown at 15V and output is constant at 12V.

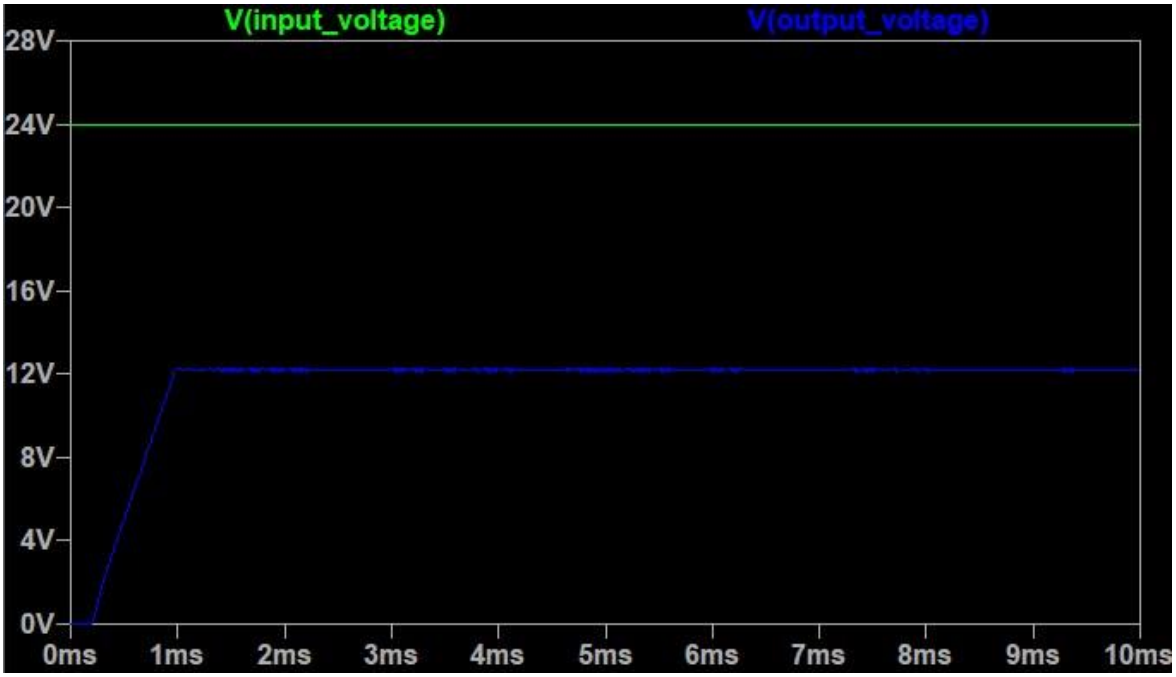

Figure 4 – Simulation of the switching voltage regulator at normal or expected operation. Input voltage shown at 24V and output is constant at 12V.

GitHub link to the LTSpice simulation file:
https://github.com/ryxcam002/IrrigationPiHat/blob/master/Simulation%20Files/PowerSim/Power%20Submodule.asc

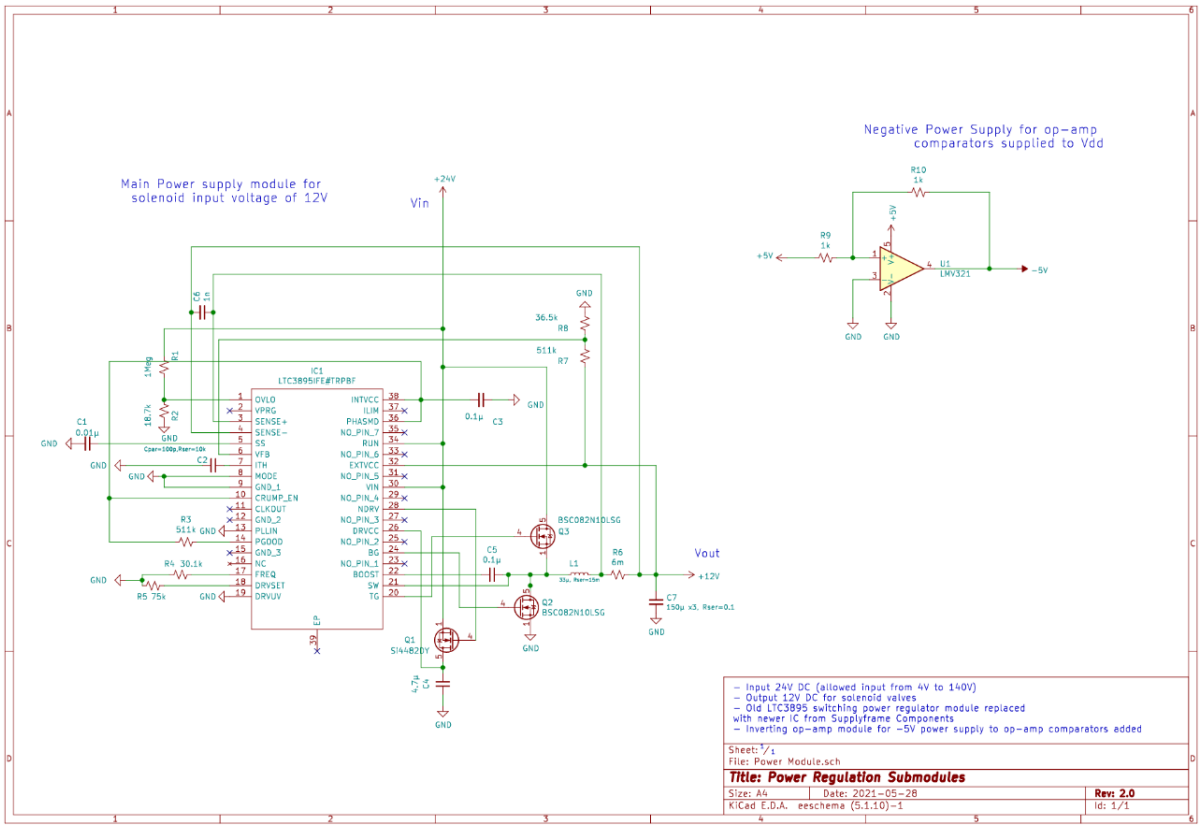

Figure 5 - KiCAD schematic for the Power Submodule with added -5V supply for other submodules.

GitHub link to the schematic show above:
https://github.com/ryxcam002/IrrigationPiHat/blob/master/Individual%20Module%20Schematics/PowerSubmoduleV2.0.pdf

Overall, this design is much safer than the initial linear voltage regulator chosen in the V0.1 schematic as it dissipates much less heat and has built-in over power protection as stated in the datasheet. The typical application circuit in the LTC3895 datasheet was perfect for the design as it provided the exact voltage and current specifications that were desired. The circuit was replicated from the datasheet and thus has no existing bugs as that circuit design has been tested and verified by the datasheet/IC package creators. Initially the linear regulator would have been marked as an issue as it was highly unsafe and took up a large amount of space and had a severe disadvantage of needing a heatsink, due to its high inefficiency and power dissipation specifications, which also takes up a large amount of space. The LTC3895 is an excellent mechanical design choice as it is overall IC footprint is also small and has many unconnected or no connection pins providing easy, uncluttered routing of copper tracks.

Status LEDs Submodule

Top-level Specifications:

1. Op-amp V_{cc}: 5V
2. Op-amp V_{ss}: 0V
3. Op-amp Control $V_{control}$: 3.3V
4. Signal V_{input}: 0V to 3.3V
5. V_{output}: 1V
6. I_{max}: 71 mA
7. ICs used: LMV324, 74HC86
8. IC Package: SOIC-14
9. Cooling Requirements: None

The inputs to the comparator subsystems come in the range of 0V to 3.3V as an analogue input signal. This signal is received from the amplifier subsystem which comes from the soil moisture sensors. When each comparator's input voltage exceeds the set threshold voltage, the LED's sequentially turn on. There are 3 LEDs to represent the different stages of soil moisture: red (dry), yellow (somewhat moist), green (very moist). All of the comparator's 'trigger' levels are manually adjustable to allow the user to manually set their desired moisture levels for the different LED's.

The output of the LED's comparator system then feed into a BJT which would then control the solenoid valve. This has been set up, using logic gates, to ensure that a signal will only be outputted when the moisture level is between the yellow and red LED. This means that the valve will be automatically turned on and allow water to flow when the soil moisture level starts to get dry and will turn off once an adequate soil moisture level has been reached.

The outputs of the comparators additionally enter the Raspberry Pi to provide optional data for users to read in. These outputs will vary between a high or low state as outputted by the Op-Amps. Essentially, the Raspberry Pi will read the LED state (high/low).

There is an also a switch present between the LED and valve control circuit, as a secondary means to manually control the valve control values. This is for those users who want to have different LED trigger levels from solenoid control trigger levels.

GitHub Link for the LED HOW-TO file:
https://github.com/ryxcam002/IrrigationPiHat/blob/master/How-Tos/DesignProcess.md#led-module

Figure 6 – Running Simulations of LED and Valve Control subsystem of Irrigation PiHAT

LED circuit simulation plot displaying:

1. A simulated input Voltage V(in) ranging from 0V (very moist soil) to 3,3V (very dry soil).
2. The adjustable trigger Voltages of the comparators V(red), V(yellow) and V(green) for the respective coloured LEDs, indicating at what input voltage these comparators turn on (consecutively).
3. An additional potentiometer-dependent voltage V(control) that can be adjusted by the PiHat user, to manually control the valves and bypass the soil moisture sensors.

GitHub Link for the simulation file:
https://github.com/ryxcam002/IrrigationPiHat/blob/master/Simulation%20Files/LED%26ValveControlSim/Led%20%26%20Valve%20Submodule.asc

Figure 7 - Running Simulations of Secondary Valve Control subsystem of Irrigation PiHAT

Valve control circuit simulation plot displaying:

1. A simulated input Voltage V(in) ranging from 0V (very moist soil) to 3,3V (very dry soil).
2. The adjustable trigger Voltages of the comparators V(dry) and V(moist), in the case that the user does not want to use the LED subsystem to control the solenoid valve.
3. An additional potentiometer-dependent voltage V(control2) that can be adjusted by the PiHat user, to manually control the valves and bypass the soil moisture sensors.

GitHub Link for the simulation file:
https://github.com/ryxcam002/IrrigationPiHat/blob/master/Simulation%20Files/LED%26ValveControlSim/Led%20%26%20Valve%20Submodule.asc

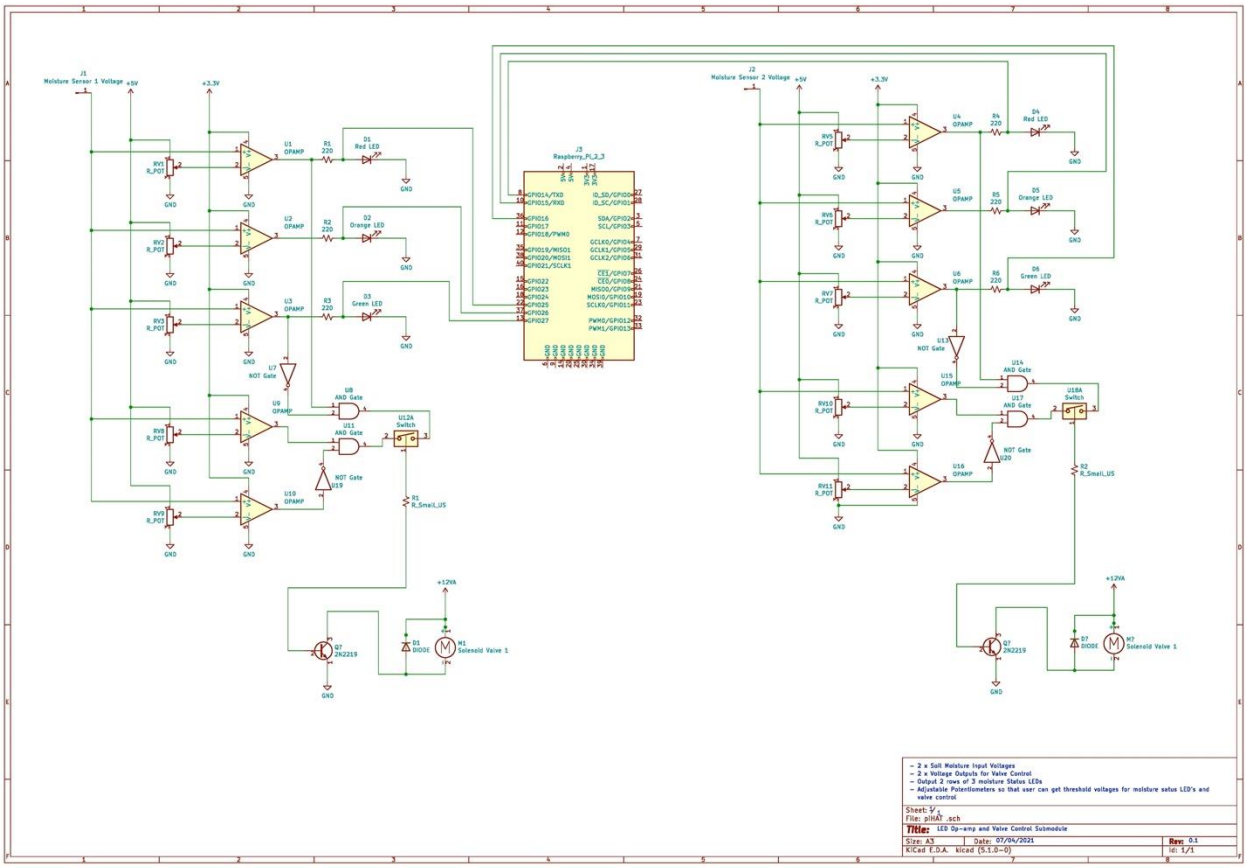

Figure 8 - KiCAD schematic for the LED and Valve Control Submodule

GitHub Link for the above schematic:
https://github.com/ryxcam002/IrrigationPiHat/blob/master/Individual%20Module%20Schematics/LedSubmoduleV2.0.pdf

The design is sufficient as the simulated circuit meets the specifications for this subsystem, as follows:
1: Three comparator circuits present, with each threshold voltage controlled by a potentiometer.
2: There are two circuits present to determine the valve control system, namely
two additional comparators for when the valve turns off and on, as well as logic gates to ensure the valves turn on at the required values.

In conclusion for this submodule, the design is safe as there are backup systems (in the form of having two choices to control the solenoid valve) to use in case one of them fails. The simulations behave exactly as we wanted them to and produces an expected output. There are no bugs or issues. We have also made a complete copy of the LED submodule; this would allow for feedback for two inputs and allow independent control over two separate watering systems. This also acts like a backup system and would allow the user to continue to use one of the LED and valve control systems in case the other one fails. This, in our opinion, is in line with good design practice.

Amplifier Submodule

Top-level Specifications:

1. Op-amp V_{cc}: 5V
2. Op-amp V_{ss}: -5V
3. Signal V_{input}: 1.5V to 2.5V
4. V_{output}: 0V to 3.3V
5. I_{max}: 2.5 mA
6. ICs used: LMV324
7. IC Package: SOIC-14
8. Cooling Requirements: None

The op-amp submodule is made up of 3 stages, with the final stages providing the amplified output that will interface with the LED comparators and manual comparator subsystems. The input to the amplifier subsystem comes in the form of an analogue signal from the moisture sensors, in the range of 1.2V to 2.5V. The output of the subsystem involves an adjusted analogue signal which is scaled to be between 0V and 3.3V to match the Raspberry Pi's output rails.

The LMV324 op amp was chosen for this circuit as it is a quad op amp, which means it is a 4-in-1 integrated circuit, and therefore is more space-efficient on a PCB. Additionally, this op amp is a Surface-Mount Device, further saving space on the PCB. The input signal was not changing very quickly with respect to time, therefore, an op amp with a high slew rate was not need; instead, we decided to go with a low power op amp so that it would consume less power, requiring less electricity to be used and is therefore more environmentally friendly.

GitHub Link for the amplifier submodule HOW-TO file:
https://github.com/ryxcam002/IrrigationPiHat/blob/master/How-Tos/DesignProcess.md#amplifier-module

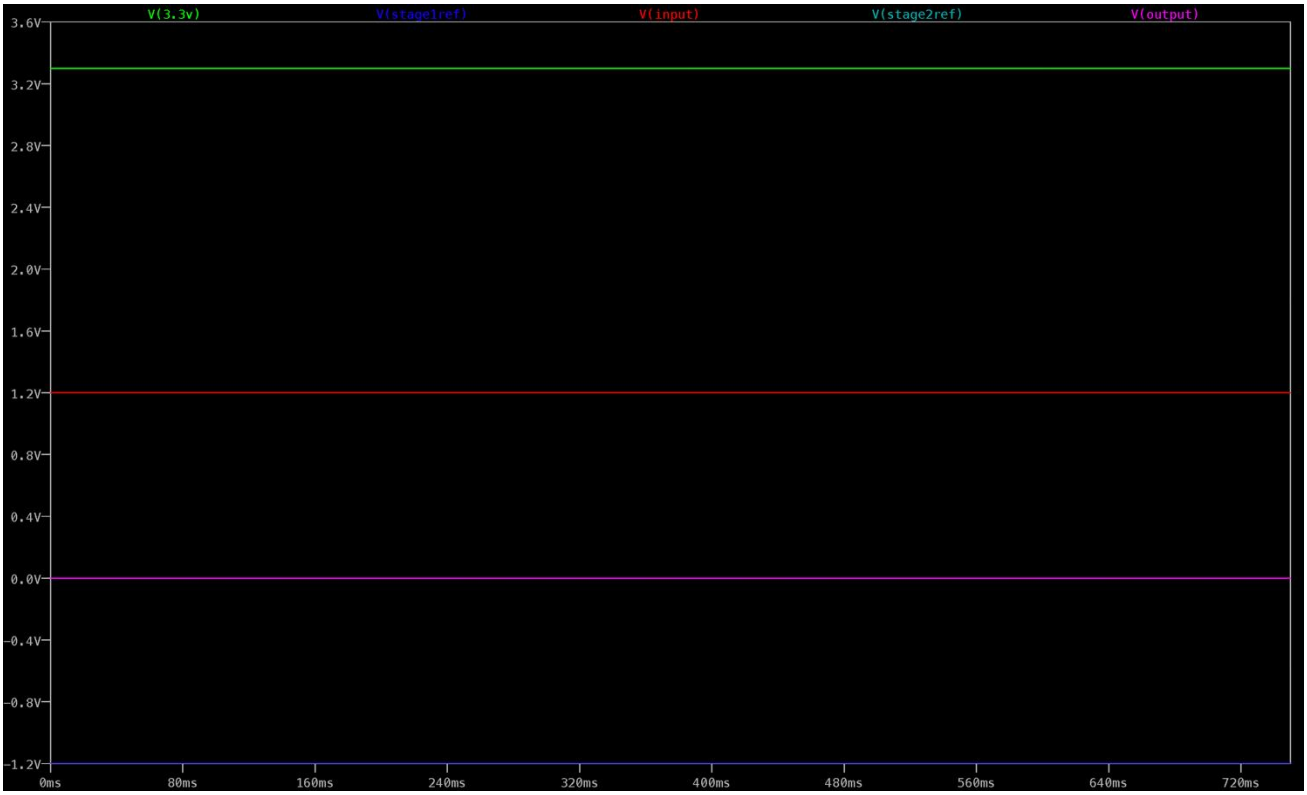

Figure 9 - Running Simulations of Amplifier subsystem of Irrigation PiHAT for the Input Signal's Lower Bound

Amplifier circuit simulation plot displaying:

1. A simulated input Voltage V(input) of 1.2V (very moist soil).
2. The output of the first stage (Vstage1ref), showing the 3.3V from the Pi being amplified to -1.2V for use in the second stage.
3. The output of the second stage (Vstage2ref), showing the inverted output of the -1.2V (from the previous stage) being added to the input voltage.
4. The output of the system V(output).

GitHub Link for the simulation file:
https://github.com/ryxcam002/IrrigationPiHat/blob/master/Simulation%20Files/AmplifierSim/Amplifier%20Submodule.asc

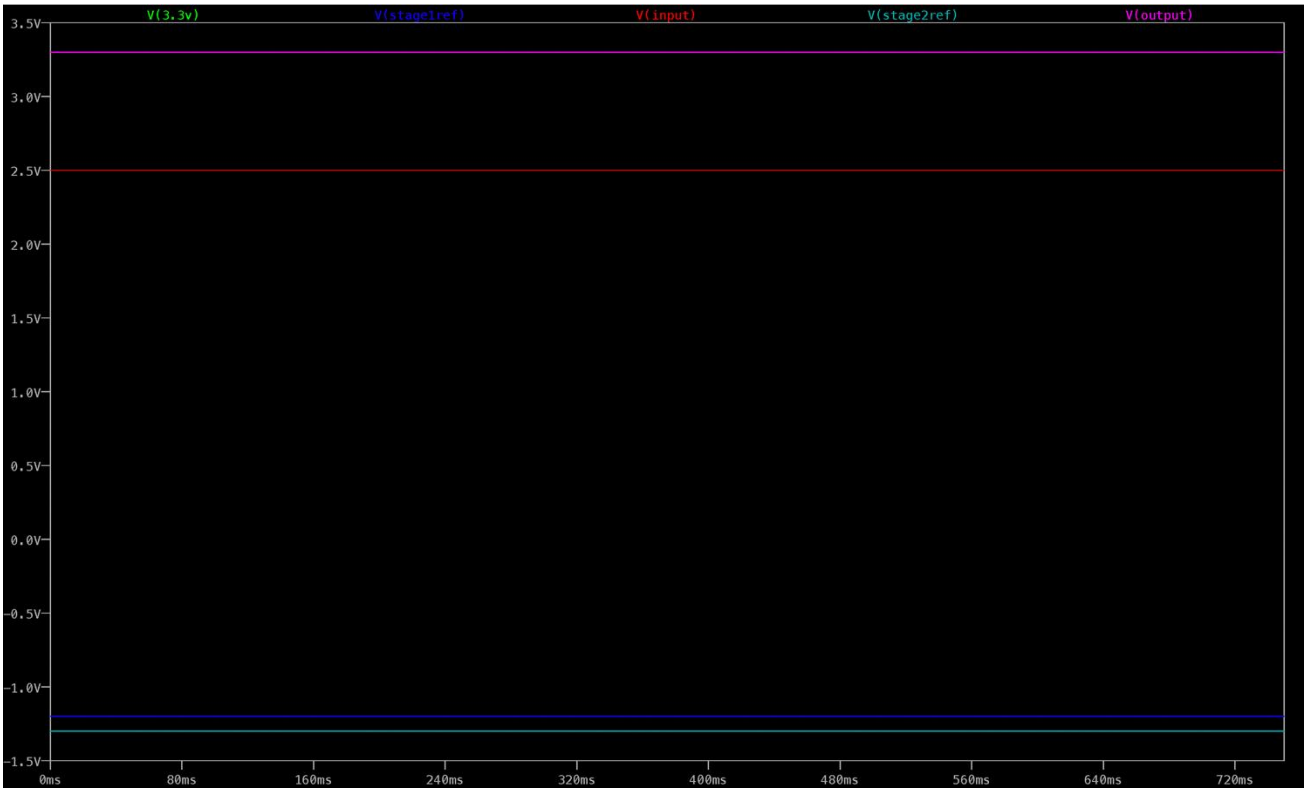

Figure 10 – Running Simulations of Amplifier subsystem of Irrigation PiHAT for the Input Signal's Upper Bound

Amplifier circuit simulation plot displaying:

1. A simulated input Voltage V(input) of 2.5V (very moist soil).
2. The output of the first stage (Vstage1ref), showing the 3.3V from the Pi being amplified to -1.2V for use in the second stage.
3. The output of the second stage (Vstage2ref), showing the inverted output of the -1.2V (from the previous stage) being added to the input voltage.
4. The output of the system V(output).

GitHub Link for the simulation file:
https://github.com/ryxcam002/IrrigationPiHat/blob/master/Simulation%20Files/AmplifierSim/Amplifier%20Submodule.asc

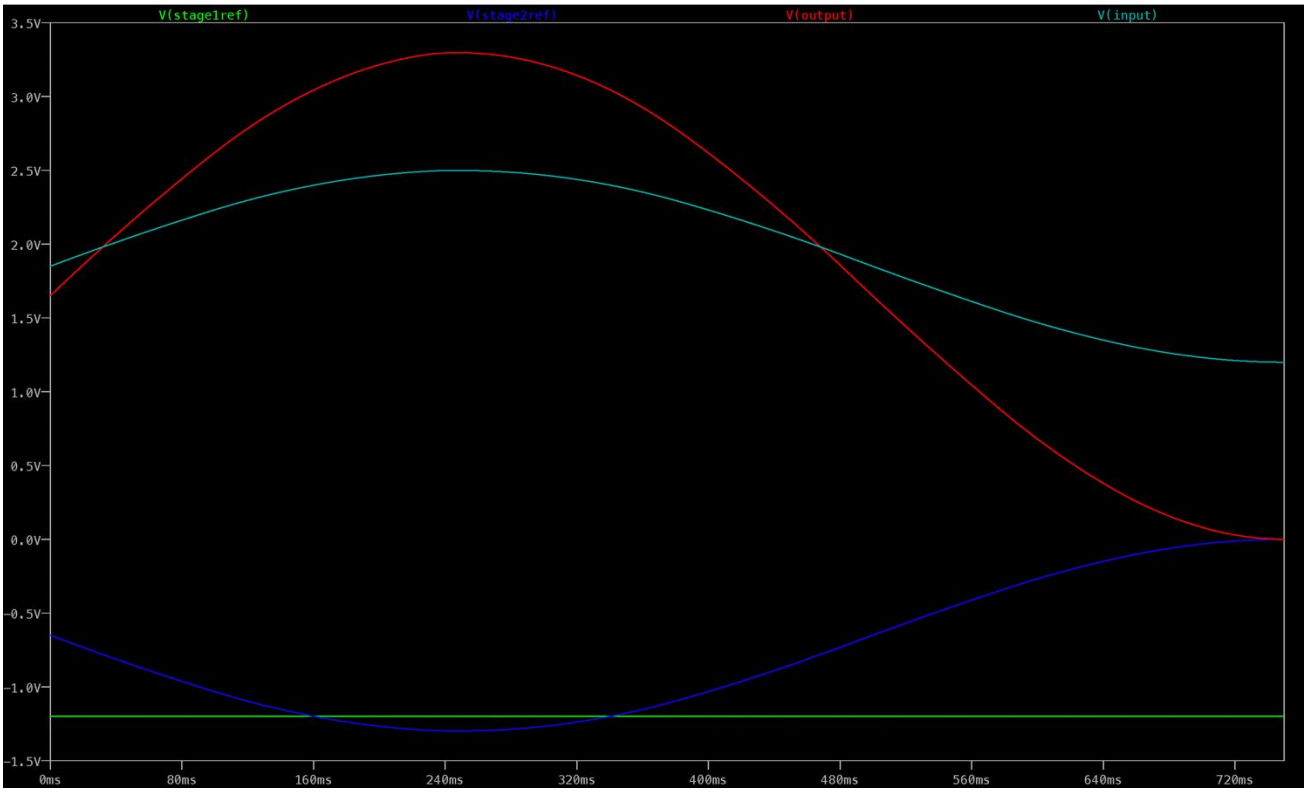

Figure 11 – Running Simulations of Amplifier subsystem of Irrigation PiHAT for a Varying Input Signal

Amplifier circuit simulation plot displaying:

1. A simulated varying input Voltage V(input).
2. The output of the first stage (Vstage1ref), showing the 3.3V from the Pi being amplified to -1.2V for use in the second stage.
3. The output of the second stage (Vstage2ref), showing the inverted output of the -1.2V (from the previous stage) being added to the inverted input voltage.
4. The output of the system V(output), ranging from 0V to 3.3V as required.

GitHub Link for the simulation file:
https://github.com/ryxcam002/IrrigationPiHat/blob/master/Simulation%20Files/AmplifierSim/Amplifier%20Submodule.asc

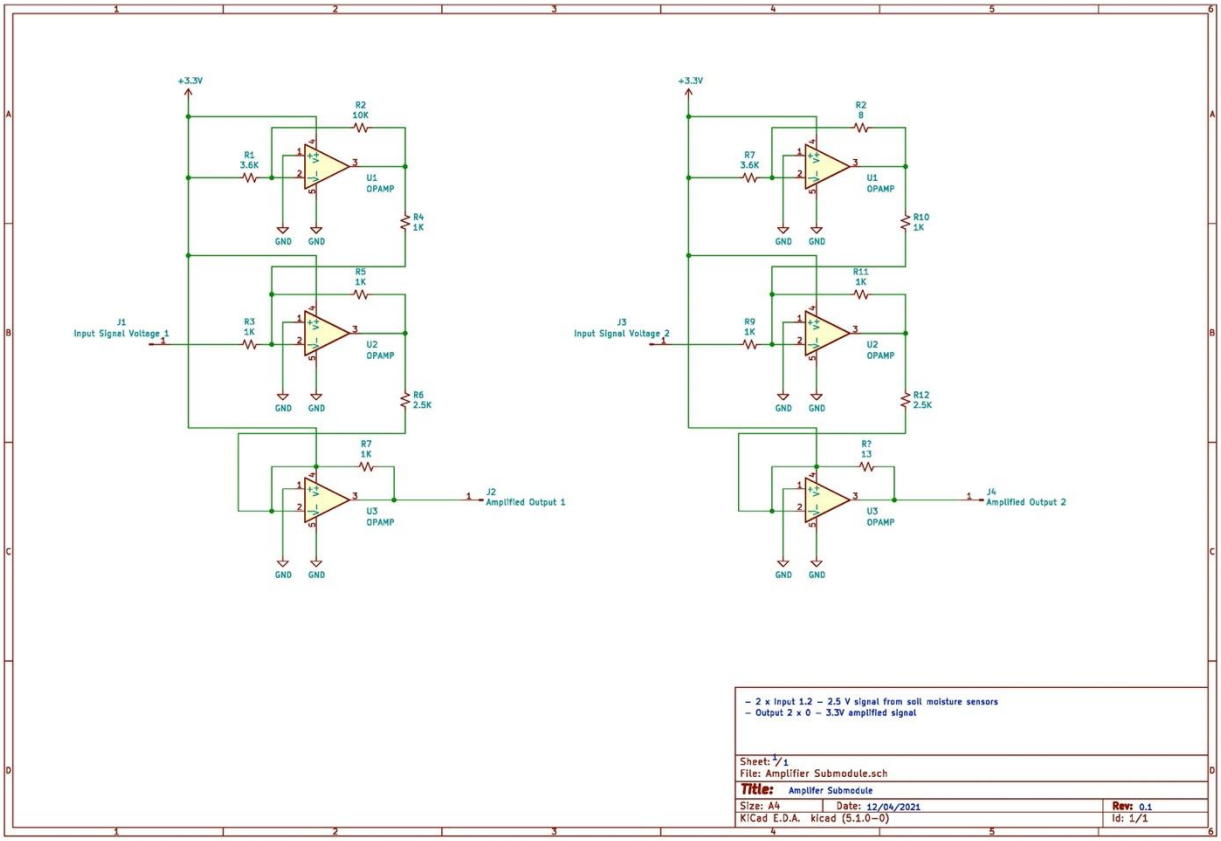

Figure 12 - KiCAD schematic for the Amplifier Submodule

GitHub Link for the above schematic:
https://github.com/ryxcam002/IrrigationPiHat/blob/master/Individual%20Module%20Schematics/AmplifierSubmoduleV2.0.pdf

In conclusion for this submodule, the design is sufficient as it meets the specification of amplifying the input signal to between 0V and 3.3V for the Pi to use. The simulations behave exactly as we wanted them to and produces an expected output. There are no bugs or issues. We have also made a complete copy of the amplifier submodule; this would allow for amplification for two inputs from two soil moisture sensors. This also acts like a backup system and would allow the user to continue to use one amplification system to amplify one input signal in case the other one fails. This, in our opinion, is in line with good design practice.

PCB

Technical Specifications:

1. No. Copper Layers: 6
2. ICs used: LMV324, 74HC76, LMV321
3. IC Packages: SOIC-14
4. Cooling Requirements: None
5. Via Diameter: 0.2 mm
6. PCB Track Width: 0.1 mm

GitHub link to the manufacturing notes:
https://github.com/ryxcam002/IrrigationPiHat/blob/master/How-Tos/ManufacturingNotes.md

GitHub link to the bill of materials:
https://github.com/ryxcam002/IrrigationPiHat/blob/master/Combined%20Schematic%20%26%20PCB/BillOfMaterials.md

GitHub link to the getting started guide:
https://github.com/ryxcam002/IrrigationPiHat#getting-started

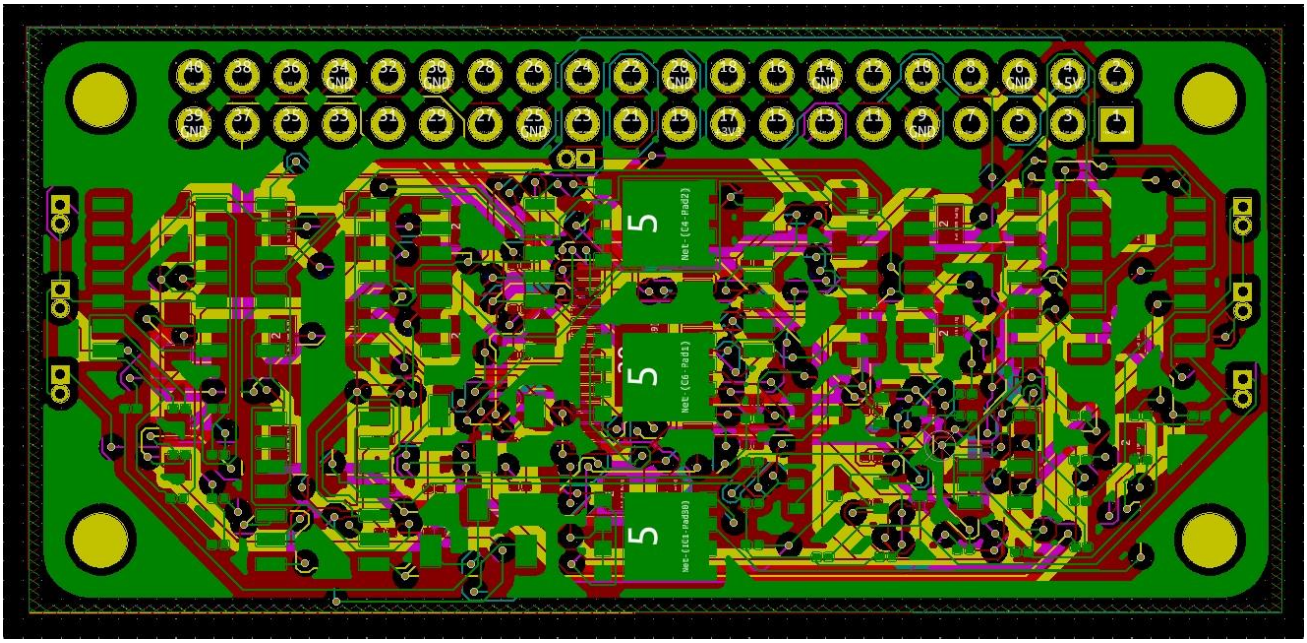

Figure 13 - KiCAD PCB schematic view of the back copper side

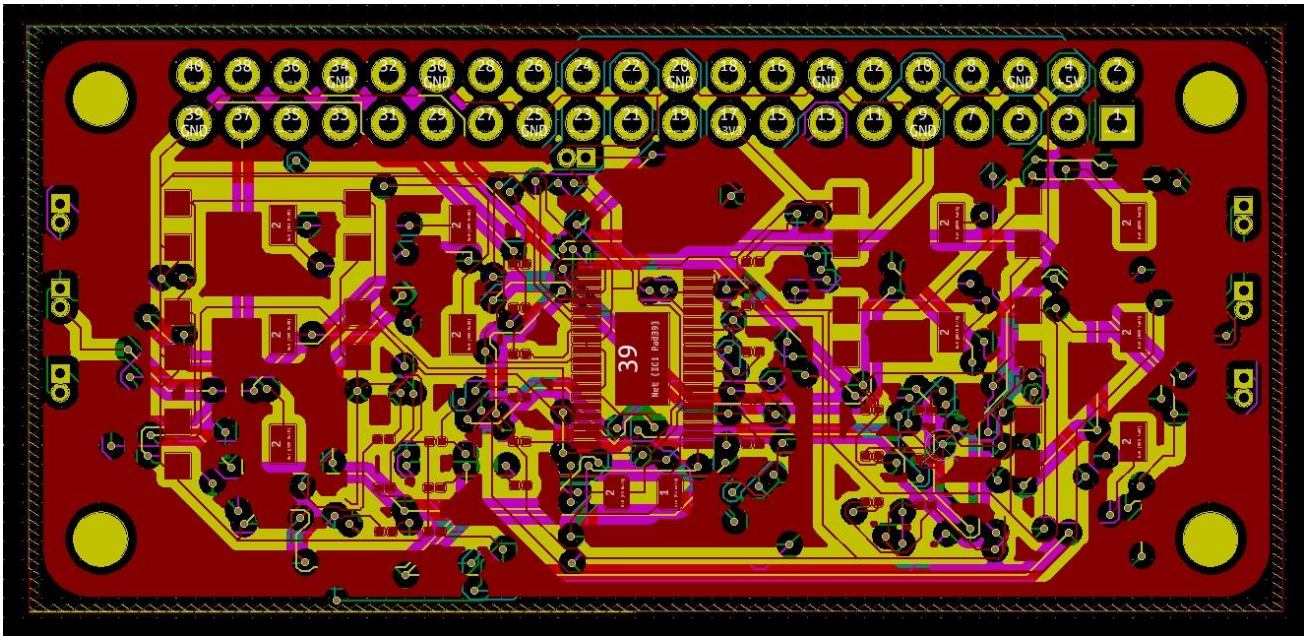

Figure 14 – KiCAD PCB schematic view of the front copper side

Conclusion

The irrigation PiHAT satisfies the project design requirements. From assessment of the design, the PiHAT could be feasibly prototyped and produced.

The functionality of the irrigation PiHAT is extensive – from irrigation of small gardens/vegetable patches, to use in lab environments, to the replenishment of water sources - the PiHAT is adaptive and has modular abilities.

Furthermore, the PiHAT allows more advanced users to continuously perform measurements or monitor the soil moisture levels by sending its input from the sensors into the main Pi GPIO header.

The PiHAT design comprises block diagrams, LTspice simulations, and PCB designs. This extensive documentation allows any user to gain a full understanding of its functioning and usage.

The irrigation PiHAT is a nifty device that is ready to make waves on the electronics market. Overall, the lessons learnt, and skills gained by the team in developing this irrigation system are the most valuable entity to come out of this PiHAT's creation.

Reflection on Design Process

The use of LTspice proved a useful supplemental aid to circuit design and optimization. It also allowed the group to gain more experience in using the software, as well as seeing the practicalities (non-ideal cases) of certain components and circuitry.

The use of GitHub was helpful as a means of becoming familiar and confident with the software. Practically, its usage better serves as a stepping-stone for future projects of bigger, more dynamic scale, as this project only comprised 3 members in constant communication – therefore the assistance of GitHub's version control and collaborative tools was not wholly necessary.

The use of block diagrams assisted in dividing the project up into workable and logical methods, thereby making the task less overwhelming. It also served as a helpful precursor to the Spice implementations.

The fortnightly design review meetings were helpful in that they provided the group a milestone for deadlines and keeping up to date with the submissions. It is recommended that these submissions, however, be optional, as in many meetings the group had to attend an hour-long session but did not have any queries/ issues.