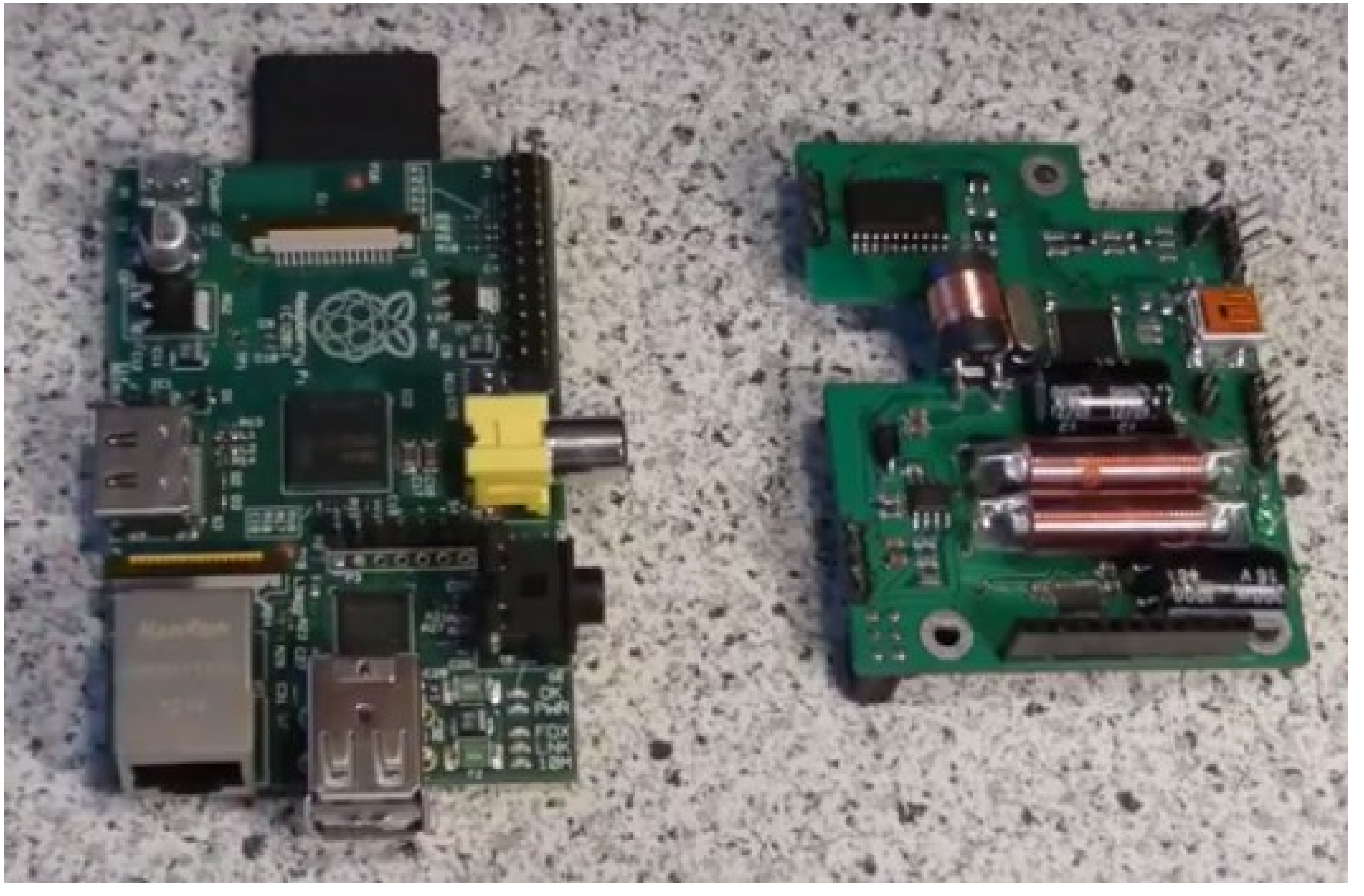
A few approaches to firmware projects design

Boards set for Raspberry Pi extension.

- ***Python*** based computer side SDK.
- ***Pawn*** embedded scripting machine.
- Boards in stack are driven by either main board, PC or Raspberry Pi.
- If coupled with Raspberry can provide it with power when needed.

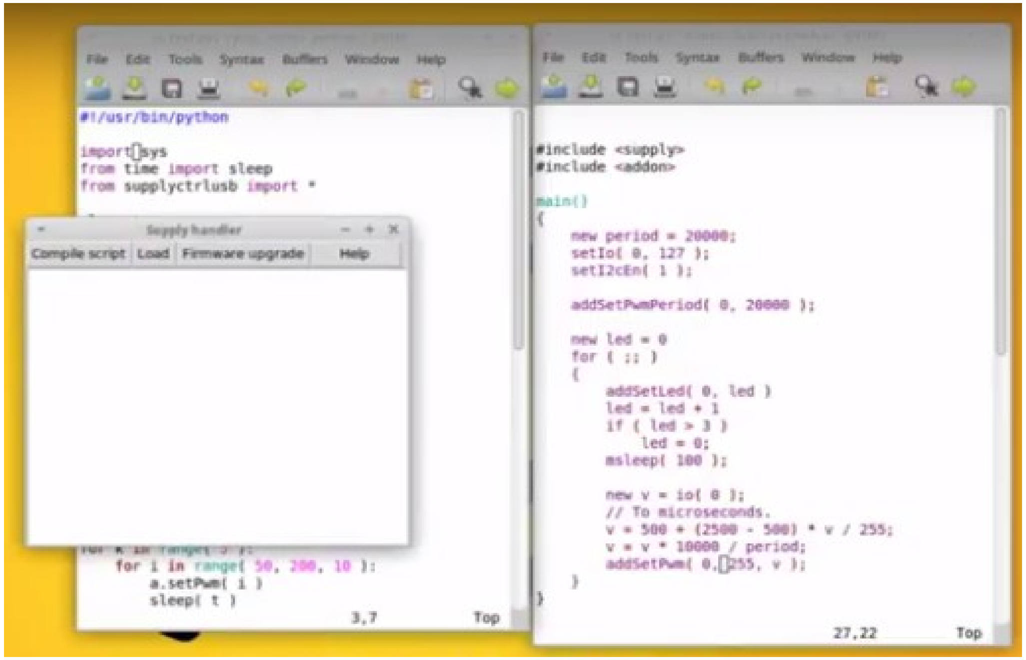
Boards set for Raspberry Pi extension.



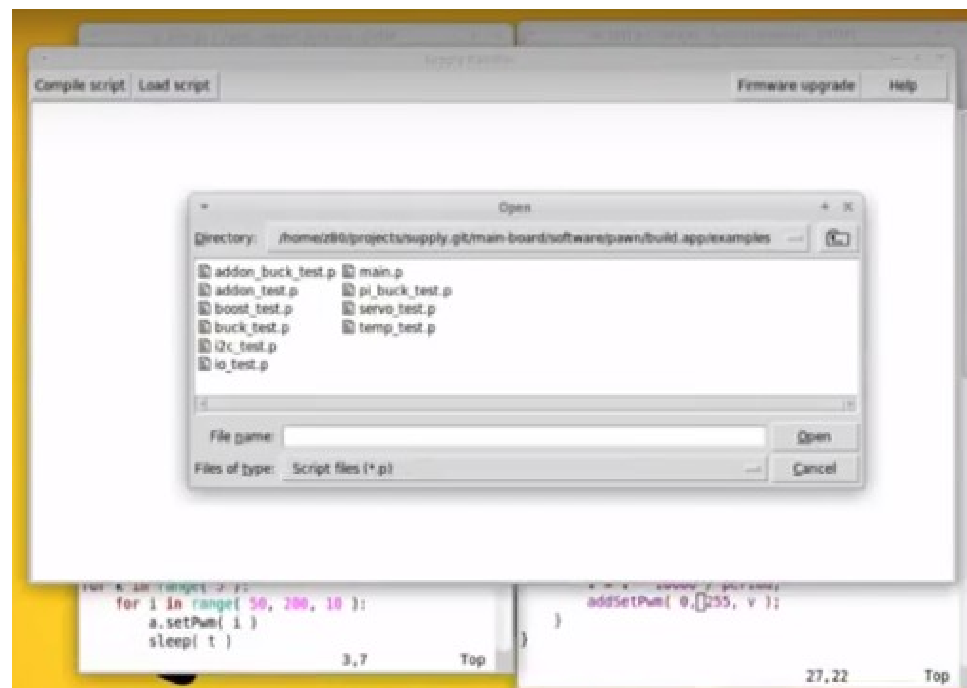
Boards set for Raspberry Pi extension.

- Two extension boards with motor drivers, PWM, IO pin arrays.
- Main board with adjustable Puck, Boost converters, USART, SPI, I2C, USB interfaces.

Boards set for Raspberry Pi extension.

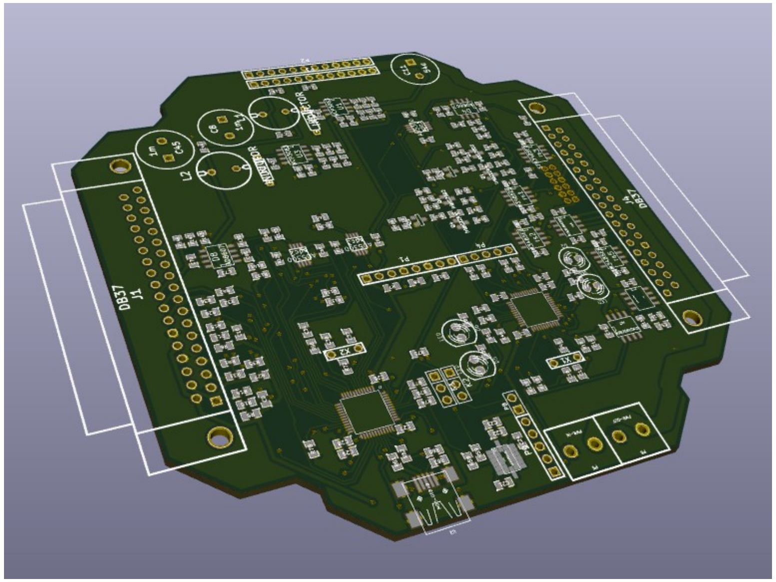


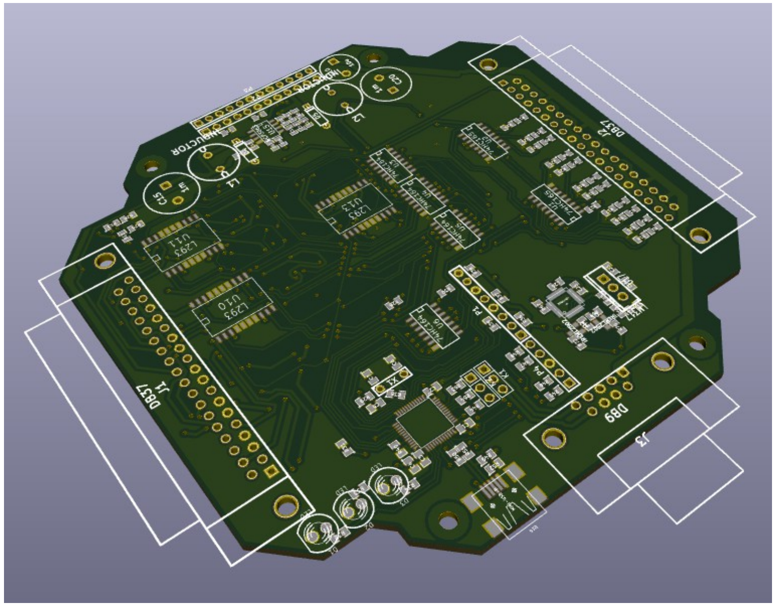
- ***Python*** and ***Pawn*** scripts examples side by side.



- New embedded script flashing dialog.

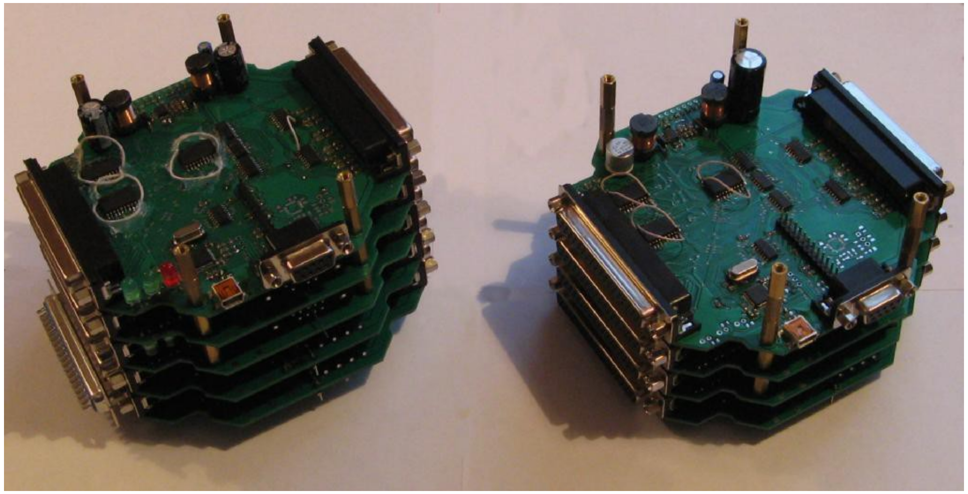
Scalable controller for pneumatic mechanism.





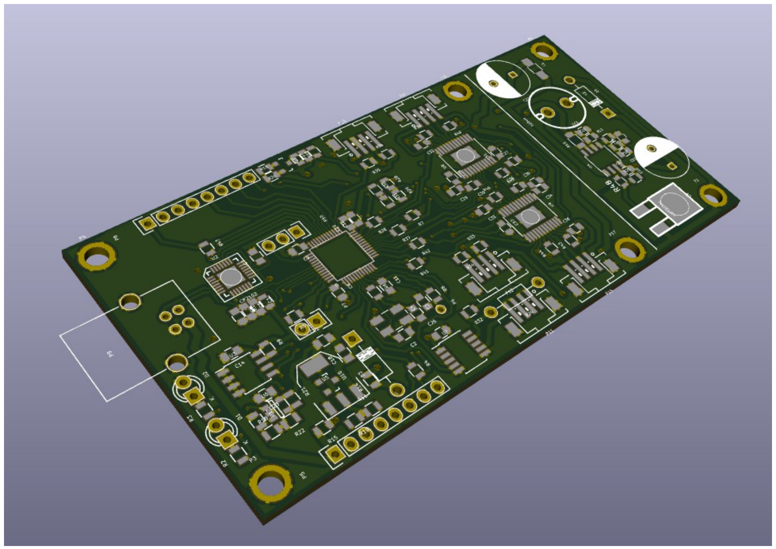
- The goal was to design cheap controller for a few mechanisms driven by pneumatic valves and electric motors.
- Minimize parts cost and PCB tooling cost.
- Number of controllers was small.
- Number of elements to control was not fixed and was relatively large ~150-200.

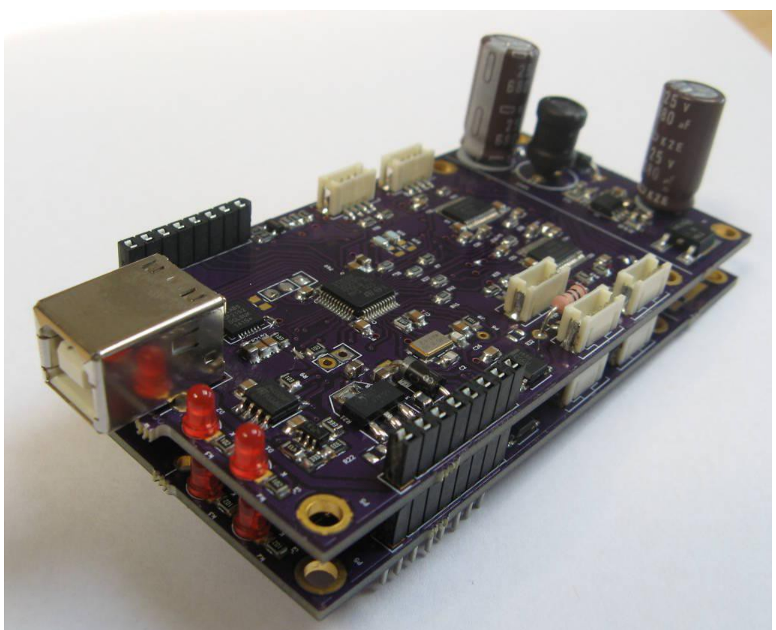
Scalable controller for pneumatic mechanism.



- Taking into account all possible combinations of pneumatic actuators, motors and sensors in all mechanism configurations
- it turned out that ***“optimal”*** numbers of different PCB types was 2.
- Boards are interconnected by ***power*** lines, common ***I2C*** bus and 2 ***digital*** lines for synchronous triggering.
- External communication was performed by either ***I2C*** or ***USB*** by single board computer running Linux OS.
- During debugging stage also was equipped by embedded ***Pawn*** machine for easy movement algorithms trying/implementing.

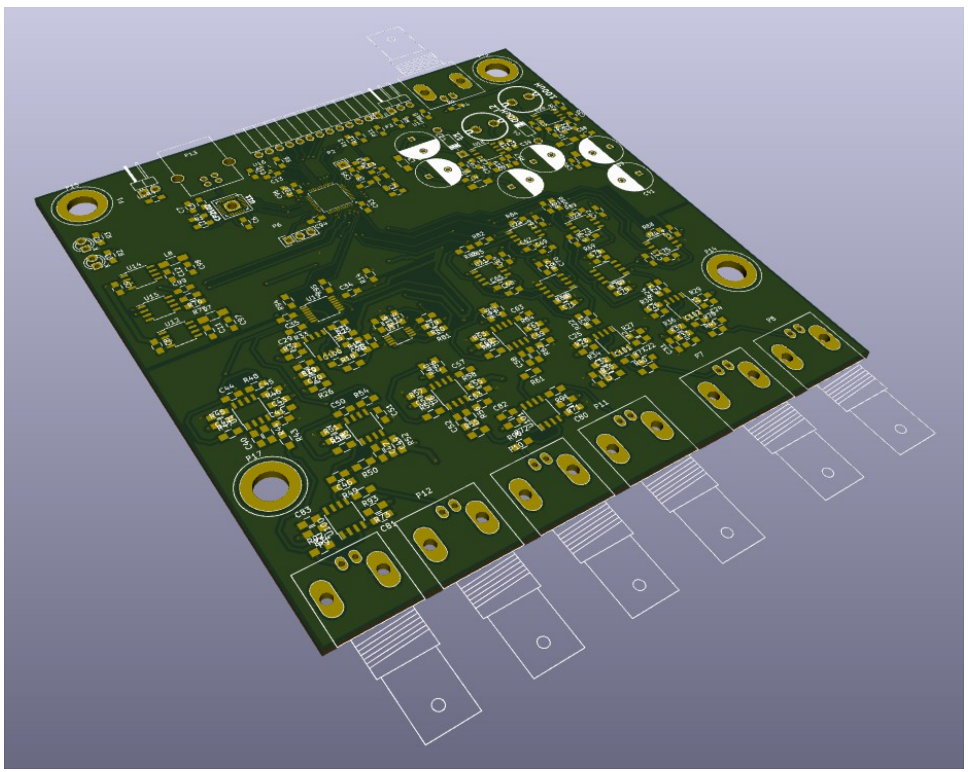
USB powered expandable motor controller





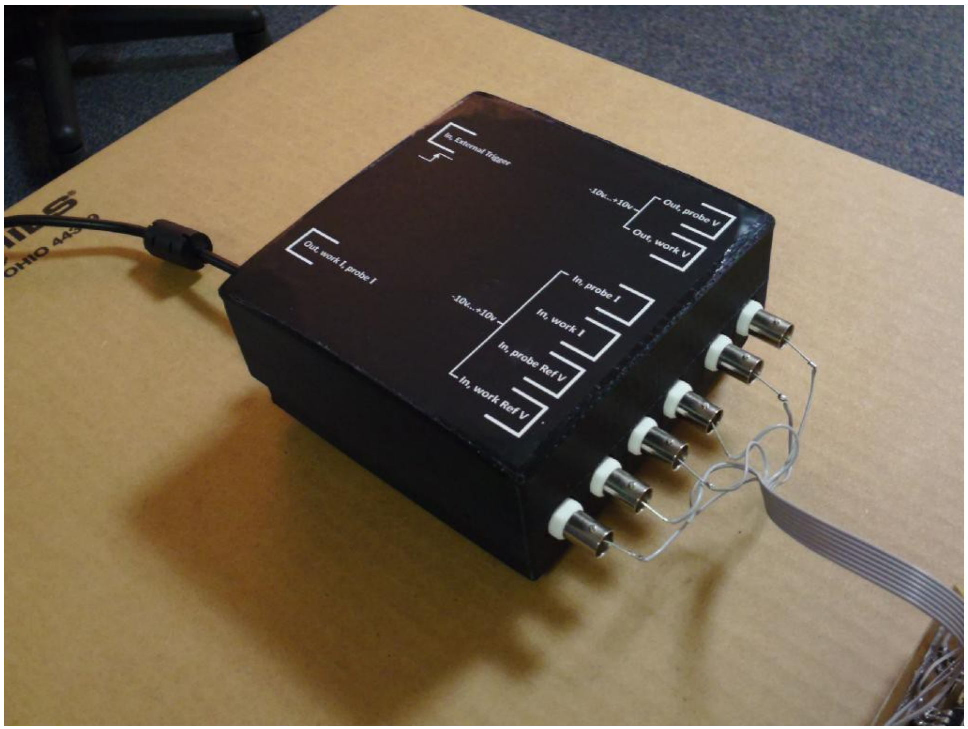
- The same approach applied to motor control application.
- PCB with integrated USB driven power supply.
- One board can drive 2 stepper motors, 1 DC motor, 2 sensors.
- Scalable design by reusing the same board a few times and addressing individual boards using I2C bus and GPIO triggering for synchronization.

3rd party bipotentiostat control module.



- 2 analog outputs +/-10V
- 4 analog inputs +/-10V
- External digital trigger
- One switchable analog output
- Analog precision to achieve 1mV
- USB computer interface
- Spend minimum resources as client had no extra money to finance the development.

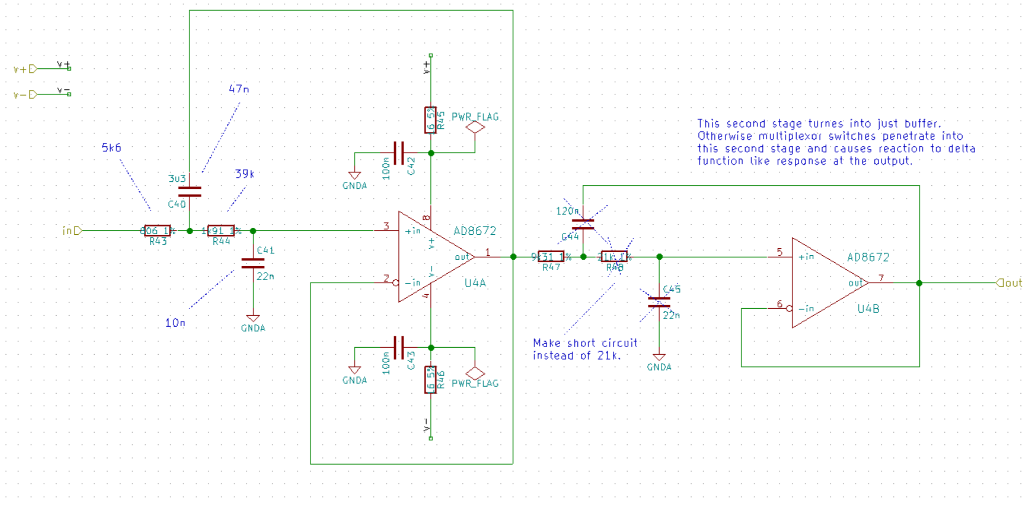
3rd party bipotentiostat control module.

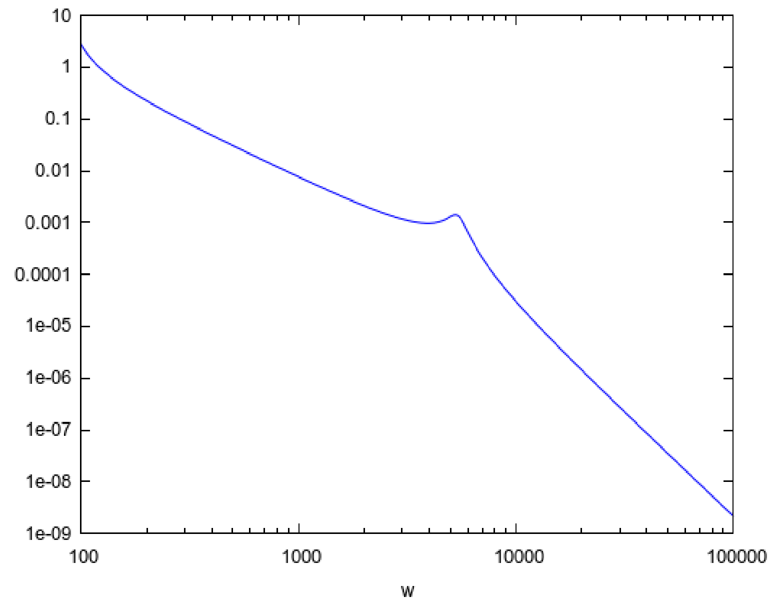


- Mostly used ***5%*** or ***1%*** resistors
- Built in ***mcp3063*** based ***boost 5V*** to ***12V*** and ***inverting 5V*** to ***-12V*** switching sources.
- On board ***temperature sensor*** for calibration purposes.

- Calibration setup was equipped with potentiometers and a heater under near the device for statistical data acquisition

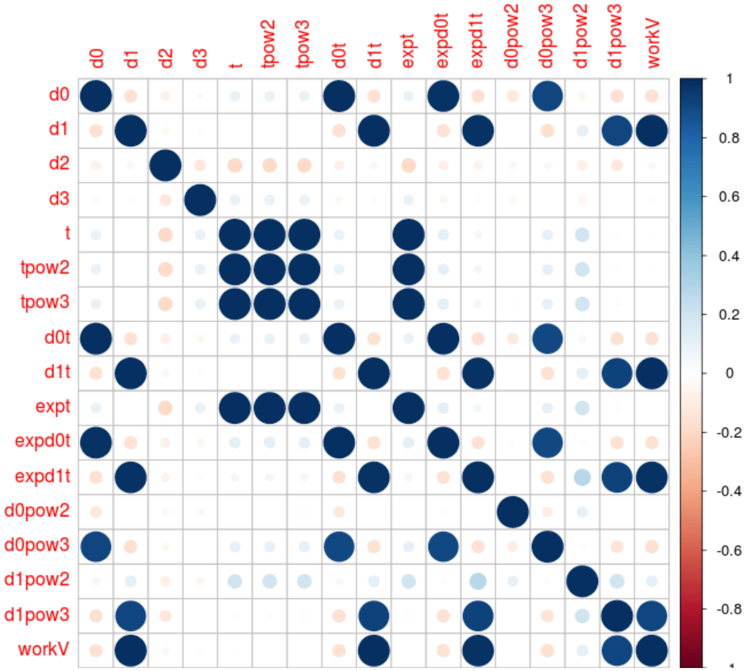
3rd party bipotentiostat control module.

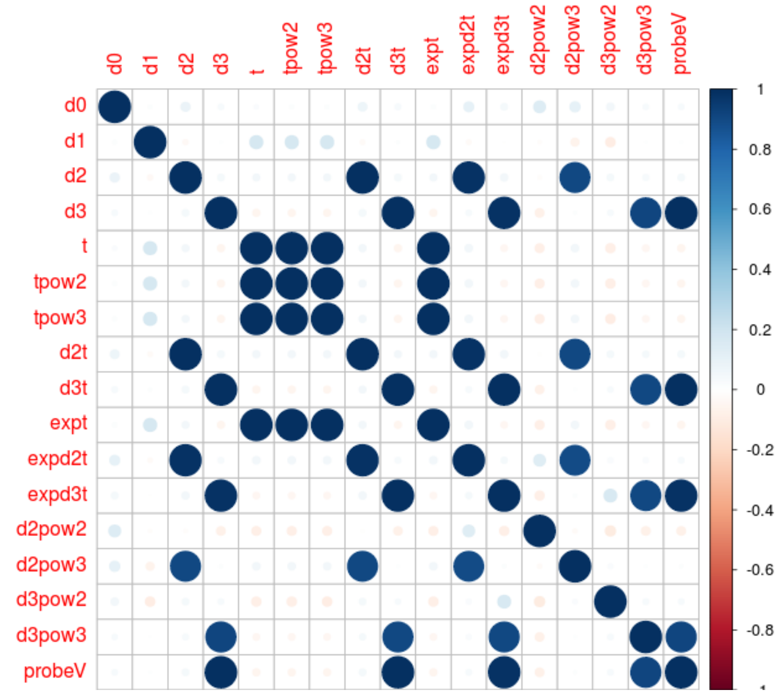




- Input filter design and Bode plot.
- Based on Chebyshev filter but with lowered response near cutoff requency to reduce oscillations caused by step response.

3rd party bipotentiostat control module.





- Examples of parameters and outputs cross correlations

- After gaining statistical data went to higher dimensional space by involving first few powers of each parameter, exponentiation and a few parameter products.
- Used this data set to find the best mean out of sample error by applying linear regression to all possible parameter combinations.
- Obtained formulas and dropped parameters which didn't improve match better then chosen margin (1^{-5})

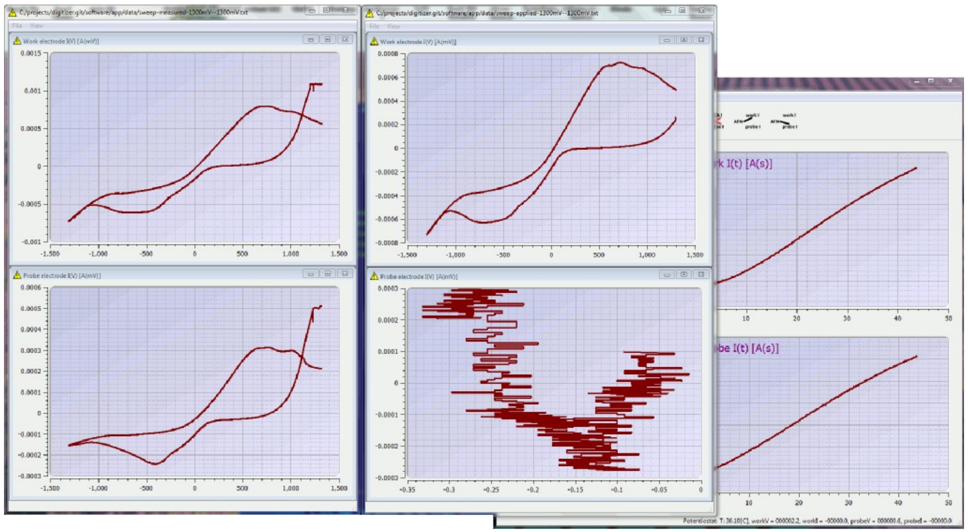
3rd party bipotentiostat control module.

- Formulas obtained for data readings conversion

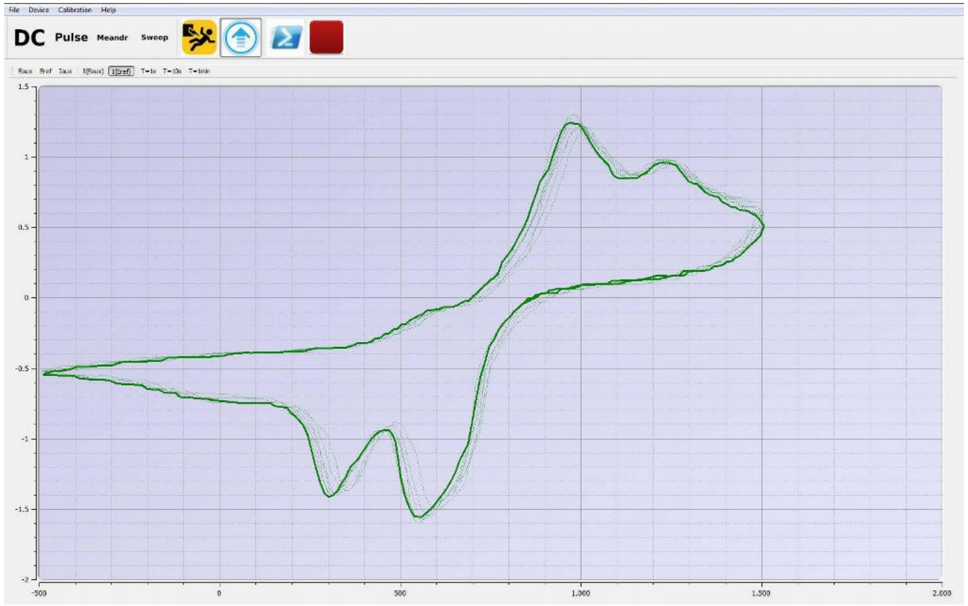
$$V_{out} = a_1 * dac_1 + a_2 * dac_2 + a_3 * dac_1 * T + a_4 * dac_2 * T + a_5 * T^3 + a_6$$

$$V_{in} = a_1 * adc + a_2 * T + a_3 * T^2 + a_4 * T^3 + a_5$$

3rd party bipotentiostat control module.



- PC based software for driving control module.



- Just nice looking electrochemical cycling I(V) curve obtained during system testing.