**Reflective Solar Tracking Control System**

*Calculating motor time intervals using astronomical data with the Raspberry Pi*

By: Nathan D. Williams and Nicholas P. Truncale, University of Scranton

Mechanical tracking systems are available to reposition a variety of devices including radio telescopes, antennae, and even television satellite dishes. These systems are large and often unaffordable for the small project developer. We are team members of a collaboration including two universities and a non-profit sustainable energy entity whose goal is to develop and commercialize a new solar energy collection device.

This article will describe the new patent pending device called the Reflective Solar Tracker (RST) and will showcase the control system’s function. The control system’s main hardware component is the Raspberry Pi, whose primary algorithm uses astronomical data to reposition the RST and collect voltage data for power and energy calculations.

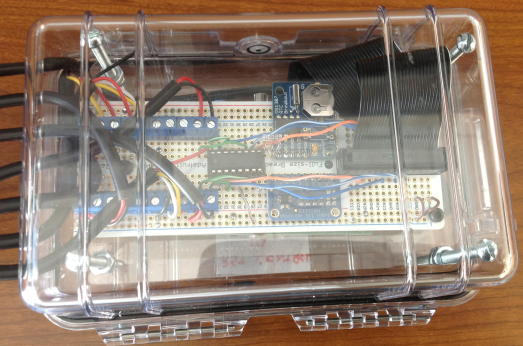
**The Reflective Solar Tracker**

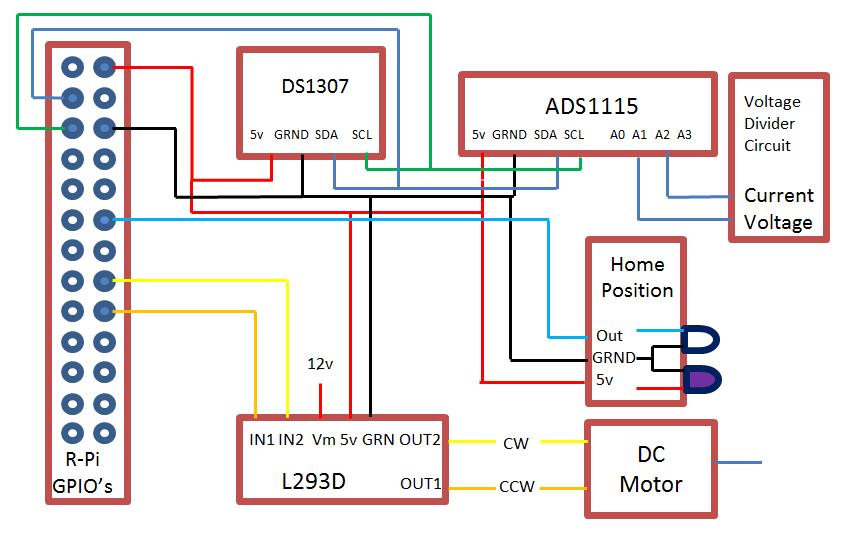
****

This device utilizes both reflected sunlight via Mylar panels and a rotating base platform to increase the energy density impinging on commercially available solar panels. The extra sunlight from reflection saturates the individual crystalline solar cells while the rotating base ensures the saturation takes place for a longer portion of the day compared to conventional stationary installations. The repositioning on the base platform is accomplished using a gear and worm screw turned by a low power DC motor controlled by the Raspberry Pi. The following describes the components for controlling a bi-directional DC motor, collecting voltage and current data from a solar panel installation, and the algorithm using astronomical data to guarantee the RST is always sun facing.

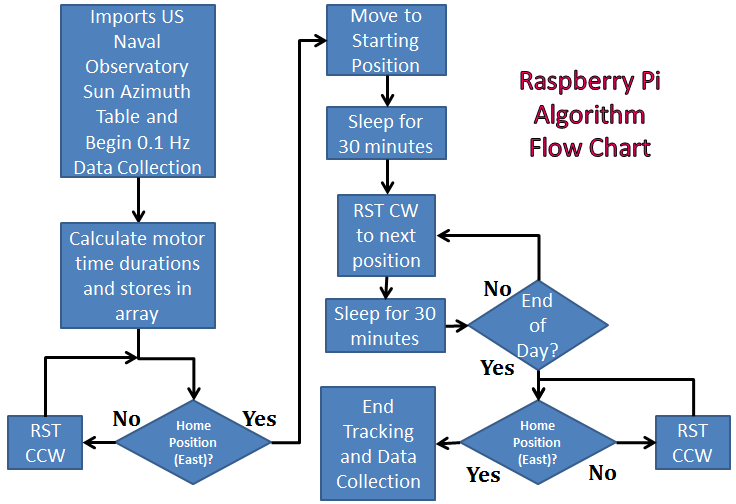
**Hardware Description**

The hardware architecture for the raspberry pi control system functions with the use of products from <http://www.adafruit.com> specifically the DS1307 Real Time Clock, ADS1115 16-Bit ADC, L293D H-Bridge IC, IR home-position sensor, and a 5200mAh power bank. The Real Time Clock is used to ensure the program executes daily at the proper time in case of power failure. To minimize the chance of failure, the Raspberry Pi is powered by a power bank, which is simultaneously being charged by a 5V 1A power adapter. The motor is directly controlled by an H-Bridge that is interfaced with the Raspberry Pi GPIO pins. In addition to the H-Bridge for motor control, an IR home-position sensor is used to ensure the motor is returned to the proper location at the end of every cycle. The hardware architecture also includes an Analog-to-Digital converter that allows us to collect the solar panel output voltage data throughout the day. We also collect the voltage across a 1Ω power resistor giving us the current.The connections of all of these components can be viewed in the provided block circuit diagram**.**

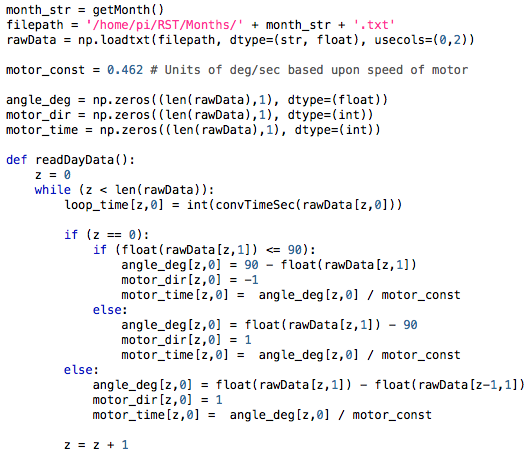


****

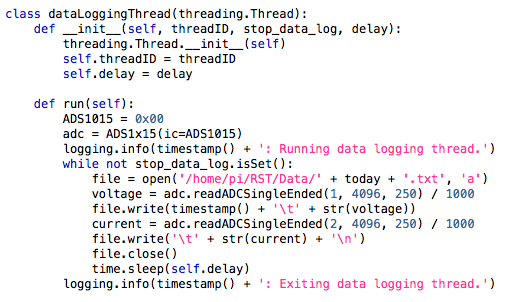
**Tracking and Data Collecting Algorithm**



The primary algorithm, executed by a daily cron job, retrieves a local sun azimuth table containing the sun’s position in degrees relative to the eastern direction for the installation’s geographic location in the current month. The hardware allows the motor to turn the base platform in both the clockwise (CW) and counter clockwise (CCW) direction. Beginning at 6:00am, the algorithm calculates the starting direction in reference to East and moves the base platform CW or CCW to this position. Throughout the rest of the day, the motor is turned on for a specific interval every half hour and moves the base platform CW, in the direction of the sun’s trajectory, to the next sun facing position. The following code, which of note utilizes the numpy library, shows the readDayData() function that calculates the specific time durations from the imported sun azimuth tables. The durations are stored in the motor\_time[] array. Once the algorithm reaches the end of the imported file, the RST rotates CCW back to the home position so it is facing east the next day.

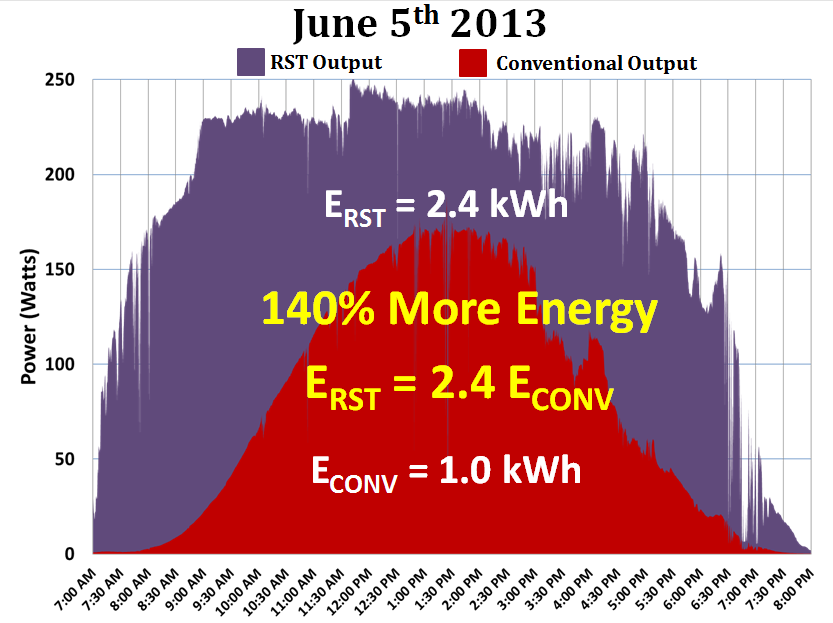


For our data logging function, we used some of the code from the Adafruit ADS1x15 class combined with the python threading class. The method itself is written as a thread that is kicked off in the main method. Once executed, the thread will open a new text file titled with the current date. Inside the file, the thread will log and timestamp data from inputs A1 and A2 every 10 seconds (0.1 Hz) until the daily tracking function ends and the thread stops.



**Reflective Solar Tracking in the Field**

Using the collected voltage and current data from the solar panels, we multiply the values together to get the power output of the panels. Integrating these power values in kilowatts over the total amount of hours of sunlight, gives you the total energy output in kilowatt-hours (kWh) of your installation. Our initial study compared the RST energy output versus a conventional stationary non-reflective installation. The plot below shows the power and energy comparison on a perfectly sunny day.



We recently sent two RSTs to two locations in Uganda where one will be used to power an electric water pump at a rural parish school and the other to St. Josephs secondary school as a science experiment where the students will collect data with the Raspberry Pi as a project and send us the files for our study. We also gave a keynote presentation at the 2013 Energy Path conference about the RST and the Raspberry Pi control system. Anyone with questions regarding the control system or project may contact us at any time. [Nathan.williams2@scranton.edu](mailto:Nathan.williams2@scranton.edu), [Nicholas.truncale@scranton.edu](mailto:Nicholas.truncale@scranton.edu)