ECE 09.402 Introduction to Smart Buildings IoT for Smart Buildings: Energy Management System

Kevin Bellomo-Whitten*, Eric Guidarelli†, Jeffery Welder‡

Department of Electrical and Computer Engineering
Rowan University
Glassboro, NJ USA
kevin.a.whitten@gmail.com*, guidar01@students.rowan.edu†, Welder81@students.rowan.edu‡

I. Introduction

When designing Smart Building architecture, one of the pivotal components is some sort of renewable energy source to supplement building electrical loads. This allows the end user to draw from their own source of power to both run their renewable technologies more efficiently, as well as export energy to the utility grid for a cost savings. The benefit that utilities gain from this is that they have many local resources to draw power from to help stabilize their grid.

There is, however, a caveat to this exchange of power. There can be points in time in which the end user would rather store their energy than use it to shed building loads. Utilities also have down times where the exported power from generation plants can cause frequency instability in the system. The solution to this situation is to have some type of smart system that manages an end users building loads as well as their renewable assets. What our team proposes is to design, simulate, and ultimately construct an energy management system which monitors vital points in a given micro grid, and intelligently directs what the end users renewable assets should do with their energy production.

II. Project Requirements

A. Monitoring System

To fit the project in the constraints of a one-semester time-frame, only the monitoring portion of the system was implemented

- Simulate a micro grid over a one day time-span
  - Utility power
  - Renewable energy source
  - Energy storage unit
  - Monitoring system
- Generate mock input data for monitoring system
- Communicate between monitoring system the simulation

B. Analysis of Requirements

Simulink was chosen as the simulation software. MathWorks has a toolbox for the SimuLink program that has power grid components. Due to its availability and its familiarity to team members, a Raspberry Pi was chosen as the monitoring system that would gather the mock relevant data. The simulation and monitoring system would communicate over wireless User Datagram Protocol (UDP).

III. Application Hardware

The hardware used in the application is shown in the following schematic:

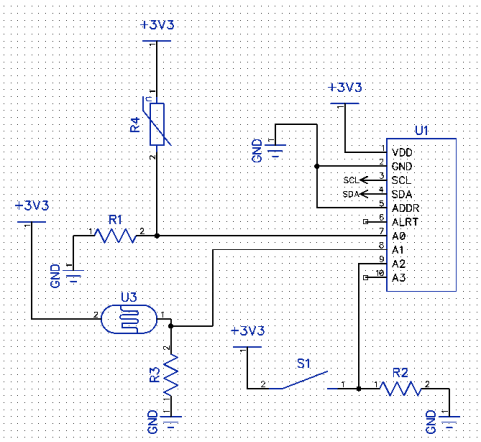

Fig. 1. Schematic of Implemented Circuit

List of components and Raspberry Pi.

- Raspberry Pi
  - Prototyping Wires
  - power cord
- Thermistor
- Photo Resistor
- Breaker Switch

All components were arranged as voltage dividers into ADC inputs, allowing a set range of voltages from $0 - V_{CC}$. The photo resistor was setup to measure irradiance in the simulation, thermistor to show temperature on the panel, and finally a breaker switch to turn on and off loads.

Although having success simulating the two components of the system singularly, combining them was not successful.

This was due to the type of calculations done by the powersim toolbox and its utilization of phasors. The simulation would run a 24 hour day in 20 seconds, however when the sensors were included, it slowed the simulation by orders of magnitude.
UDP was used to transmit sensor and simulation data back and forth between the host laptop and the raspberry pi.

IV. Application Software

A. Connecting the I^2c ADC to Simulink

The ADS1115 was connected to Simulink in a rather roundabout fashion. MATLAB was used to compile C language, including library dependencies located on the raspberry pi, and then push it onto the linux machine. Mathworks has a special version of a linux distribution Raspian that they released in order to run external simulations on the Pi.

B. Simulink Block Diagram

The simulated uGrid was expressed as a power network of single-phase AC (200 V). Solar power generation (maximum 5kW) was attached as a renewable energy source. Power sources were system power, solar power generation, and a storage battery (150 V, 30 Ah). The storage battery is controlled by a battery controller, and it absorbs surplus power if there is surplus power in the uGrid or it supplies insufficient power if there is a power shortage in the uGrid. Three generic household devices consume power (maximum 2.5kW) as electric loads.

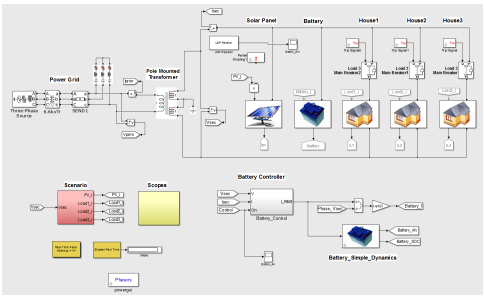

Fig. 2. Micro-Grid in Simulink

To more accurately represent a full, self-sustaining grid, the uGrid is connected to the system power via a pole-mounted transformer. The voltage source (66 kV) of three phase alternating current of the system power is connected to a transformer (primary 66 kV /secondary 6.6 kV) which decreases voltage from 66KV to 6.6 KV. The pole-mounted transformer (primary 6.6kV /secondary 200 V) changes the voltage from 6.6 kV to single-phase AC (200 V). The frequency of AC cycles is set to 60 Hz. The solar power generation and the storage battery are simulated as DC power sources converted into single-phase AC. These are both connected to the uGrid. It is assumed that in control strategy, the uGrid does not depend on system power for power consumption, and required power is provided by solar power generation and the storage to the extent possible.

C. UDP's Role In Simulation

As shown in Figure 3, the appropriate output data was first sent to a scope ScopeSPS to visually verify that all parts of the simulation were being output correctly. Then, using specialized UDP Send blocks obtained from the Simulink Support Package for Raspberry Pi , each dataset was broadcast to a specific port ID on the local wireless network. With the Raspberry Pi connected to the wireless network, and the virtual Raspberry Pi interface connected via Simulink, UDP packets were then able to be sent from the simulation wirelessly to the Raspberry Pi, and then broadcast on the network from the device hardware. It was then a simple procedure to call on the specified ports using UDP Receive blocks native in Simulink and then viewing the outputs on a separate scope.

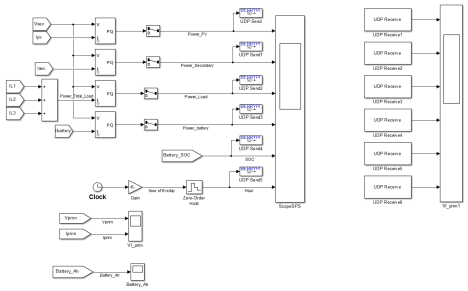

Fig. 3. Simulates generating, send, and receiving data over UDP

To implement the real-time sensor data readings, the simulation took advantage of the Raspberry Pis built in I2C capabilities. To correctly interface the I2C components with Simulink, a block was used from the Matlab Exchange network to create the architectural links between the physical ADS1115 readings and the simulation.

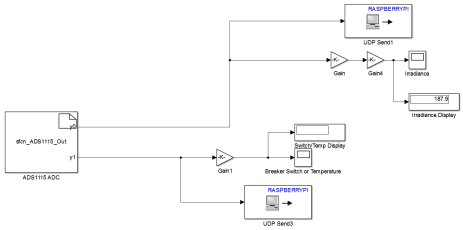

Fig. 4. Simulink model saved to the Raspberry Pi

Using this block, the ADS1115 was connected to Simulink in a rather roundabout fashion. Matlab was used to compile C language, including library dependencies located on the Raspberry Pi, and then push it onto the linux machine. Mathworks has a special version of a linux distribution Raspbian that they released in order to run external simulations on the Raspberry Pi.

V. Testing

Due to the taxing requirements from a simulated uGrid, the Simulink model was set up to only simulate a 24 hour period of load requirements. This was done using the phasor version of Simulinks powergui, to allow for complex power

to be simulated. However, by using the phasor representation of AC power, the simulation would not run in real-time. To help simulate real-time, and to allow data manipulation to occur in the simulation, a Pacer block was implemented from the Matlab Exchange to help step into the 24 hour simulated data. A scenario was generated to provide minute granular data each load aspect of the Simulink model (residential loads, PV generation). This data was created based off of the secondary power provided by the 200VAC power transformer. The following criteria was used in the 24 hour simulation:

- As a typical load change, the amount of electric power load reaches peak consumption at hour 9 (6,500W), hour 19, and hour 22 (7,500W)
- From hour 20 to hour 4 (nighttime hours), solar power generation is 0W. It reaches the peak amount (5kW) from hours 14 to 15
- Battery control is performed by the battery controller, which can be manually switched on or off

Other things to note are followed:

- The storage battery supplies the insufficient current when the power of the micro-grid is insufficient and absorbs surplus current from the micro-grid when its power is surpasses the electric load
- A simulated loss in load occurs at hour 8 for 10 seconds. This spike can be observed in the active power on the secondary side of the pole transformer and the electric power of the storage battery
- A simulated decrease in irradiance occurs at hour 11 for 1 minute. This spike can be observed in the active power on the secondary side of the pole transformer and the electric power of the PV generation.

The simulation results are shown in Figure 5

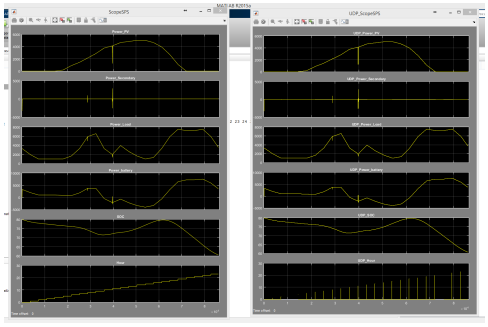

Fig. 5. Example output of a full days power usage

This yielded a relatively accurate wireless data transfer of the simulated information through the Raspberry Pi (see figure 7). As shown, even interruptions of as little as 10 seconds on the system were successfully transferred via UDP packets and translated to a separate interface via the Raspberry Pi. This was promising, in that we could assume that the loss was very minuscule in the grand scheme of this design, since most commercial monitoring systems do not get more granular than 1 to 5 minutes. Having such a fine resolution on our system would allow for increasingly accurate readings for the end user.

VI. Future Work

Built-in energy storage blocks in Simulink did not realistically simulate storing energy. Including free weather APIs that could provide information on cloud coverage, reducing PV arrays effectiveness.

Further work creating a two-way smart inverter in the Simulink environment was discussed. This would enable the possibility of charging batteries from the grid during low cost energy periods in the day. Implementing this would show the most cost saving for a solar building, as this information reduces grid stress during peaks, and allows for buildings to become self sufficient when their system is fully charged.

VII. Conclusions

Two working components, but issues communicating between live sensor UDP and simulation UDP kept them separate. Simulink proved to be an adequate platform for power simulation and interface between physical sensors. Although much time was spent learning the tools, and interactions between blocks, simulink remained an integral part of our energy monitoring system. The group was successful implementing monitoring and has since had many thoughts on how to further the work to include energy management.

It can be hypothesized that should this design be implemented using an existing uGrid, rather than a simulation, that the data could be accurately transferred in real-time, and then combined with sensor data also supplied. This project then opens the door for more robust designs, including converting this into an intelligent EMS. The RUEMS can be an extremely using hardware/software package in the future, as its low cost design can finally make residential monitoring systems more appealing to the end user, thus giving more incentive to move ones home to a Smart Home.

References

[1] Drogon. *TWiring Pi GPIO Interface Libarary for the Raspberry Pi.*. [Online]. Available: http://wiringpi.com/, 2015.

[2] Vishay. *NTC Thermistors, Radial Leaded, Standard Precision.* Available: http://cdn.sparkfun.com/datasheets/Sensors/Temp/ntcle100.pdfhttp://cdn.sparkfun.com/d 2012

[3] Sparkfun. *CdS Photoconductive Cells* Online. Available: http://cdn.sparkfun.com/datasheets/Sensors/LightImaging/SEN-09088.pdf, 2014

[4] Pchan. Raspberry Pi 2 Model B GPIO 40 Pin Block Pinout Online. Available: http://www.element14.com/community/docs/DOC-73950/l/raspberry-pi-2-model-b-gpio-40-pin-block-pinout, 2015

[5] NXP Semiconductors. *I2C-bus Specification and User Manual* Online. Available: http://www.nxp.com/documents/user_manual/UM10204.pdf, 2015

[6] Raspberry Pi Foundation. *About the Raspberry Pi* Online. Available: https://www.raspberrypi.org/, 2015

[7] Adafruit. *New Product! ADS1115 16-bit ADC* Online. Available: http://www.element14.com/community/docs/DOC-73950/l/raspberry-pi-2-model-b-gpio-40-pin-block-pinout, 2012

[8] Simplified Model of a Small Scaled Micro-Grid [Online]. Available: http://www.mathworks.com/help/physmod/sps/examples/simplified-model-of-a-small-scaled-micro-grid.html?refresh=true

[9] Raspberry Pi Support from Simulink [Online] Available: http://www.mathworks.com/hardware-support/raspberry-pi-simulink.html

[10] Real-Time Pacer for Simulink [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/29107-real-time-pacer-for-simulink

[11] Simulink Raspberry Pi Simulink Driver Blocks [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/51232-raspberry-pi-simulink-driver-blocks-adc-dac-pwm
[12] Simulink Raspberry Pi Driver Blocks [Online]. Available: http://engineer.john-whittington.co.uk/2015/06/simulink-raspberry-pi-driver-blocks/