Project Design Document (PDD)

Smart Home Appliance Control System (SHACS)

Jeremy Nicholson
Fall 2014

Approvals

Name _____ **Date** _____
Name _____ **Date** _____

1. ***Architectural Design***

   The Smart Home Appliance Control System will consist of four different modules that will all interact with a main control module in a STAR type network topography (5 modules in total(Fan will be Optional) – figure 1).

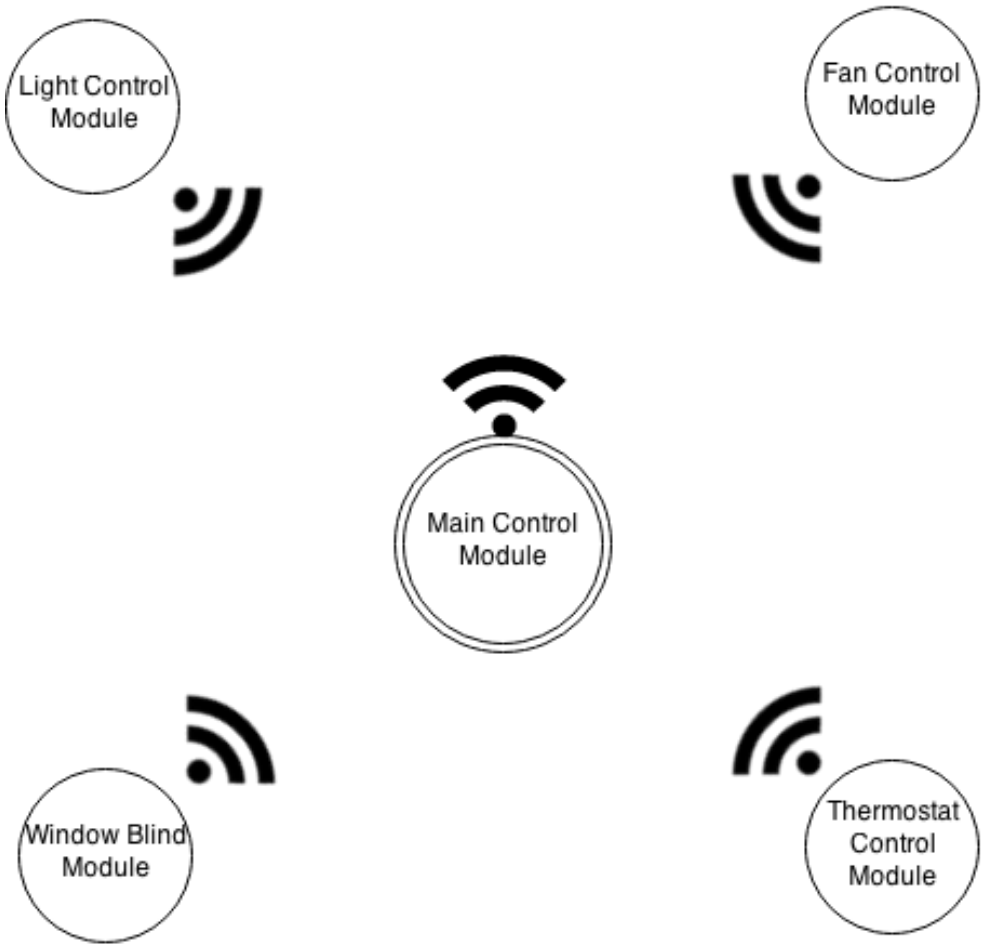


Figure 1 – High Level Diagram of Smart Home Appliance Control System

2. Hardware Design

The main control module will handle all of the user input for the device and will be the host for the main menu and menu screens.

Main Control Module:

- STM32F429 w/ TFT LCD screen
- Xbee series 1
- 3x AAA batteries

The light control module will handle the user input for the light module from the main control module.

Light Control Module:

- STM32F3
- Xbee series 1
- Light Dimming Driver Circuit (see 2.b for details)
- Lights voltage and current measuring circuit (see figure 3 in 2.b for details)
- PIR Sensor
- Light Bulb
- 3x AAA batteries

The window blind control module can be automated or controlled by the user input for the window module from the main control module.

Window Control Module:

- STM32F3
- Xbee series 1
- Light sensor
- Blinds
- 3x AAA batteries

The fan control module will handle the user input for the fan module from the main control module.

Fan Control Module:(Optional)

- STM32F3
- Xbee series 1
- Fan Control Driver Circuit (see 2.b for details)
- Fans voltage and current measuring circuit (see figure 3 in 2.b for details)
- Fan
- 3x AAA batteries

The thermostat control module will handle the user input for the thermostat module from the main control module.

Thermostat Control Module:

- STM32F3
- Xbee series 1
- LCD screen
- Temperature sensor
- Humidity sensor
- 3x AAA batteries

a. State Flow Diagram

On power up the main module will send out commands to initialize the sub modules and start the automation on the window blind module. After that has been achieved, the main control module waits for user input from the menu screen. From there the system will select the correct destination module and process the commands from the main control module. Once it has completed, the system waits for another input from the user (See Figure 2).

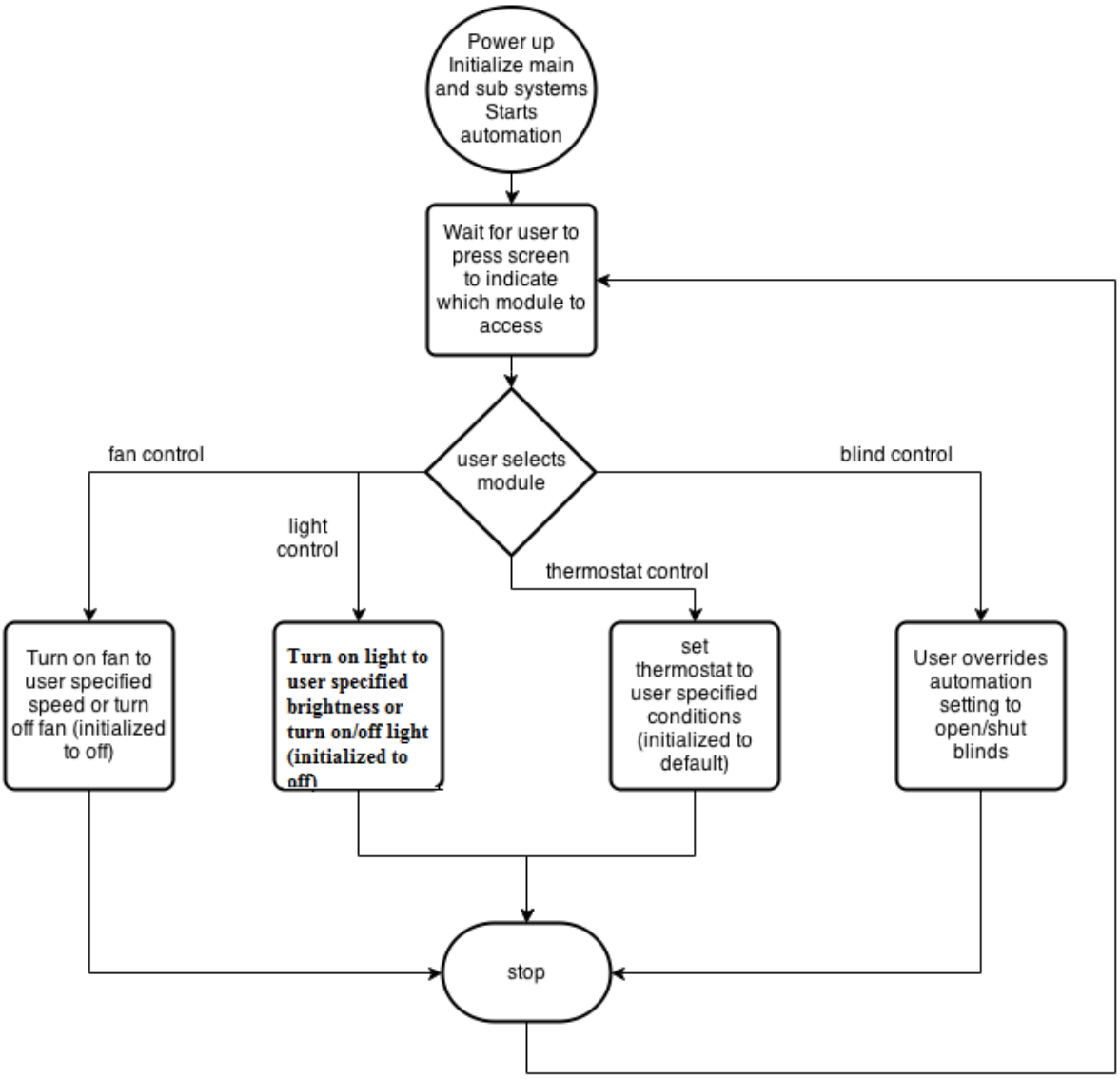
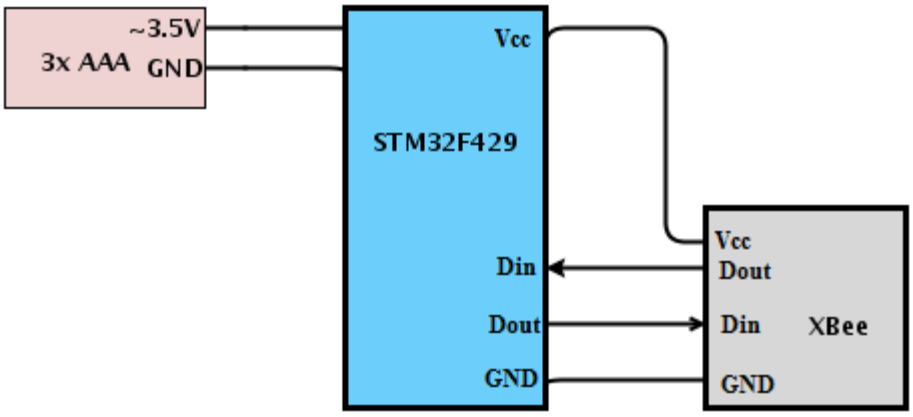


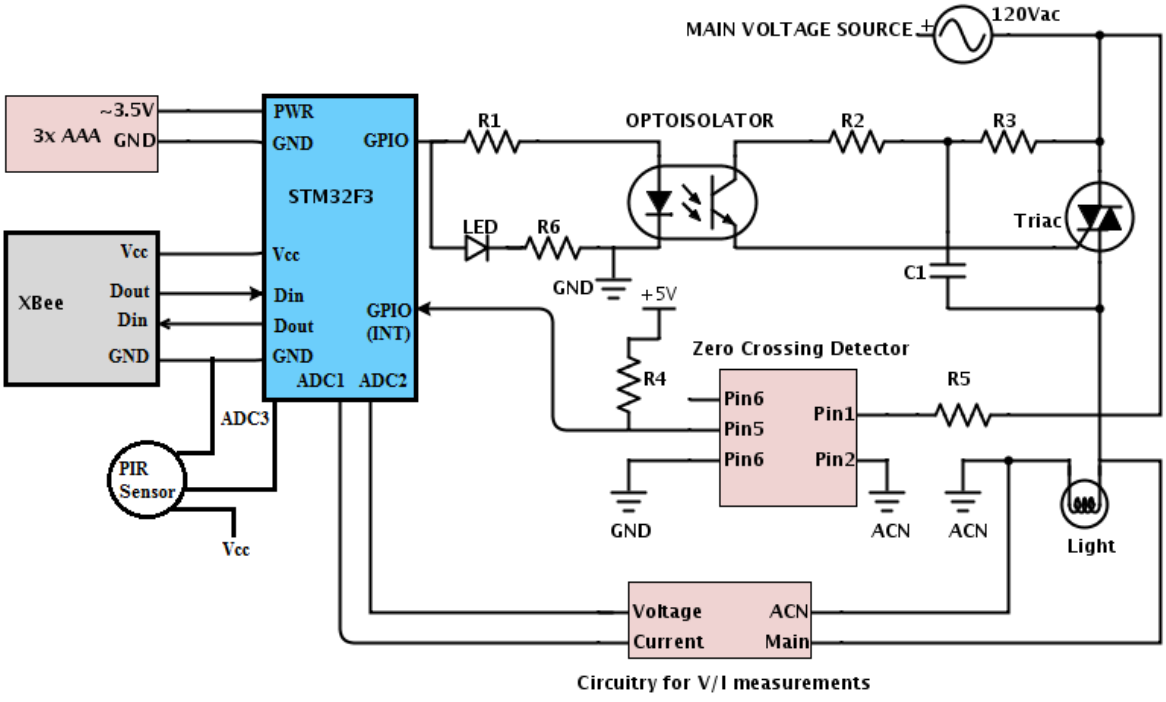
Figure 2 – State flow diagram of Smart Home Appliance Control System

b. Schematics

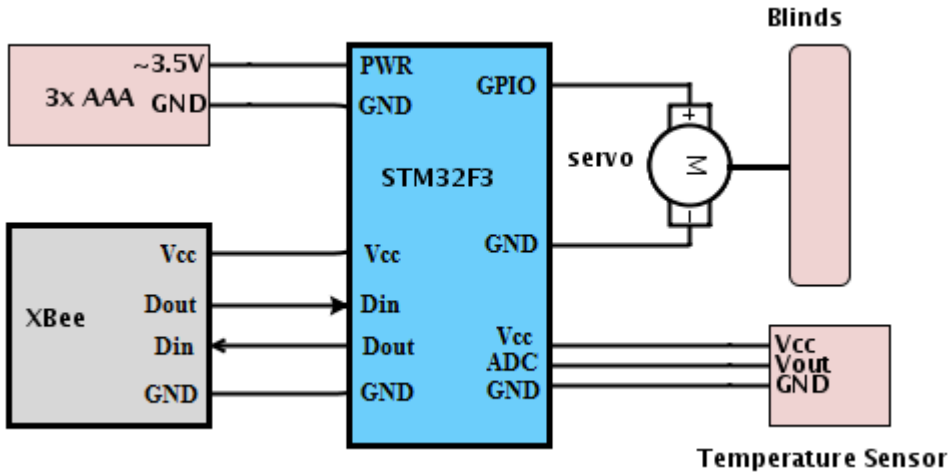
Main Control Module Schematic:



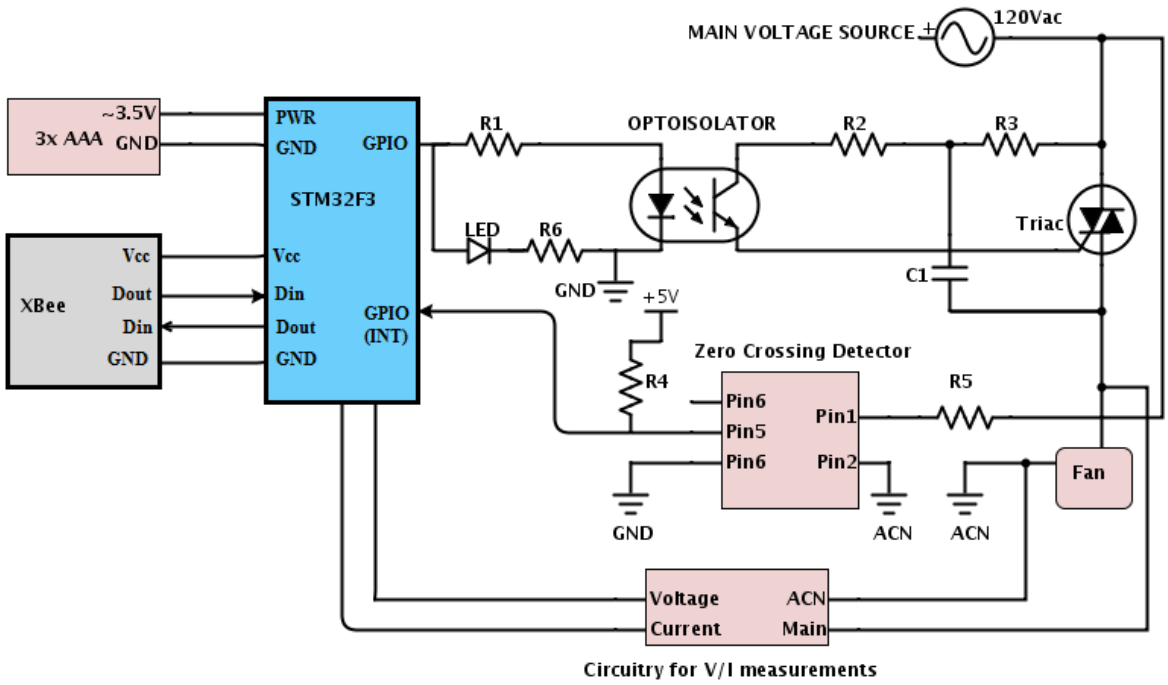
Light Control Module Schematic:



Window Blind Control Schematic:

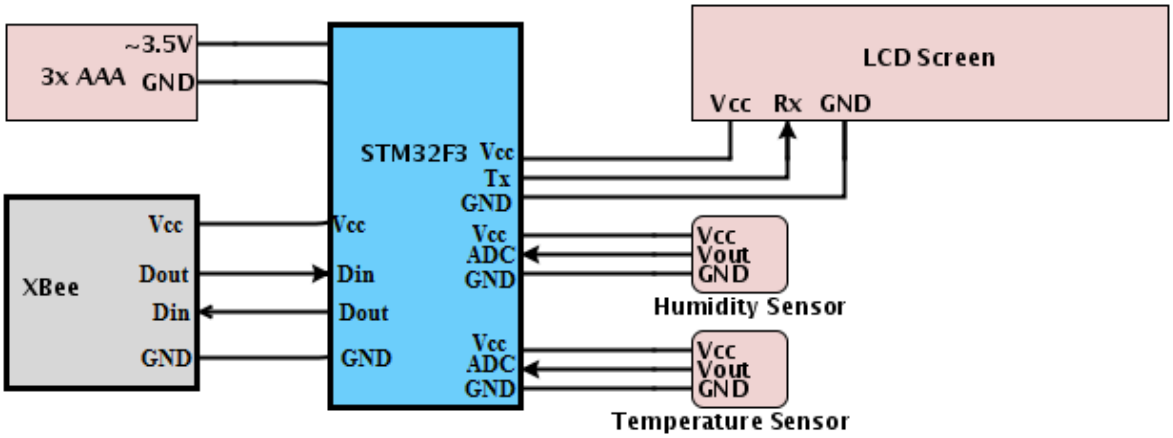


Fan Control Schematic:



(Optional)

Thermostat Control Schematic:



c. Timing Diagrams

The timing for the whole project is mostly asynchronous due to the Xbee using an UART. However, each individual module will require some kind of timing to gather input from ADC's. The timing diagrams will look something close to this generic ADC timing diagram:

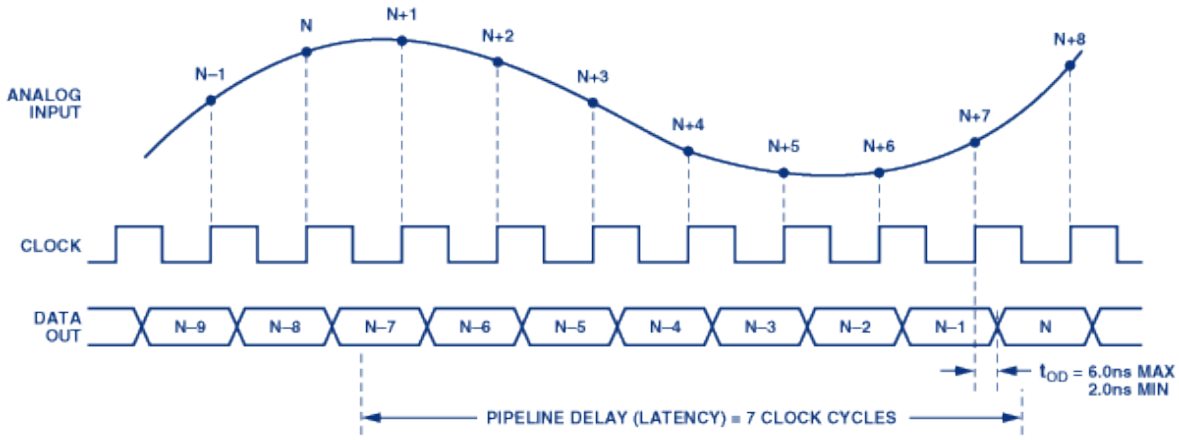


Figure 4: ADC timing diagram

Another part of the project that will require precise timing are the light/fan modules. The timing for these modules are required to dim/limit the power to the appliance in order to dim/slow down the appliance. The following diagram shows a chopper circuit. Since main's voltage produces a 60Hz sin wave, I will have to sync the microcontroller's timing to match that in order to get the proper lighting/fan effect that I want.

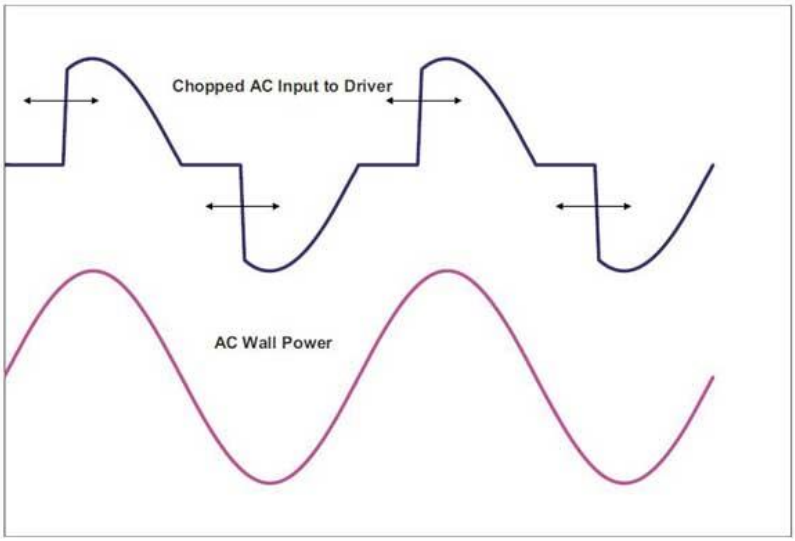


Figure 5: AC Dimming Chopper Circuit (75% Power)

d. Theory of Operation

Since the SHACS has multiple modules, each module will initialize itself on startup. The **main control module** will initialize the wireless communication (Xbee) for the star topology with the other modules as well as displaying the startup screen. With a click on the screen the user can easily be at the main menu screen (maybe, if able to, have a log in screen before the main menu). Starting with the Light control module and moving through the other modules, I will go into detail of each's initialization.

Light control module: Initializes Xbee and the light will start in the off state.

Fan control module(Optional): Initializes Xbee and the fan will start in the off state.

Thermostat control module: Initializes Xbee and will display current temperature and humidity of the room.

Window blinds control module: Initializes Xbee and will start in automation mode by reading the lux sensor and determining whether to open the blinds or not.

After the initialization phase has completed, the system waits for a user input for which module to control. Once that instruction is finished, the cycle repeats.

The system currently doesn't have a troubleshooting option besides restarting the whole system fresh.

3. Software Design

The system will be written in C using the IAR workbench environment. A lot of the software in this project will be used for programming GPIO, ADC, TIM, INT and Serial Communications. The other part of the software in this project will having to program the easy-to-use user menu.

a. Static Design

The static design of this system will be all the initialization of the peripherals that I am using for each module described above, but for a quick list of the peripherals: GPIO, UART, ADC, TIM and INT.

b. Functional Design

Main Control Module: Software to handle interrupts from Xbee and capture the incoming data.

Light Control Module: Software to handle timers for ADC's, GPIO's and INT that comes from the ZCD circuit. The GPIO is required to turn on and off the triac. The ADC will be started by a timer and will capture the data coming in from the PIR, and current/voltage sensing circuits. The V/I readings will then be stored in memory. Also, will have software to control the Xbee communications.

Fan Control Module(Optional): Software to handle timers for ADC's, GPIO's and INT that comes from the ZCD circuit. The GPIO is required to turn on and off the triac. The ADC will be started by a timer and will capture the data coming in from the current/voltage sensing circuits. The V/I readings will then be stored in memory. Also, will have software to control the Xbee communications.

Window Blind Control Module: Software to handle timers for ADC's and GPIO's. The Lux sensor will be captured by an ADC that is driven by a timer. The readings will then be stored in memory. Also, will have software to control the Xbee communications.

Thermostat Control Module: Software to handle timers for ADC's and GPIO's. The ADC will read the temperature and humidity sensors and will display the results to the LCD screen. The readings will also be stored in memory. Also, will have software to control the Xbee communications.

c. Dynamic Design

The dynamic part of the software in this project will be to program the easy-to-use user menu. The way I plan on implementing the menu is on startup you will be welcomed with a welcome screen and with a touch anywhere on the screen will move on to the main menu screen (maybe implementing a log in screen in between those two screens). I plan on implementing a stack based menu screen display that will put the different menus onto a stack so you can either push on the stack to go to another menu or pop off the stack to go back a menu page. This is an idea as of right now and will become clearer when actual implementation is started.

In addition, if the log in screen in implemented, with user name and password, then I might be able to add the feature of being able to customize setting for individual people.

4. Glossary

ADC – Analog to digital converter
TIM – Timer peripheral on the STM32 boards
GPIO – General Purpose Input Output on the STM32 boards
INT – Interrupt on the STM32 boards
Stack – Programming data structure with usually a FILO structure
ZCD – Zero Crossing Detection circuit that will feed into an interrupt on an STM32 board