Gesture-controlled Automated Residency Via Intelligent Systems (GARVIS)

Joshua Illes, Andres Mujica, Jackson Schleich, and Sarah Strauss

Dept. of Electrical Engineering and Computer Science, University of Central Florida, Orlando, Florida, 32816-2450

***Abstract* — The Gesture-controlled Automated Residency Via Intelligent Systems is a household control network that provides the necessary tools for a household environmental database and an algorithm based approach to regulating a residential environment. User interface is modernized with a capacitive touch light switch, a touchscreen central manager, and a motion sensor glove interface. Environmental control is achieved though AC load control with triacs, DC load control, and an actuated HVAC vent damper. The system modules all share a common power, communication and control module, and the desired function is achieved through specialized interface circuit boards.**

***Index Terms* — Home automation, gesture recognition, algorithms, capacitive touch, HVAC, traic.**

I. Introduction

With prevalence of smartphones, and the movement toward convenience through autonomy we began to wonder how one's residence could also benefit through increased autonomy. GARVIS aims to make up for where other "home automation" systems fall short. GARVIS is a network of interconnected sensors and actuators that are all connected to a central host and is designed to intelligently automate one's residence rather than merely controlling it with a complicated set of timers. Our sensors are housed in our "Smart Switch", a light switch replacement. Each smart switch will collect data from each room and send it back to our Central Manager. The Central Manager can interpret this data and relay commands to the actuators placed throughout the residence. We have designed two types of actuators, the first being our Load Controller which has analog control over both AC and DC electrical loads. The second type of actuator is our HVAC control, which is charged with controlling the heating/cooling cycles, the HVAC fan, and the damper system. While the primary goal of GARVIS is to keep user interaction to a minimum, we do offer three unique ways of interacting. The first being through a 7 inch LCD touch screen housed within the Central Manager. We have a smart switch to take capacitive touch inputs from the user from each room. Our final user interface method is a motion sensitive glove. The glove can be mapped to specific functions via the Central Manager to control the residence's actuators.

II. System Overview

GARVIS is the combination of a smart home automation system with gesture based control. The goal of GARVIS is to create a home automation system that adapts to the user's lifestyle through adaptive algorithms and gives the user control of their house with multiple interfaces. It's also hoped to achieve this with a design that is plug and play so the user has to do as little work possible to get the system working.

GARVIS consists of 5 major systems. The Central Manager, the Smart Switches, the Load Control, the Vent Control, and the Smart Glove.

The Central Manager is the brains of the operation. This system has a 7 inch LCD screen that allows the user to control their whole house from it. It also collects data from all the Smart Switches in the house in order to make "smart decisions" through adaptive algorithms. For example, turning off the HVAC system when it knows you aren't home. It also processes gestures from the Smart Glove.

The Smart Switch is a device that replaces the standard light switch. Not only does it allow you to turn devices on and off, it allows you to control more than one device at a time as well as allowing you to control the percentage of power being used by them. So you can dim your lights or control the speed of your fans. It does this by having a capacitive touch interface with multiple pads that allows for both single taps and fluid, natural feeling slide motions. It also collects data from sensors that it sends to the Central Manager to process for its adaptive algorithms.

The Load Control is a device that allows control over two standard AC outlets and two DC RGB LED strips. The commands for the level of power to output can come from either the smart switch in the room or from the central manager. This allow the central manager to take a more active role in reducing household power usage.

The Vent Control is a device that allows control over the airflow to each room. With this device, the system can further reduce power consumption by only heating the

rooms that are being used by the resident. This serves as a critical factor for adaptive control.

The Smart Glove is a device that allows us to combine wearable technology with home automation. The user can wear this glove that can control devices in the house by predefine gestures such as swiped your hand across the air or holding up three fingers.

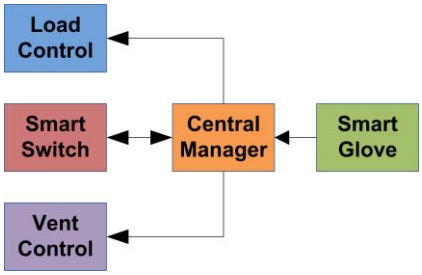


Fig 1. High level system integration chart

III. Central Manager

The Central Manager acts as the control station for GARVIS. It is the destination point for all data and it acts at the main interface for the user to control the system, with alternative interfaces being the Smart Switches and the Smart Glove. The Central Manager is composed of a BeagleBone Black Rev C, a 7 inch touch LCD screen display from 4D Systems, and a custom built cape for I/O and HVAC control.

As the main controller of the system, the Central Manager has three critical responsibilities—data collection, data interpretation, and controlling the system via adaptive commanding. Data is collected as sensor data from any number of Smart Switches, as gesture data from the glove, and through user input from the touch screen. Once data has been obtained, it is processed in order to provide current home environmental information to the user and to create commands to control the home. Furthermore, if the system is in Smart Decision Mode, additional adaptation features are available to control the home. The Central Manager has the ability to send control messages to the Load Controllers to turn on and off home electronics and appliances. While all of this is happening, current home information is easily accessible to the user on the LCD screen as a Qt embedded application.

A. Hardware

The BeagleBone Black Rev C with an ARM Cortex A8 processor was the chosen development board because of its high number of GPIO pins and peripherals, the ability to custom create a cape for the board, and library availability for integrating with the BeagleBone Black hardware. The use of the LCD screen tied up a lot of pins on the development board so it was critical that the BeagleBone Black had more than enough to begin with. With 512MB of RAM, the same amount as the Raspberry Pi B+, it had plenty of RAM for the system's needs.

In order to provide the best user experience, the main interface for user interaction was decided to be an LCD screen. At a minimum it needed to be of a 7 inch size and touch screen. 4D System's 7 inch LCD screen provided the GARVIS system with those features at an 800x480 resolution display. In addition it was very easily set up, with the drivers already installed on the Debian distribution used on the BeagleBone Black. The LCD screen hardware is a cape for the BeagleBone Black which leaves it securely connected at all times but it also has headers for additional capes to be stacked.

A BeagleBone cape was created to mate with the LCD screen and allow the central manager to communicate with the peripheral devices, and control AC and DC loads. The communication is achieved through interface with the half-duplex RS-485 bus, and the load control is achieved through electromechanical relays. For the prototype system created, these relays control demonstration aids to indicate if the HVAC system is heating, cooling or off. In a full implementation, these relays would control household HVAC systems.

B. Backend Software

All Central Manager backend software is housed on the BeagleBone Black running the embedded Debian Linux distribution. Creating the software with Qt on this platform allowed for the most optimal library availability and software functionality for the Central Manager.

A basic overview of the main software components in the Central Manager software architecture is as follows. Integrating the Central Manager software with the UART and GPIO ports of the BeagleBone along with the touchscreen required the use of and IO class, the IO Manager. Constant polling for environmental data and the processing of that data occurred in the Command Creator class. Synchronizing communication for the RS485 protocol between hardware devices occurs in the Status Monitor class. The user interface is managed by the Main Window class of the Qt application. Gesture recognition and glove setup is managed from the Glove API and while housed on the Central Manager, will be explained in great detail in Section VII.

The IO Manager utilizes the singleton design pattern because it is very critical to all other classes. By creating only one instantiation of the IO Manager class, this ensures that all other classes have access to the same data at all times. The IO Manager's responsibility is to control UART communication and house all current home status

data, making it readily available for all other classes. It is able to control UART communication through the use of a separate UART class and an outside library named BlackLib [1]. The UART class provides for a threading scheme to allow for the IO Manager to be able to spin off a read UART thread to constantly poll for new UART data while still carrying out all other functionality.

The next class, the Command Creator, also is a thread class that monitors all data coming in and determines how to process it. The Command Creator interprets raw messages from the Smart Switches around the house to determine what type of sensor data they contain and how to use that data. For example, if the capacitive touch screen was activated on a Smart Switch the Command Creator would receive that information and formulate a Load Controller command message to send out accordingly.

The last fundamental part of the Central Manager backend software is the Status Monitor which is responsible for synchronizing the RS485 communication of the entire system. Through the use of timers and message sending the Status Monitor is able notify other hardware in the system of who is allowed to use the communication line at any point in time.

C. Adaptive Algorithms

One of the notable features of GARVIS is its ability to adapt to certain environmental conditions if the user desires. This is done through the use of adaptation algorithms in the Command Creator. The user is able to setup thresholds for a room—whether they are minimum or maximum temperatures, humidity, or lighting needs. With these thresholds set, whenever the Central Manager receives sensor data from a Smart Switch it compares that data to the user's desired thresholds. If this data is out of bounds from these thresholds then the Command Creator will find the nearest device to control to remedy the situation and send the appropriate Load Controller message. This will turn on or off the device or limit the power given to that device for a dimming effect. For example, if user sets the maximum temperature to 75 degrees Fahrenheit for their bedroom and their bedroom Smart Switch detects a temperature of 80 degrees, the Central Manager will receive this information and check if there is a fan in the room. If so it will turn on the fan by sending a Load Controller message. If not it will command the HVAC system to turn on to cool the room.

D. User Interface

The frontend of the Central Manager consists of a Qt graphical user interface with event driven functionality. Qt is beneficial because it is cross platform, open source, written with C++, and usable in desktop, mobile or embedded applications. Kits can be used to assist with cross platform build and run settings to allow for different back ends to support the same front end of a UI. They configure environmental values for specific devices and tool chains. This ability to run the same Qt application on different platforms gives GARVIS the ability to grow and host the Central Manager on an Android device or home computer with very minimal effort if at a future date the team wants to add remote control to the system.

The UI portion of the Central Manager has four main pages—the homepage, room manager page, setup and configuration page, and help page. The homepage has the current date readily available and average house information displayed. The room manager page allows for the user to select a room, view all of its devices and their current statuses along with environmental information about the room. It also allows the user to manually control any of the room's devices and set thresholds for adaptation. The setup and configuration page lets the user add any number of rooms to the system, to add or remove smart devices to or from any room, and to enable or disable Smart Decision Mode. The last page, the help page, provides the user with some important information for using their GARVIS system.

IV. SMART SWITCH

The Smart Switch is a powerful yet simple data acquisition system and control interface for the home automation system. Since the goal of GARVIS is to make a system that can adapt to the user's lifestyle, the Smart Switch has to gather as much data as possible from each room for the Central Manager. This data includes temperature, humidity, and motion through passive infrared, and ambient light. The Smart Switch also gives the user control of devices in the room such as the lights and the fan or anything hooked up to the outlets of the Load Control.

The Smart Switch hardware fits in a single gang box so any standard light switch in a home can be replaced with a Smart Switch. The only things needed to make a Smart Switch active is power from power lines and 2 wires for the RS485 bus that connects every device together. RS485 was utilized since these wires need to run for multiple feet.

The Smart Switch consists of three custom PCBs. The Front End Board, the Microcontroller Board, and the Power board. The Front End Board is unique to the Smart Switch but the Microcontroller board and the Power board are made to be reused in the Load Control, and Vent Control. This makes for a modular design which allows us

to make new devices for the home automation system if desired. The created possibilities for modular combinations are shown in figure 2, below.

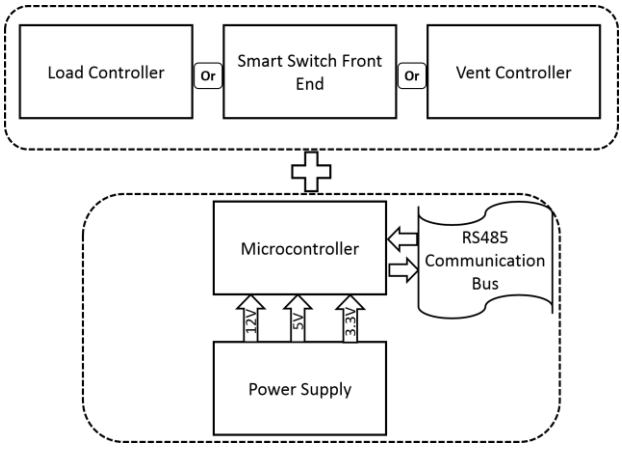


Fig 2. System Peripheral Modularity

A. Front End Board

The goal for the Front End Board is to provide the user with a sleek user interface. To achieve this a custom PCB was created with 9 capacitive touch pads. A big pad in middle that acts as a button and then 4 pads for a slider on the left and 4 pads for a slider on the bottom. It also has 8 RGB backfire LEDs that light up an acrylic plate that diffuses the light. These RGB LEDs will be used for the user to see the device at night and also to convey certain data to the user which is explained further in the software section.

The Front End Board is composed of Atmel's QT1085 capacitive touch chip which handles all the processing and detection of the capacitive touch pads as well as control for the haptic engine. It uses Texas Instrument's TLC5947 LED driver paired with 8 of the HSMF-C113 RGB LEDs, the TI DRV2603 haptic motor driver, and the Avago APDS-9006 ambient light sensor. There is also a passive infrared sensor that physically rests on the board but doesn't actually connect to it. The PIR connects to the microcontroller board.

B. Microcontroller Board

The goal for the Microcontroller Board is to create a reusable custom PCB that can be used as the control for the Smart Switch, Load Controller, and Vent Controller. Along with this it also provides the communication interface for the peripheral devices. The requirements of interfacing with multiple, unique interface boards meant that careful design of the locations of signals and the capabilities of the microcontroller pins that they were driven by, resulting in efficiencies such as the load controller and the front end boards using the same SPI communication lines.

In addition to the microcontroller controlling the devices on the various interfacing boards, it also mounts several sensors. It can measure temperature with the TMP36 temperature sensor by analog systems, allowing it to monitor the temperature in a room (smart switch), an air duct (vent controller), or the running temperature of the load controller, which could overheat at heavy loads. The HIH-4030 humidity sensor is also mounted to this board, but due to the high cost of humidity sensors it would only be mounted on the smart switch. The board also has a set of three mounting holes for the three leads of the EKMC1603 passive infrared sensor by Panasonic.

The microcontroller Board is made of Texas Instrument's MSP430F5529 microcontroller paired up with an external 16MHz crystal oscillator. It has a TMP36 analog temperature sensor from Analog Devices and a HIH-4030 analog humidity sensor from Honeywell. It also has an EKMC1603 passive infrared sensor from Panasonic hat can detect motion up to 12 meters. It uses an RS485 interface to talk to all the other devices in the house. This is done by using Texas Instrument's DS3695 RS485 logic converter. This chip allows us to convert the UART channel in a differential RS485 communication interface.

RS485 was utilized since these wires need to run long distance, and it is better suited for long distance communication than other common protocols. The use of RS485 calls for differential pairs of twisted wires to improve signal quality. To achieve this the group utilizes the use of common CAT5 cables and snap connectors, which allows the system to be connected with widely available and cheap cabling. Each board requires two Ethernet jacks so that all of the peripheral devices are on a single bus without the use of any type of router or signal switch as a central hub.

B. Power Board

In order to allow the modularity of the various interface boards, the power supply needed to be powerful and versatile enough to meet the needs of a diverse set of tasks. This required a 3.3V supply capable of supplying power to the microcontrollers, sensors and the various other digital ICs described, as well as a 5V line for providing power to several parts that were unable to run at 3.3V. A high power 12V line was also needed for moving the motor of the vent controller and supplying the power to the DC loads of the load controller. The 12V line was also to be used for a power line communication interface, but this

communication protocol was dropped and replaced by RS485.

The power distribution hierarchy is as follows and is graphically displayed in Figure 3. The household mains are transformed down to 12VAC, then rectified to ~17VDC. The 12V line is created with the TPS54336 DCDC switching regulator by TI. The 5V line is created in a similar manner, except using a LM43602 DCDC switching regulator by TI. The 3.3V line is generated by using a LMS8117 adjustable low dropout regulator by Texas Instruments.

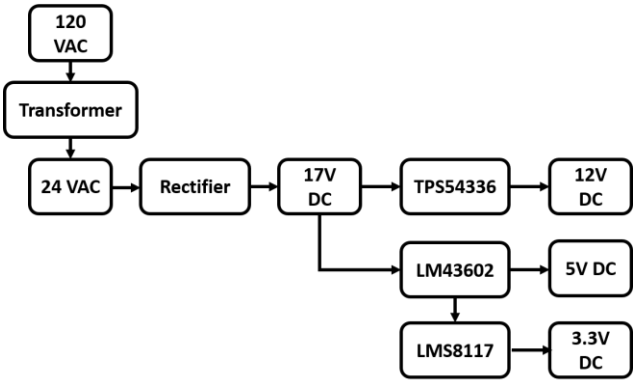


Fig 3. Power Supply Voltage Hierarchy

The use of switching DCDC regulators for the large voltage drops allows the power supply to be both cool and power efficient, both of which are critical to getting reliable temperature measurements from a smart switch.

D. Software

The Smart Switch software is in charge of taking in data from the sensors, process the data from the capacitive touch chip and convert it into commands, manage the LED peripherals on the front end, and handle the RS485 communication interface.

The first thing to consider is the LED management. The colors of the RGB LEDs will tell the user what device is being controlled. For example, red means that light is being controlled, and blue indicates the fan is being controlled. Variations in a pattern of LED flashes will tell the user whether the message is being sent or not.

The communication is also handled by the microcontroller. A half duplex RS485 interface was used because multiple devices needed to give data as a master. But no device can talk at the same time due to bus contention. To avoid this each Smart Switch will have a unique ID. This will be combined with a time multiplexing technique. The Central Manager is going to send enable messages to all the Smart Switches constantly in certain order. These messages will be sent with a separation of 62.5 milliseconds. Whenever a Smart Switch receives an enable message it now has 62.5 milliseconds to talk on the bus knowing no other Smart Switch is going to talk during its turn. Now all Smart Switches can talk on the bus without bus contention. They just have to hold any messages they want to send until they get the enable signal from the Central Manager.

The Smart Switch has three different types of messages it can send to the Central Manager. The first type is a passive infrared message. Whenever the passive infrared has a rising edge or falling edge it sends a message in order for the Central Manager to know when the user is in the room. The next type of message is a capacitive touch message. The message sent depends on what device is selected. Which device is selected depends on what color the LEDs are set to. For example if the LEDs are red then the user is controlling the lights, if the LEDs are blue the user is controlling the fan, etc. The user can change what device he has selected by sliding their finger on the bottom slider. If the user touches the big button, or slides the left slider then a message will be sent. The big button is to turn the device that is currently selected full on or full off. The left slider is to control the amount of power going into a device allowing the user to dim the lights or control the speed of the fan. This interface is shown in figure 4. The last kind of message the Smart Switch can send is a response to a poll message. A poll message is the message that the Central Manager sends the Smart Switch when it wants to know all the analog sensor data. When that message is received the Smart Switch will respond with 3 messages. One with the temperature, one with the humidity, and another one with the ambient light data.

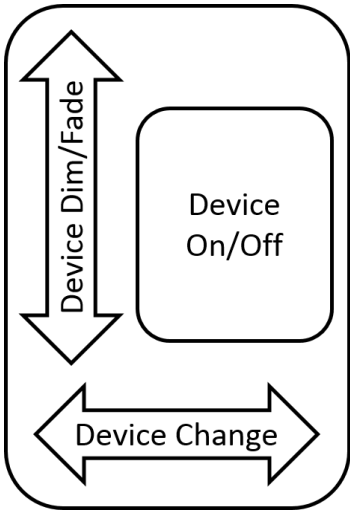


Fig 4. Front End User Interface, Smart Switch

V. Load Controller

The load controller board is designed to be able to control both AC loads and DC loads in an analog fashion. The load controller also monitor's load current so it can determine energy efficiency/ circuit faults. Using a Microchip Technologies MCP9701 temperature sensor, the board temperature is monitored so power can be cut if the temperature reaches an unsafe temperature.

To control the six DC lines, a TLC5947 is used. The TLC5947 is a shift register that takes in values from the microcontroller board and outputs the appropriate PWM signal for each of the six channels into a power MOSFET. The load controller can control up to six 12V DC lines, or two common RGB led strips. To read the current through the DC lines, a low value, high power, shunt resistor is used and the voltage is read across it using an analog to digital converter.

To control the two AC lines, triacs in combination with a zero crossing detector are used. Triacs are a type of a silicon controlled rectifier that allows current to travel in two directions when its base is biased. The triac latches off when the voltage across it drops to zero, which in the case of 60 Hz US mains, happens 120 times a second. We use a zero crossing detector circuit to feed an interrupt into our microcontroller. On the interrupt, a delay is in effect before sending a signal to the triac gate, turning it on and allowing current to flow. The analog control, or the ability to "dim the lights" is established by adjusting the delay time before activating the triac. The longer the delay, the less power delivered. The current of the AC lines is monitored by an ACS714 chip by Allegro which outputs a corresponding analog voltage. To read all the analog voltages from the current monitors, there is an external I2C ADC from Maxim, their MAX11123. AC mains voltage levels do not integrate well with low voltage digital applications. All the AC/DC interactions are galvanically separated, that is we used optoisolators between the two levels. We used two types of optoisolators, the first for the zero crossing detector was one that contained two internal LEDs, so that the current could flow either way which alerts us of both zero crossings in a 60 Hz period.

VI. Vent Controller

The Vent Controller is a digitally controlled, motor driven air damper that allows the central manager to impact the temperature in individual rooms of the house. The mechanical framework of the Vent Controller is a commercially available vent damper manufactured by SunCourt. Devices to open or shut the HVAC vents leading to a room already exist, but most require constant power to a motor to counteract a spring that normally opens or normally closes the vent.

For the Vent Controller a modified SunCourt damper was used with a stepper motor instead of an AC motor so that the vent can be partially open. The stepper motor is controlled with a low cost, off-the-shelf stepper motor driver. The use of a stepper motor allows for a position to be selected, and then the motor driver is powered down to conserve energy.

The Vent Controller is a piece of hardware that would have to be installed as the house is being constructed, or the installation costs would be very large. Other features of the Vent Controllers include the ability to return the temperature of the air in the HVAC ducts to the central manager, which allows for better environmental controls.

VII. Smart Glove

The Smart Glove is an alternative form of controlling the home automation system. It is a non-critical part of the project but was added in order to give the project a twist that would catch people's attention. The idea was inspired from wanting to making a glove to be compatible with the Oculus Rift for virtual reality gaming. Instead it is going to be used to control the home automation system inspired by JARVIS from Ironman. This device gets the kinesthetic data from the user's hand and fingers which get interpreted into gestures by the Central Manager. These gestures are mapped to control certain devices like your light or fan.

The Smart Glove is made up of a custom PCB that is powered from a single cell 850mAh polymer lithium ion battery. The data acquisition part of the PCB is composed of a 3-axis accelerometer to gather linear acceleration data, 3-axis gyroscope to gather radial acceleration data, 3-axis magnetometer to interpret the direction relative to north, and flex sensors to determine the bend angle from every finger on the hand. Each flex sensor also has a dedicated potentiometer on the PCB in order to fine tune the readings. The PCB also carries a microphone for the user to give voice commands. This is included in hardware but is currently not used by the software due to time limitations. All this data is sent wirelessly with the Xbee radio to the Central Manager.

Along with the data acquisition system, the PCB also has external peripherals in order to make the device user friendly. To achieve this the Smart Glove is equipped with a haptic motor, a piezo buzzer, two pushbuttons, and two RGB LEDs in order to give the user as much data possible so it is relatively easy to learn how to use the device. The glove also comes with a SPDT switch that allows the user to turn on and off the glove to conserve battery life.

A. Hardware

The LSM303DLHC chip from STMicroelectronics was used for the 3-axis linear acceleration and magnetometer. This has a ±16g scale with 16 bit resolution for linear acceleration and ±8.1 gauss scale with 16 bit resolution for magnetic field readings. This chip was paired up with the L3GD20 Gyroscope also from STMicroelectronics. This chip has a 2000 degrees per second (dps) scale with 16 bit resolution. Both of these chips have an I^2C interface which made them perfect candidates for the PCB in order to make it as small as possible.

The flex sensors being used are the FSL0055253ST from SpectraSymbol. These are a simple carbon film flex sensor that changes resistance proportional to the bend of the sensor. There is a total of five of these in order to get the bend angle of each finger.

The microcontroller being used is the ATMEGA328 paired up with an 8MHz crystal oscillator. The microcontroller is used to gather all the data from the sensors through I^2C and its ADCs, manage all the peripherals, and send the data to the Xbee through UART.

The battery being used is the 063048 by Unionfortune. This has a nominal voltage of 3.7V, and a capacity of 850mAh. This is paired up with a MCP1825 3.3V LDO voltage regulator by Microchip technology. The Smart Glove has a peak current draw of 100mA and an average current draw of 70mA. Due to the drop out voltage of the regulator, once the battery drops below 3.4V the board starts behaving unpredictable. Therefore the battery life of the glove is around 4 hours, leaving around 40-45% of the battery unused.

The peripherals of the system include the PS1240 Piezo Buzzer from TDK, the CEM-C9745 electret microphone by Challenge Electronics, two RGB LEDs, and a small pushbutton for resetting the system or interfacing with the user.

For wireless transmission it was in the team's best interest to go with the 2.4GHz Series 1 XBee module from Digi. This module uses the 802.15.4 stack. This is a very simple plug and play module with 100 meter range.

B. Glove Software

The goal of the Smart Glove is to gather data as fast as possible for the Central Manager but also to make the device as user friendly as possible. Upon startup the user must lay his hand as parallel to the ground as possible. The microcontroller will gather 10 readings from the accelerometer, gyroscope, and magnetometer during this time in order to calculate an offset. Then the device will go into calibration mode. This is a mode where the user has to put his hand in 3 different predefined positions in order for the Central Manager to gather important data helping it recognize gestures. These positions are flat palm down, fist palm up, and flat palm to the left.

When in calibration mode the RGB LEDs will first turn red letting the user know to go to position 1, then turn blue to go to position 2, and then turn green to go to position 3. Every time the user needs to change the position the piezo buzzer will also beep so it is even easier for the user to know when to change positions. Once calibration is over then 1 RGB LED will remain off and the other will be a heartbeat for the user to know you are in normal operation mode. This means the glove is now gathering data and sending it to the central manager constantly. If the LED that is supposed to be off during normal operation turns red, it means that the communication speed is not meeting the 100Hz requirement.

If the glove is not working well the user can reenter calibration mode by pressing the pushbutton closest to the microcontroller or they can reset the glove by using the pushbutton to the left of that one.

There are a couple of important numbers that need to be noted with the software of the Smart Glove. The microcontroller gathers all the sensor data through I^2C and ADCs. This is packaged into a 34 byte message where the first and last two bytes are for syncing. The first two bytes are always 'S' and 'T' and the last two bytes are always 'E' and 'N'. Therefore the amount of actual data is 30 bytes. This full 34 byte message is sent at a 100Hz to the central manager in order to have good enough resolution of hand movement.

The glove communicates with the Xbee with 38.4 kBaud UART. This baudrate was picked in order to have enough speed to send the whole 34 byte message at a 100Hz without having too much clock divider error since 115200 would have 7.8% clock divider error [2] with the 8MHz clock.

C. Glove API Software

The Glove API is a program that allows for the integration of the Smart Glove with external devices such as desktop computers running Linux or Windows operating systems and the GARVIS Central Manager with further potential applications including the Oculus Rift. The Glove API software is responsible for receiving the raw glove data, converting it into usable sensor data, and interpreting that data to recognize gestures, and turning that user gesture into a command to control various entities. These include mouse control, keyboard control, GARVIS Qt application control, and external application control such as PowerPoint.

In order to recognize the sliding gestures accelerometer data is collected constantly at 100Hz. The sliding motion

creates acceleration in the axis coming out of the palm of the hand, regardless of orientation. By taking this acceleration along with the acceleration due to gravity, the direction, magnitude and duration of the sliding gesture can be recognized. Gravity helps to orient the glove's direction and the palm acceleration shows how long the user is sliding. Other gestures that are recognized are the holding up of a certain number of fingers. This is done by taking flex sensor data and determining which fingers and to what level they are flexed to figure out which number is being gestured by the user. The main gestures the Glove API is recognizing to convert to commands is the holding up of fingers to denote which room to control and sliding gestures to turn on or off devices accordingly.

VII. Conclusion

The GARVIS system began from two project ideas, a smart gesture recognition glove and an adaptive home automation system. It has converted into a hybrid of both ideas to form a very user friendly, energy efficient, and modular system. Through plenty of research and design development, the team was able to create a home automation system capable of taking input from a smart glove, a smart switch to replace a light switch on a wall, and a central control station with a touch LCD screen.

We were able to accomplish our goals for this project by examining the team's strengths and molding our expectation and requirements based on our skill set. After doing this and considering our time and money constraints we determined our needs and wants for the system. Overall we feel that GARVIS was very successful as a usable system and as a learning tool to prepare us for professional engineering projects.

Acknowledgement

We would like to thank our review committee for their time and consideration in their critique of our project. We would also like to thank Dr. Richie for his assistance, advice, and encouragement throughout the design and development process for GARVIS. Thank you to our sponsors—SoarTech for providing us with advice and funding to accomplish our goals as well as Leidos and Duke for their financial contributions.

Additional thanks to Kenneth Schleich of WiTronix for financial assistance and technical consulting throughout the design and implementation process.

References

[1] "BlackLib V2.0 - Main Page." BlackLib V2.0. Doxygen, 2 Oct. 2014. Web. 05 Apr. 2015. <http://blacklib.yigityuce.com/index.html>.

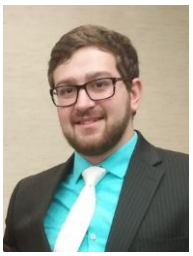
[2] "WormFood's AVR Baud Rate Calculator." WormFood. WormFood, 2013. Web. 08 Apr. 2015. <http://wormfood.net/avrbaudcalc.php>.

Biography

Andres Mujica will be graduating from the University of Central Florida with a Bachelors of Science in Electrical Engineering. During his time at UCF, Andres held a co-op position with Lockheed Martin focusing on firmware design. After graduation Andres will be entering Lockheed Martin's Engineering Leadership Development Program as a full time FPGA firmware engineer.

Sarah Strauss will be graduating from the University of Central Florida with a Bachelors of Science in Computer Engineering. During her time at UCF, Sarah held a co-op position with Lockheed Martin focusing on embedded software. After graduation Sarah will be entering Lockheed Martin's Engineering Leadership Development Program as a full time embedded software developer.

Jackson Schleich will be graduating from the University of Central Florida with a Bachelors of Science in Electrical Engineering. While studying at UCF, Jackson held a co-op with Lockheed Martin. After graduation Jackson will be working with Texas Instruments as a MEMS characterization engineer in Dallas, Texas.



Joshua Illes will be graduating from the University of Central Florida with a Bachelors of Science in Electrical Engineering in August 2015. During his time at UCF, Josh participated in the CWEP program with Lockheed Martin, and later worked as a student intern with Digikey.