Software Formalization

Year: \_\_2019\_\_\_\_ Semester: \_\_Fall\_\_\_\_\_\_ Team: \_\_10\_\_\_ Project: Gesture Controlled Remote for Smart Home\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Creation Date: \_\_\_\_\_10/3/2019\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Last Modified: 10/4/2019

Author: \_\_\_\_Timothy Huang\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Email: \_\_\_\_\_huang983@purdue.edu\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Assignment Evaluation:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item** | **Score (0-5)** | **Weight** | **Points** | **Notes** |
| **Assignment-Specific Items** | | | | |
| **Third Party Software** | 4 | x2 | 8 | IFTTT missing |
| **Description of Components** | 5 | X3 | 15 | Good |
| **Testing Plan** | 5 | x3 | 15 | Good |
| **Software Component Diagram** | 5 | x4 | 20 | Needs formatting |
| **Writing-Specific Items** | | | | |
| **Spelling and Grammar** | 4.5 | x2 | 9 |  |
| **Formatting and Citations** | 5 | x1 | 5 |  |
| **Figures and Graphs** | 4.5 | x2 | 9 |  |
| **Technical Writing Style** | 4.5 | x3 | 13.5 |  |
| **Total Score** | 94.5 | | |  |

5: Excellent 4: Good 3: Acceptable 2: Poor 1: Very Poor 0: Not attempted

General Comments:

*Relevant overall comments about the paper will be included here*

1.0 Utilization of Third Party Software

|  |  |  |  |
| --- | --- | --- | --- |
| Name | License | Description | Use |
| APDS9960 library |  | This is a C source library for the gesture sensor. It was written by [elektronikaembedded](https://github.com/elektronikaembedded)  (GitHub). This open source provides functions to initialize the apds9960 gesture sensor, power on the sensor, and send data from the sensor to the microcontroller we are using. [1] | We will use this library with our I2C library to set up the sensor and send data from the sensor to the microcontroller. |
| STM32F4 HAL Library |  | This is also a C source library for STM32F4. It was written by Majerle Tilen [2]. This open source provides functions to establish communication between master (STM32F4) and slave (APDS9960), as well as reading and writing data to master/slave.  (<https://github.com/MaJerle/stm32fxxx_hal_libraries/blob/master/21-STM32Fxxx_USB_TO_UART/User/system_stm32f4xx.c>) The UART source file in hal library is used for communication between microcontroller and wifi module | We implement this I2C library to work with our APDS9960 and UART for Wifi module. |
| IFTTT |  | This is a website that allows users to customize their own applets. Through the applets, the ESP8266 can connect to smart home devices and control them. | We utilize the applets on IFTTT to connect to smart home devices. For example, turn on and off Philip Hue Light Strip. |

2.0 Description of Software Components

*Define the software/firmware components of your design (i.e. functions, objects, etc.). Differentiate between components being developed by your team, components that are being ported to your project, and components that you intend to use as-is from previous projects or third-party vendors (a table may be a very useful way to convey this information). The use of a function call structure or software component diagram is strongly recommended here.*

The software for our project includes 5 major components: functions for gesture sensor, functions for the LCD, functions for the WiFi module, battery gauge monitoring interface, and the IFTTT applet.

The gesture sensor receives I2C data and the LCD display proceeds to print out the sensor status on through the SPI interface. We also use a battery gauge monitoring IC to monitor the status of the battery life.

2.1 Gesture Sensor I2C Interface:

* [1] typedef struct gesture\_data\_type: This structure stores data received by the gesture sensor. The gesture\_data\_type structure has the following attributes: u\_data[32], d\_data[32], l\_data[32], r\_data[32], total\_gestures, in\_threshold, and out\_threshold. The first four attributes are specifically for the up, down, left, and right gestures detected by the gesture sensor. Each of them can store up to 32 gestures in a FIFO structure. The total\_gestures keeps track of the number of gestures detected since the beginning.
* [1] int apds9960ReadSensor(void): This function is called in main function. It reads the detected gesture and returns the gesture (up, down, left, or right)
* [1] int isGestureAvailable(void): This function was called by apds9960ReadSensor(void). In order for the apds9960ReadSensor() function to read gestures, it has to call this function to make sure the gesture sensor has detected one or more available gestures. This function checks the value in the gesture sensor’s status register and returns a value that tells the apds9960ReadSensor() function whether it has detected a valid gesture or not.
* [1] int readGesture(void): This function is called by apds9960ReadSensor() function when the isGestureAvailable() returns a valid value to the apds9960ReadSensor() function. Inside this function, it processes all the gestures stored in the register by decoding the detected gestures and returns to the apds9960ReadSensor() function.

2.2 LCD Display SPI interface:

* To communicate with the LCD display we utilize the SPI interface. To start the device, we have functions for initializing the SPI device on STM32F4, initializing spi gpios, and initializing the normal display mode.
* To print out information and status of the device, libraries containing different characters and symbols would need to be implemented, with functions that color pixels based on the location and the specific characters needed.
* The spi interface would read the interpreted gesture sensor data and process WiFi module data and print out current status. Interrupt functions based on i2c data and uart data received would be implemented to trigger the printing and messages.

2.3 WiFi module UART interface:

* The wifi module connects our device to internet by generating a wifi signal so that the user could connect our generated wifi to the smart home wifi.
* The microcontroller would trigger the wifi module to start generating WiFi signal from an interrupt function based on the interpreted sensor I2C data. When the function is triggered, the wifi module would prompt the user to connect the device.
* The data would be sent and received through a UART interface. Once the device is connected, there would be functions for each smart home device and depending on the operations the user is trying to achieve, different functions would be used.

2.4 Battery Gauge monitoring i2c interface:

* Since the battery gauge monitor ic uses the same I2C interface as the sensor, we could reuse the basic i2c interface developed for the gesture sensor.
* To tailor it specifically for the ic, we would include functions for specific register initialization and library that includes all the read/write registers.
* When the battery reaches its end of life cycle, it would trigger functions that print out low battery messages on the LCD screen. At other normal operation times the battery status will be approximated and displayed in a corner of the display.

2.5 IFTTT Applet:

* The IFTTT applets connect smart home devices with the WiFi module.
* The applet’s job is to receive instruction from the WiFi module and send it to the designated smart home device.

3.0 Testing Plan

3.1 Gesture Sensor:

* Toggle an LED when a gesture is receives
* Toggle different LED based on the decoded gesture
* Use oscilloscope to make sure the data bus is running as expected

3.2 LCD Display:

* Print out welcome message on the screen
* Check the MOSI signal as operating the display on the oscilloscope
* Write to specific pixels and see how the lcd responds

3.3 WIFI Module:

* Check the UART interface behaviors on the oscilloscope
* Write to specific registers
* Read from registers and see if the data match as expected values

3.4 Battery Gauge Monitoring IC:

* Check the I2C interface behaviors on the oscilloscope
* Write to specific registers
* Read from registers and see if the data match as expected values
* Connect to the battery and print out received battery life values

3.5 IFTTT Applet:

* Check if the WiFi module is connected to the applet
* Check if the designated smart home device has made any adjustments based on user’s requests

4.0 Sources Cited:

[1] S., Sarath. “IR\_Gestrue\_APDS9960” GitHub: <https://github.com/elektronikaembedded/IR_Gesture_APDS9960/tree/master/source/application> [May 17, 2018]

[2] Tilen, Majerle. “Library 09- I2C for STM32F4” Internet: <https://stm32f4-discovery.net/2014/05/library-09-i2c-for-stm32f4xx/> [May 1, 2014]

Appendix 1: Software Component Diagram









