Oswego State University of New York

Automated Home Security [Preliminary Report v0305]

(THIS COVER WILL BE REPLACED WITH DEPARTMENT MANDATED VERSION)

Ashish Kharka, Jonathan Castillo, Nathan Loucks

Submitted in Partial Fulfillment of the Requirement for the B.Sc. Degree,
Electrical and Computer Engineering Department

I Abstract

Home security systems are becoming a staple for home owners in America. The expansion of the embedded systems market over the past few decades has made microprocessors and embedded RF communication commonplace in today's society. Most cell phones today contain several of these devices with WiFi, Bluetooth, GSM, and even satellite navigation right in your pocket. In the home, many people are using "Smart Home" devices such as voice assistants, automated thermostats and light switches. As these devices have gained a significant amount of popularity in recent years, "Smart" home security is also a rapidly growing trend.

There are a plethora of viral videos out there showing footage of criminal activity occurring right outside someone's home, without their home security system they may have never been alerted to any potential danger.

In a study released in 2010[1] by the U.S. Department of Justice, Bureau of Justice Statistics, an estimated 3.7 million burglaries occurred annually over preceding years. About 12% of homes burglarized while the occupant was present resulted in the homeowner facing an offender armed with a firearm.

A security system of any complexity is a good step in the pursuit of ensuring that you and your loved ones are protected home. The autonomous nature of our system has the added advantage of operating in a stand-alone fashion, sparing the homeowner the trouble of setting an alarm or monitoring the sensors them self.

Most modern security systems have sensors that can detect carbon monoxide and motion or glass-break, and record and send live video for the user to view remotely. This provides a safer living environment for the occupants when they're home or away, as well as peace of mind. Major security companies often require a service or subscription that a user pays periodically, as well as relatively complicated installation.

An inexpensive and easy to use home security system would let the user take control of their own home security. Making erstwhile wired sensors operate over a wireless channel would allow for the greatest ease of use for a typical consumer.

Using XBee RF modules and a Raspberry Pi, one can create their own automated home security system with the aforementioned functionalities. The XBee modules are used as communication devices that report sensor data back to a main unit, housing a system that processes the data and responds accordingly.The Raspberry Pi will serve as the main control unit to coordinate the signals coming from each sensor module, as well as transmit or store video depending on user configuration.. The goal of this project is to create an automated home security system that is inexpensive yet delivers the user a range of key capabilities to help protect their home and loved ones from preventable harm.

[1] U.S. Department of Justice Office of Justice Programs Bureau of Justice Statistics, Special Report, National Crime Victimization Survey Victimization During Household Burglary September 2010 NCJ 227379 Shannan Catalano, Ph.D., BJS Statistician, https://www.bjs.gov/content/pub/ascii/vdhb.txt

II Acknowledgments

We would like to express our gratitude to Mario Bkassiny, for his constant support and guidance.

Our sincere thanks goes to Dennis Quill for all his help with parts as well as his expertise on working with Raspberry Pi devices.

Finally we would like to thank the ECE department as a whole for providing the resources necessary to make this research possible despite having to work remotely due to the unique circumstances of the spring 2020 semester. Without the planning and preparation made by the ECE department this project and many others would not have been possible.

III Student Statement

As one of the final steps in the pursuit of our degrees, this project has been a long but rewarding process. The Spring 2020 semester was significantly impacted by the COVID-19 outbreak.

All of us, students and staff, were required to come up contingencies on short notice to facilitate the completion of our projects. As a result we needed to find solutions not only to the problems initially proposed, but also to the new problems created by the aberrant situation.

Our erstwhile well planned timeline was thrown entirely out the window. Luckily, we had been ahead of schedule prior to the transition to remote collaboration. Within a short time we had a tentative plan to bring our system to fruition. Ultimately our end product lacks some functionality that was initially proposed due to a variety of extenuating factors, beyond our control.

Contents

IV Research and Design

Supplying Power

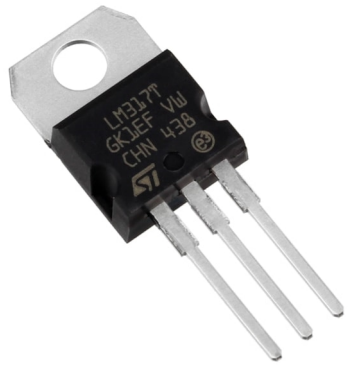


Figure 1: LM317 Linear Regulator (TH)

LM317 Linear Voltage Regulator The LM317 is an IC designed to maintain a constant voltage level. Voltage regulators usually have an input voltage range, but for the LM317 voltage regulator, the range is not specified because it has no ground connection. It has an output voltage range from 1.2 V to 37 V at 1.5 A. Since it is a linear regulator, the output voltage is always lower than input. The LM317 is used in our project to power the XBee modules which require a supply voltage between 2.1 V and 3.6 V. We designed a circuit incorporating the LM317 to output 3.3 V, which is ideal to power our devices.

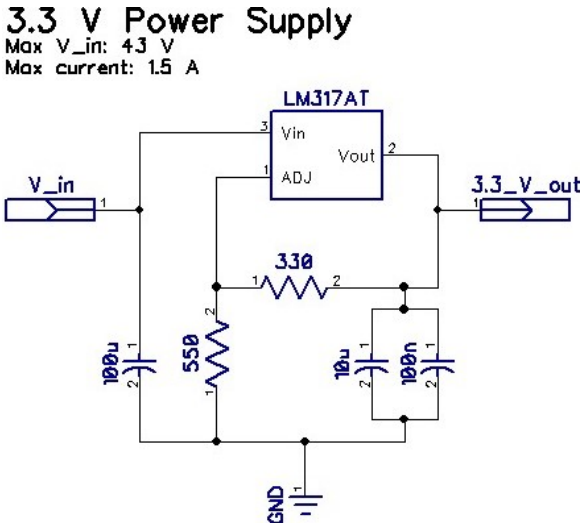
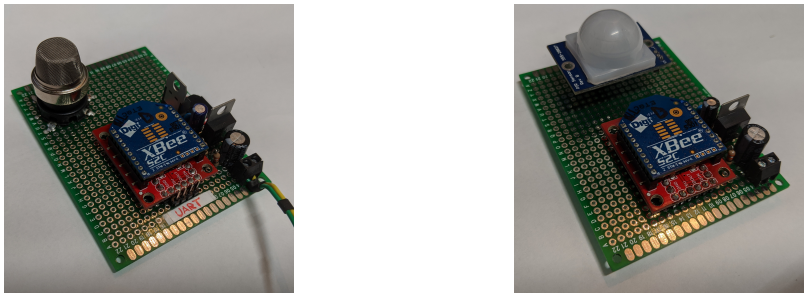


Figure 2: Complete power supply circuit.

As in figure 2, the values we know are $V_o = 1.2$ V, $V_{out} = 3.3$ V and setting R1 = 330 Ω. The value of Iadj is very small hence it can have considered negligible. The value of R2 came to 550 Ω. We then added the C1 = 100 μF, C2 = 0.1 μF and C3 = 10 μF for the improvement of transient response of a system to change from steady state.

LM7805 5V Voltage Regulator To power our devices that require a 5 volt source such as the MQ-6 LP gas sensor, we chose to design a circuit around the LM7805 voltage regulator. This regulator has an input range of 5 to 18 volts, ideal for out 6 volt battery packs, and produces a constant 5 volt output.

Physical Construction of Modules The modules themselves will be constructed on 6x8 cm perf-board and will be contained in a housing. Each module will contain all circuitry necessary including power, data acquisition and transmission.



(a) Gas Module (b) Motion Module

Figure 3: Physical construction of modules

The gas sensor module requires two power supplies, one 3.3 volts to power the XBee itself and 5 volts to operate the gas transducer.

Our system consists of 4 separate modules:

- Coordinator Module - Connects directly to the PC, connects to user interface and controls messages between subsequent end point modules.
- LP Gas Module - A liquid petroleum gas sensor to detect gas leaks in the home. Can detect a wide range of flammable gasses commonly found a typical home.
- Motion Sensing Module - The motion sensing module will detect movement and trigger a system response to suit. The Motion module is stand alone to allow the as much freedom of placement as possible.
- Door Camera/Lock Module - This module, when directed to do so, can take an image of the area as well as lock the door automatically. This is the most complicated module in the system and includes a Raspberry Pi to transfer frames over the local area network.

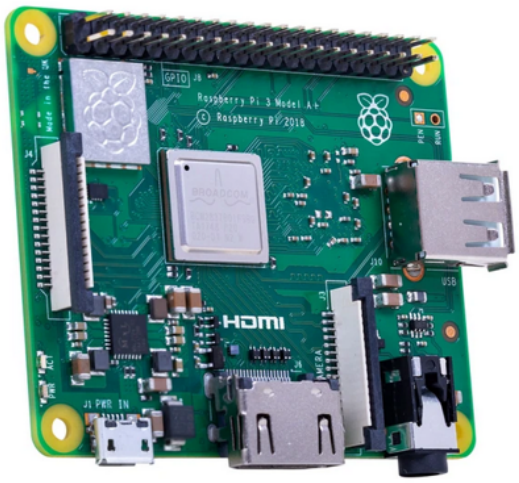


Figure 4: Raspberry Pi 3 Model A+

Raspberry Pi

The Raspberry Pi 3 Model A+ is the latest product in the Raspberry Pi 3 range, weighing in at just 29 g. Like the Raspberry Pi 3 Model B+, it boasts a 64-bit quad core processor running at 1.4?GHz, dual-band 2.4?GHz and 5?GHz wireless LAN, and Bluetooth 4.2/BLE. The dual-band wireless LAN comes with modular compliance certification, allowing the board to be designed into end products with significantly reduced wireless LAN compliance testing, improving both cost and time to market. The Raspberry Pi 3 Model A+ has the same mechanical footprint as the older Raspberry Pi 1 Model A+.

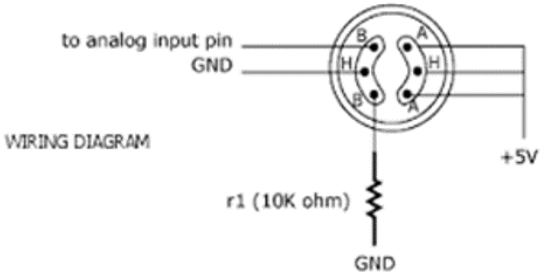
- Processor
  - Broadcom BCM2837B0, Cortex-A53
  - 64-bit SoC @ 1.4 GHz
- Memory
  - 512MB LPDDR2 SDRAM
- Connectivity
  - 2.4 GHz and 5 GHz IEEE 802.11.b/g/n/ac wireless LAN
  - Bluetooth 4.2/BLE
- Data and I/O interfacing
  - Extended 40-pin GPIO header
- Video and Sound

  - 1x HDMI
  - MIPI DSI display port
  - MIPI CSI camera port
  - 4 pole stereo output and composite video port
- Multimedia
  - H.264, MPEG-4 decode (1080p30)
  - H.264 encode (1080p30)
  - OpenGL ES 1.1, 2.0 graphics
- SD card Support
  - Micro SD format for loading operating system and data storage
- Input Power Specifications
  - 5 V/2.5 A DC via micro USB connector
  - 5 V DC via GPIO header

MQ6 LP Gas Sensor

The MQ-6 gas sensor is a highly sensitive sensor designed to detect different atmospheric gases, such as propane, butane and other LPG (Liquid Petroleum Gas), also response to natural gas. It can also detect different some other combustible gases, notably methane. The sensor's conductivity changes proportional to the ambient gas concentration. The relatively low cost and wide range of detection make it an excellent choice for our system. The MQ-6 sensor operates at 5-volts.

(a) MQ6 Liquid Petroleum Gas Sensor Package

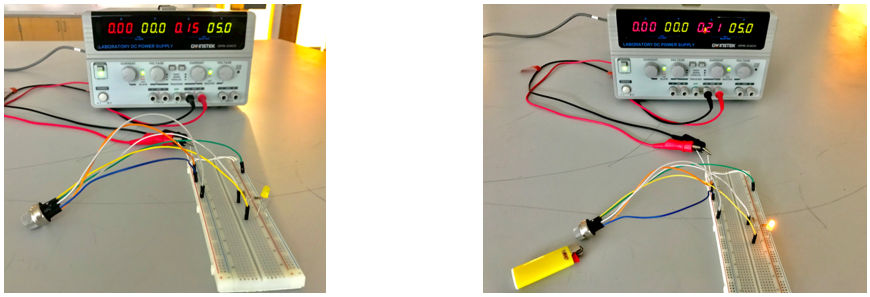


(b) MQ6 Liquid Petroleum Gas Sensor Diagram

Figure 5: MQ6 Gas Sensor

Testing the MQ6 gas sensor

Next, we wanted to test this sensor independently. To do this we first started off with reading the datasheet to find out the appropriate voltage input and pin layout. We found out that it operates at 5 volts with a minimum of 0 volts. We supplied 5 volts from the power supply to our breadboard and we used an LED to our output pin to see if it detects any gas.



(a) LED on via pin 18 (b) LED off via pin 18

Figure 6: Simple LED activation over 802.15.4 (2)

Properly connected, the gas sensor produced quantifiable results. Using a common butane lighter, we tested the sensors ability to detect ambient gases. The LED was able to indicate the presence of LPG, thus we now know our gas sensing components are functional.

Since the XBee modules analog to digital converter requires an input in the range of 1.2 volts, we must reduce the 5 volt output of the gas sensor down to this level to be quantified and processed. To do this a simple voltage divider was designed (1). With voltages constant and one resistor value constant we only need to solve for one unknown (2).

$$V_{\text{out}} = \frac{R1}{R1 + R2} * V_{\text{in}} \tag{1}$$

Solving for R1:

$$1.2 = \frac{5100}{R1 + 5100} * 5 \tag{2}$$
$$R1 = 16150\Omega$$

Python, Primary Programming Language

Python is an interpreted, interactive, object-oriented programming language. It incorporates modules, exceptions, dynamic typing, very high-level dynamic data types, and classes. Python combines remarkable power with very clear syntax. It has interfaces to many system calls and libraries, as well as to various window systems, and is extensible in C or C++. It is also usable as an extension language for applications that need a

programmable interface. Finally, Python is portable: it runs on many Unix variants, on the Mac, and on Windows 2000 and later.

Creating the Wireless Network between the Coordinator and Router modules

We wanted to create a wireless network between the coordinator and router modules this time. Firstly, we started off with soldering our XBee Explorer to connect our router modules on top of them. The reason we chose Xbee Explorer was because it will allow us to solder the input power voltage and the output pins that will be connected to the sensors. We used the application XCTU to help us set up each Xbee as a coordinator or router modules. We first set our coordinator in API mode that will allow us to communicate directly with the sensors via router modules. At the same time, we set the other three router modules to be in a AT or Transparent mode. This will allow sensors to send data directly to the coordinator. We also made sure all the XBee modules used the same Channel and Personal area network (PAN) ID.

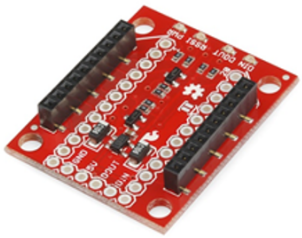
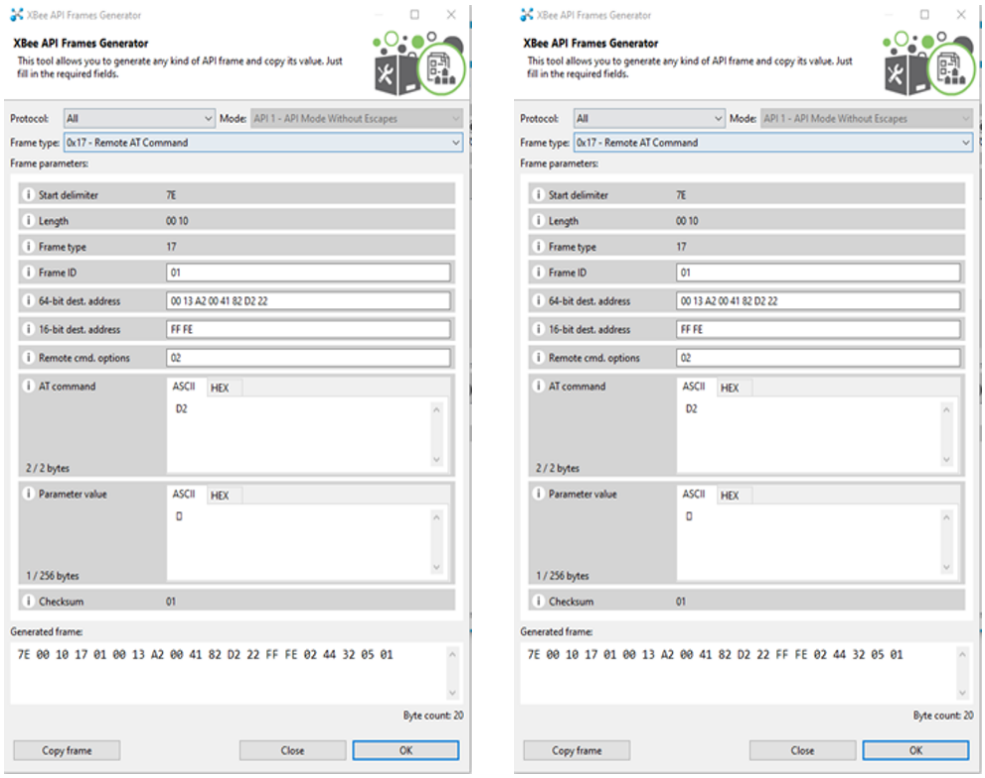


Figure 7: Small XBee interface to be used in each module

Controlling the Router XBee with XCTU Remote AT Command

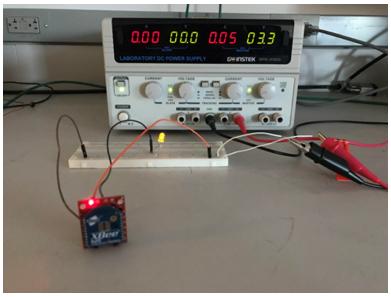
We decided to test our router XBee remotely by sending some AT Commands remotely. We used the XCTU application to first create the Remote AT Command to turn on and off pin 18 of the router Xbee. To see if we receive any output, we put an LED to see if the pin was turned on or not. In the figure below you can see the LED turning on and off via Pin 18 of the Xbee device. The pictures help visualize what the pin is doing corresponding to the LED.



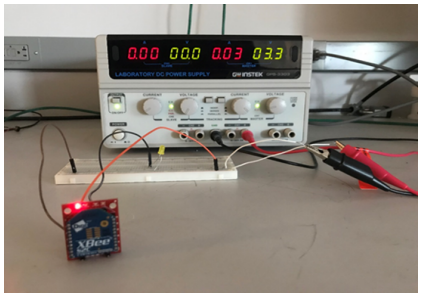
(a) Turn On Pin 18

(b) Turn Off Pin 18

Figure 8: Simple LED activation over 802.15.4



(a) LED on via pin 18



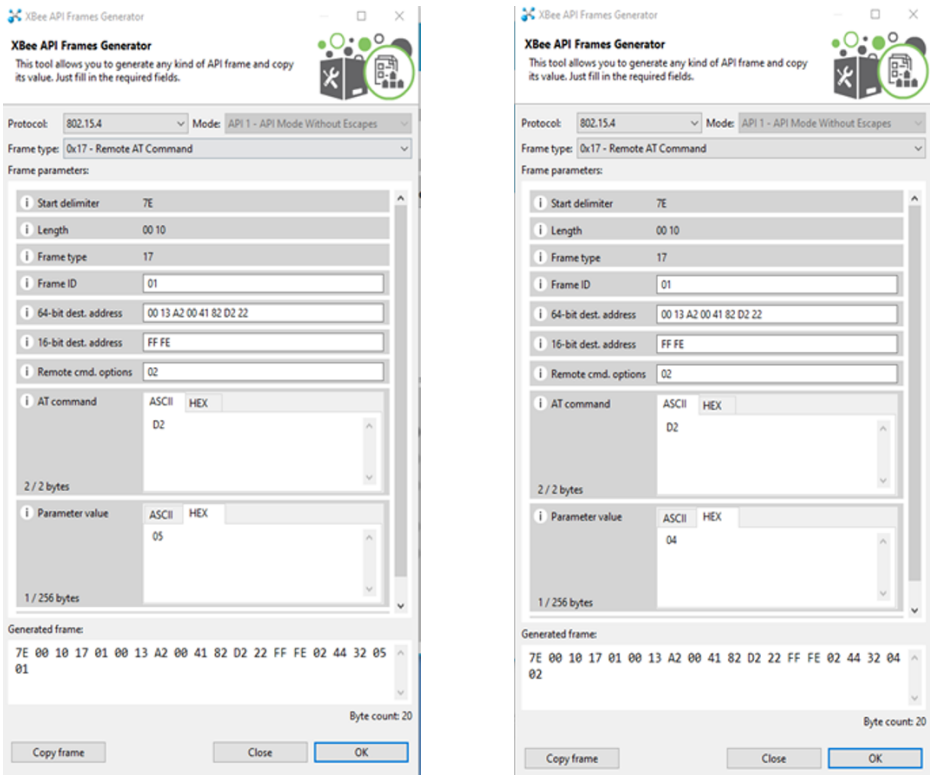
(b) LED off via pin 18

Figure 9: Simple LED activation over 802.15.4 (2)

Controlling XBee module using Python

Since we now know that remote router XBee can be controlled by the coordinator XBee wirelessly, we decided to use python program for the same task. First, we connected our coordinator Xbee module to the computer and set it to API mode. Then we powered up the router module and now we know that it has been connected in a wireless mesh network. We used the XCTU application to generate the same Remote AT Command frame to control the pin 18 of the router module. Then opened the python program to start writing our program in a new script file. API frame generated to Turn on & Off pin 18:

```
#Configure AD2 and DIO2 to control pin-18

import serial
import binascii
import time

ser = serial.Serial(COM3)
ser.baudrate = 9600

#XCTU API Frame remote AT command used to control pin-18
#Power ON
turnOn = "7E 00 01 17 01 00 13 A2 00 41 82 D2 22 FF FE 02 44 32 05 01"
#Power OFF
turnOff = "7E 00 01 17 01 00 13 A2 00 41 82 D2 22 FF FE 02 44 32 04 02"

#Use binascii to convert hex instruction to binary
messageOn = "".join(turnOn.split())
on = binascii.unhexlify(messageOn)

messageOff = "".join(turnOff.split())
off = binascii.unhexlify(messageOff)

#Test write commands, flash led at 5 sec intervals
count = 0
while (count < 10)
   ser.write(on)
   time.sleep(5)
   ser.write(off)
   count = count + 1
```



(a) Turn On Pin 18 (b) Turn Off Pin 18

Figure 10: API Frames created in XCTU

V Problem Formulation

Problem Statement

Home Security systems can be expensive and require monthly payments to have installed and to monitor your homes. These systems can be very inexpensive and should not have to require "professional installers" to implement into your home. There is no way to properly monitor if systems are working correctly in the home you are living in. One way to access that information of your house is through the home security automation. Security not only means being protected from outside danger as one would normally think. However, there are other factors that play into being "Safe". In a home there are systems such as heating, and cooking that can pose danger to tenants if not properly take care of or monitored. If there is a gas leak, then that means that the system is flawed and could cause enough issues to be fatal to those living in the house. Through the security system it makes the user aware of the gas levels in the house and it determines if there is cause for concern/alarm, or if it just means that the tenant is cooking.

List of Figures