IQBox
A Smart Mailbox for Collecting Weather Data and Improving Postal Efficiency and Convenience

William Sussman

Abstract

The Internet of Things is transforming our world. Smart watches, smart speakers, smart TVs, smart thermostats and more are now commonplace. It is curious, then, that mailboxes are not smart. There exist add-on products that notify users when mail is delivered, but I propose an integrated solution: IQBox. Not only does IQBox detect incoming mail, it detects outgoing mail and doubles as a weather station. These features may seem insignificant, but in the aggregate they will enrich meteorological datasets and enable systemic postal improvement. I built a prototype (Fig. 1) that connects to Wi-Fi, magnetically detects changes in the state of the IQBox door and flag, and measures air temperature, pressure, and humidity. I then programmed a local server to handle IQBox reports, and demonstrated that the system is feasible.

1 Introduction

Engineers are notoriously lazy. At university I have a post office (P.O.) box to receive mail, and it is inconvenient to check it. Oftentimes it is empty, so it is not worth checking daily. When I do check it, sometimes it is empty and the trip is wasted; other times it is not, but the mail has been ignored for days (or, admittedly, weeks). Frustrated, I began to devise a smart P.O. box, which I named IQBox, that would notify me when mail is delivered.

Then COVID-19 struck. I was sent home for the remainder of the semester, and both of my parents fell ill. As a result, I was responsible for checking the mail. My pet peeve had followed me home. However, I realized that not only would IQBox alleviate my frustration, it could notify the postal service when my household has outgoing mail, and if the whole neighborhood has IQBoxes, mail delivery routes could be more efficient. I additionally realized that these IQBoxes could capitalize on their fixed location outdoors and function as weather stations.

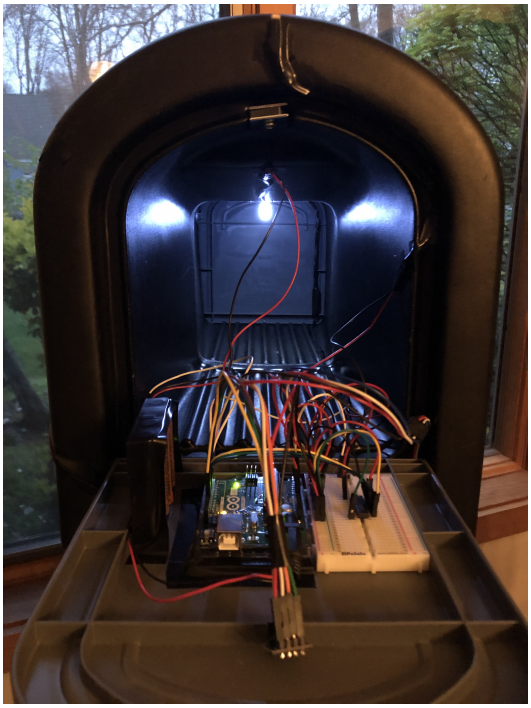


Figure 1: Rear view of IQBox prototype interior.

2 Background

The United States Postal Service (USPS) Risk Analysis Research Center (RARC) issued a 2015 report on "The Internet of Postal Things," which explores "interconnecting the postal network." It envisions several innovations, including a "connected mailbox." RARC writes that "a smart mailbox equipped with sensors would be able to collect and transmit in real-time a variety of postal and non-postal data, such as mail delivery and pick up time, temperature, and data on the external environment." However, the report is only theoretical. In this paper I expand the vision and present practical challenges with solutions.

3 System Design

The rudimentary system consists of two parts: An IQBox prototype, and a local server.

3.1 IQBox Prototype

The IQBox prototype has many components, arranged as shown in Figure 2.

Plastic mailbox: Step2 MailMaster Plus. Most of the other components are fixed to the inside of the back door. A plastic model was chosen to avoid interfering with the magnets and Wi-Fi signals.

Microcontroller: Arduino UNO Rev3. Based on the ATmega328P processor with several analog and digital pins. Two digital pins are configured in software as interrupt pins with internal pullup resistors. Supplies 3.3V to all powered components, including the LED via the shift register. Separating the power supply remains future work.

Wi-Fi module: ESP8266. Acts as a modem controlled by AT commands. Requires a 3.3V source.

Weather sensor: BME280. Measures air temperature, pressure, and humidity. Requires a 3.3V source. Fixed below main compartment of mailbox for air exposure. Can communicate via I^2C or SPI. I^2C was chosen to minimize wires and avoid level-shifting.

Magnets: COM-08914. One is fixed to the front frame of the mailbox, the other is fixed to the tip of the mailbox flag.

Non-latching Hall effect sensors: AH1815. Each pulls its output line low near a magnet. One is fixed to the rim of the mailbox front door, the other is fixed to the side of the mailbox near the tip of the mailbox flag when down.

Photoresistor: SEN-09088. Fixed to the back frame of the mailbox. Forms a voltage divider with a 4.7kΩ resistor. The center node voltage increases as illuminance increases, and is measured on a microcontroller analog pin with 10-bit precision.

Diffused white LED: COM-11121. Fixed to the interior roof of the mailbox. In series with a 4.7Ω resistor. Controlled by the shift register.

Shift register: SN74HC595. Extends the microcontroller digital pins and controls the LED.

Battery pack: PRT-09835. Holds four AA batteries. Powers the microcontroller.

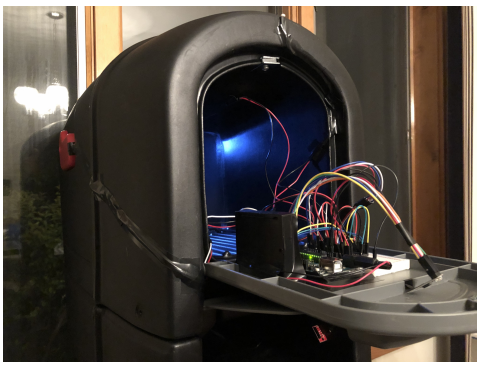
After initializations, the microcontroller turns off all peripherals, particularly the Wi-Fi module if on, and goes to sleep. When the mailbox door or flag changes state, a Hall effect sensor detects the presence/absence of a magnet and toggles its interrupt pin, which wakes up the microcontroller.

If the IQBox door is changed to open, mail is either being delivered or retrieved. These cases cannot be distinguished, but the resident knows when mail is retrieved, so a "you've got mail" notification is generated regardless. Before it is sent, the weather sensor measures the air temperature, pressure, and humidity, and the reading is packaged with the notification. Additionally, the LED is turned on if the illuminance is sufficiently low. It is turned off when the IQBox door is changed to closed, or when 60 seconds elapse.

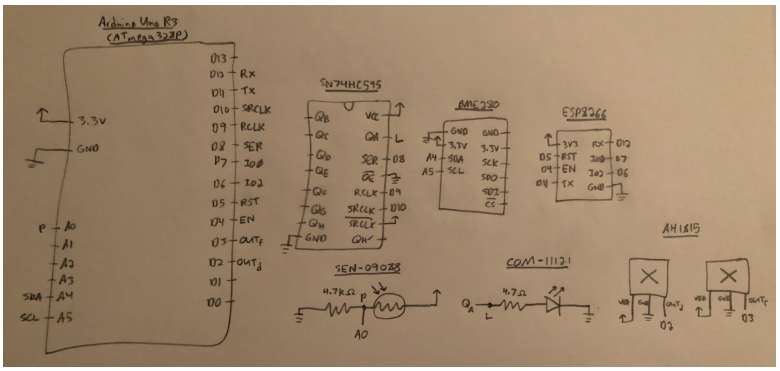
If the IQBox door is changed to closed, it was opened a moment earlier, so a notification and weather reading are not necessary. If the LED is on, it is turned off.

If the IQBox flag is changed to up, the resident opened the IQBox door a moment earlier to deposit outgoing mail, so a weather reading is not necessary. However, the postal service is interested in the state of the flag, so a notification is generated.

If the IQBox flag is changed to down, the mail carrier opened the IQBox door a moment earlier to collect outgoing mail, so a weather reading is not necessary. Again, however, the postal service is interested



(a) Physical arrangement.



(b) Electrical schematic diagram.

Figure 2: Arrangement of IQBox prototype components.

in the state of the flag, so a notification is generated.

3.2 Local Server

The local server is considerably less complex than the IQBox prototype. It is a Raspberry Pi 4 with 2 GB RAM and 16 GB flash memory, connected to my basement router with a Cat 5e Ethernet cable (Fig. 3). Its eth0 IP address was configured to be static 192.168.1.250, and it runs a program that waits for connections and interprets received information.

Figure 3: Raspberry Pi server connected to router.

4 Results

The Wi-Fi module initially did not work. Its default UART baud rate was 115200, but the microprocessor was incapable of reading so fast. The result was corrupted output from the Wi-Fi module. Echoing is on in Figure 4a, and even the simplest command "AT" could not be read back accurately. Fortunately I was able to use the microprocessor's hardware serial as a passthrough to change the rate to 57600, which resolved the issue (Fig. 4b).

Once the microprocessor could communicate with the Wi-Fi module, the next step was to connect to Wi-Fi. Figure 5a shows a sequence of successful Wi-Fi commands. First "AT+CWLAP" commands the Wi-Fi module to list available APs, and it finished in 2.30 seconds. Notice that my "[redacted]One" router appears twice, with relatively high RSSI values of -53 and -51 (the difference is likely due to measurement errors), but the similarly-named 5GHz SSIDs do not appear. This is because the Wi-Fi module can only receive 2.4GHz signals.

Then "AT+CWJAP" commands the Wi-Fi module to join password-protected "[redacted]One2.4GHz," and it finished in 5.00 seconds. Successful connection is confirmed by the next three commands. "AT+CIFSR" shows that the Wi-Fi module was assigned an IP address of 192.168.1.122; "AT+PING" shows that the Wi-Fi module could reach an In-

```
Command: AT
Response: AR

OK

Command: AT+CWMODE_CUR=1
Response: AT+CU?T?5UI??j

OK

Command: AT+GMR
Response: AR+GMR
AT vY??Z?K?r?r?r?BRd????????x?H?Jj
SDK wersioo:1.5.4.1(39,?NX&??H(???????????
D 43
```

(a) Corrupted output from the Wi-Fi module.

```
17:37:32.375 -> AT+GMR
17:37:32.375 -> AT version:1.2.0.0(Jul  1 2016 20:04:45)
17:37:32.375 -> SDK version:1.5.4.1(39cb9a32)
17:37:32.408 -> Ai-Thinker Technology Co. Ltd.
17:37:32.408 -> Dec  2 2016 14:21:16
17:37:32.408 -> OK
```

(b) Accurate output from the Wi-Fi module.

Figure 4: Before and after resolution of rate issue.

```
17:39:40.995 -> AT+CWLAP
17:39:43.119 -> +CWLAP:(3,"DIRECT-33-HP ENVY 5540 series",-54,"b0:5a:da:[redacted]",1,-6,0)
17:39:43.157 -> +CWLAP:(0,"[redacted]ne2.4GHzGuest",-53,"66:38:e0:[redacted]",1,-37,0)
17:39:43.157 -> +CWLAP:(3,"[redacted]ne2.4GHz",-51,"60:38:e0:[redacted]",1,-39,0)
17:39:43.157 -> +CWLAP:(4,"jpmnkmwifi-A403E0",-89,"54:b8:0a:[redacted]",1,-34,0)
17:39:43.191 -> +CWLAP:(4,"samsline",-87,"00:8e:78:[redacted]",1,-11,0)
17:39:43.191 -> +CWLAP:(3,"HP-Print-89-Photosmart 6520",-94,"c4:34:6b:[redacted]",1,-26,0)
17:39:43.191 -> +CWLAP:(3,"Home",-83,"18:90:88:[redacted]",1,-4,0)
17:39:43.225 -> +CWLAP:(3,"Home",-81,"4c:01:43:[redacted]",1,-7,0)
17:39:43.225 -> +CWLAP:(4,"CleverMonkey_RE",-88,"f8:7b:8c:[redacted]",1,-7,0)
17:39:43.225 -> +CWLAP:(3,"ngHub_319443N80764C",-90,"04:a1:51:[redacted]",1,-14,0)
17:39:43.258 -> +CWLAP:(3,"tmsphqgp",-90,"14:dd:a9:[redacted]",6,-36,0)
17:39:43.258 -> +CWLAP:(4,"D20E80",-87,"b0:c5:54:[redacted]",6,-22,0)
17:39:43.258 -> +CWLAP:(3,"FiOS-877W0_2GEXT",-58,"a0:63:91:[redacted]",10,-39,0)
17:39:43.258 -> +CWLAP:(3,"FiOS-877W0",-78,"48:5d:36:[redacted]",10,-12,0)
17:39:43.295 -> +CWLAP:(4,"FBI TEAM 1",-82,"00:3e:5d:[redacted]",11,-21,0)
17:39:43.295 -> +CWLAP:(3,"TVNP",-81,"a6:40:a0:[redacted]",11,1,0)
17:39:43.295 ->
17:39:43.295 -> OK
17:40:04.026 -> AT+CWJAP="[redacted]ne2.4GHz","[redacted]"
17:40:06.246 -> WIFI CONNECTED
17:40:07.236 -> WIFI GOT IP
17:40:09.029 ->
17:40:09.029 -> OK
17:40:20.435 -> AT+CIFSR
17:40:20.435 -> +CIFSR:APIP,"192.168.4.1"
17:40:20.469 -> +CIFSR:APMAC,"de:4f:22:[redacted]"
17:40:20.469 -> +CIFSR:STAIP,"192.168.1.122"
17:40:20.469 -> +CIFSR:STAMAC,"dc:4f:22:[redacted]"
17:40:20.469 ->
17:40:20.469 -> OK
17:40:41.010 -> AT+PING="8.8.8.8"
17:40:41.044 -> +14
17:40:41.940 ->
17:40:41.940 -> OK
17:41:00.217 -> AT+CIPDOMAIN="google.com"
17:41:00.217 -> +CIPDOMAIN:172.217.11.46
17:41:00.217 ->
17:41:00.217 -> OK
```

(a) Output from Wi-Fi commands.

```
17:41:56.695 -> AT+CIPSTART="TCP","192.168.1.250",3480,0
17:41:56.695 -> CONNECT
17:41:56.695 ->
17:41:56.695 -> OK
17:42:07.380 -> AT+CIPSTATUS
17:42:07.380 -> STATUS:3
17:42:07.380 -> +CIPSTATUS:0,"TCP","192.168.1.250",3480,30854,0
17:42:07.414 ->
17:42:07.414 -> OK
17:42:53.470 -> AT+CIPSEND=13
17:42:53.470 ->
17:42:53.470 -> OK
17:42:53.470 -> >
17:43:00.725 -> busy s...
17:43:00.725 ->
17:43:00.725 -> Recv 13 bytes
17:43:00.725 ->
17:43:00.725 -> SEND OK
17:43:16.738 -> AT+CIPCLOSE
17:43:16.738 -> CLOSED
17:43:16.738 ->
17:43:16.738 -> OK
17:43:50.651 -> AT+CWQAP
17:43:50.651 ->
17:43:50.651 -> OK
17:43:50.651 -> WIFI DISCONNECT
```

(b) Output from network commands.

Figure 5: Successful operation of the Wi-Fi module.

ternet server and finish in 0.93 seconds; and "AT+CIPDOMAIN" shows that the Wi-Fi module could obtain the IP address of a domain name almost instantly.

Now connected to my router, the Wi-Fi module could connect to a server. The local server started listening for connection requests on port 3480, and the sequence of commands in Figure 5b was executed. "AT+CIPSTART" requests a connection with the local server on remote port 3480, which was accepted and ready almost instantly; "AT+CIPSTATUS" shows that the connection was successful, and that the local port was 30854; "AT+CIPSEND" allocates a buffer, and "Hello, world!" was sent to the local server almost instantly; finally, "AT+CIPCLOSE" and "AT+CWQAP" respectively ended the connection with the local server and disconnected the Wi-Fi module from the network.

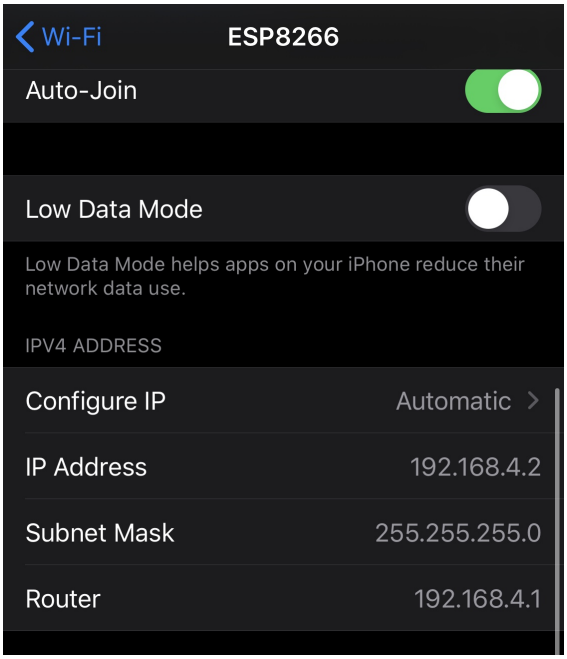
With these commands, the IQBox prototype successfully sent notifications and weather data to the local server. For one test I opened the IQBox door, and a few seconds later the local server received that the air temperature was 70.1°F, the barometric pressure was 102.0 kPa, and the relative humidity was 37.0%. The test was conducted indoors, where the thermostat read 70°F. The pressure and humidity could not be independently verified, but the results are reasonable (in particular, the pressure was approximately 1 atm). Notifications of IQBox flag state changes were also successfully received by the local server.

The Wi-Fi module could also act as an AP. This can allow the resident to configure the IQBox. Figure 6 shows a successful early test.

"AT+CWSAP_DEF" sets the SSID to "ESP8266," and "AT+CWMODE_DEF" enables AP mode. At first "AT+CWLIF" shows that no devices are connected, but after joining with a smartphone, it shows that a device with IP address 192.168.4.2 is connected. This is confirmed on the smartphone.

```
17:37:56.831 -> AT+CWSAP_DEF="ESP8266","",6,0,1,0
17:37:56.831 ->
17:37:56.831 -> OK
17:38:11.324 -> AT+CWMODE_DEF=3
17:38:11.324 ->
17:38:11.324 -> OK
17:38:55.810 -> AT+CWLIF
17:38:55.810 ->
17:38:55.810 -> OK
17:39:30.019 -> AT+CWLIF
17:39:30.019 -> 192.168.4.2,b0:19:c6:[redacted]
17:39:30.019 ->
17:39:30.019 -> OK
```

(a) Output from AP mode commands.



(b) Corroboration of smartphone IP address.

Figure 6: The Wi-Fi module acting as an AP.

5 Conclusions & Future Work

Several lessons can be learned from the rudimentary IQBox system. It is proof that a smart mailbox as specified in this paper is feasible. However, some issues must be addressed. Great care is taken in the software to minimize power consumption. Still, a battery pack will not last the lifetime of the IQBox, especially with the LED. Solar recharging would solve this issue, but remains future work. As noted in section 3.1, separating the power supply from the microcontroller also remains future work.

While the Wi-Fi module works as a modem, it is inefficient to use AT commands. The next iteration of IQBox will be equipped with a microcontroller with on-board Wi-Fi capability. It must have at least two interrupt pins and enough analog and digital pins to control the other components. This will also enable over-the-air updates.

Section 4 demonstrated that the IQBox can act as an AP. A future iteration will use this functionality to obtain Wi-Fi credentials from the resident, which are currently constants in the software. Additionally, a web server that forwards notifications and maps the weather data and flag states will be substituted for the local server.

Finally, the IQBox components will be integrated into the mailbox, and many IQBoxes will be distributed for testing. If successful, I will contact USPS for a large "Internet of Postal Things" trial.

6 Acknowledgments

I would like to thank Professor Wenjun Hu, Professor Rajit Manohar, and R. Ivan Zelaya for their mentorship. I would also like to thank Berkeley College for the Robin Berlin Fellowship, which will support the continued development of IQBox.

7 References

The Internet of Postal Things, August 3, 2015. `https://www.uspsoig.gov/sites/default/files/document-library-files/2015/rarc-wp-15-013_0.pdf`