**Project:**

**Energy Monitoring and Management System**

**Project Record:**

**Wifi Module**

**Design: Electrical**

**Version 1.0**

**12/14/18**

**Bennett Andrews**

**Zachery Holsinger**

**David Williams**

**Version History**

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Description/Notes** |
| **1** | **12/6/18** | **Initial Record** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Revisions Since Previous Version**

None since this is the initial version

**Table of Contents**

[**Abstract**](#_gjdgxs) **4**

[**Context**](#_md6671iv03md) **4**

[**Design**](#_esqtswrajgbz) **4**

[**ESP8266:**](#_m6wzscg7diug) **6**

[**PIC24F16KL302:**](#_3rqyr2i70eff) **6**

[**Elegoo Uno:**](#_1a0nhuvnw3mk) **7**

[**Future Plans**](#_9562dlkuc5u5) **8**

[**Conclusion**](#_lnxbz9) **8**

[**References**](#_pbfiesuml8un) **9**

# **Abstract**

Concern was given over the security of our current meter. One reason for this was that the password into the UI box provides little privacy. Aside from privacy, the meters can also be hard to access or tedious if a large sum of meters are present on site. The idea was then given of wireless communication to the meter. Wireless communication was decided to be transmitted through Wifi because of the teams familiarity on the subject and its simpler integration with computers. Research was conducted on materials and communication, and the ESP8266 was chosen for the wireless front-end of the module. The current prototype has a completed TCP and UART protocol issuing data into and out of the module.

# **Context**

The Wifi Module is not critical for the general function of the meter, however, it is just as important. The implementation of this board allows users access without going to confined spaces where the meter is mounted by monitoring the meter through Wifi from the comfort of their office. The wireless connection also adds a layer of security onto the meter. It removes the past problems of curious bystanders watching and repeating admin passwords to increase their personal allotment of energy. In the future, we plan to write software to control multiple meters simultaneously, and to log the data received for further analysis.

# **Design**

A main goal was to understand Universal Asynchronous Receive Transmit (UART) communication and the SPI communication within the existing meter to establish a wireless external link to the meter in a self-contained ‘Wifi Module’. Within the module, an ESP8266 receives a TCP connection from a computer (wirelessly) and relays the string to a PIC24f08KL302 (Pic24) via UART which sends it to our command board via SPI (See Fig 1 and Fig 2 block diagram). The module is also designed to take the response of the command board and use the reverse process to push that response to the user in the TCP connection. During the summer of 2016 the team constructed a basic code that uses circular buffers to communicate the UART and SPI without causing data loss. The original circuit began with a Pic18 (the same function as the Pic24, but with 18 GPIO pins instead of 24), but due to premade code and the limitation of a single UART port, the Pic24 became the weapon of choice.

Wiring Diagram:

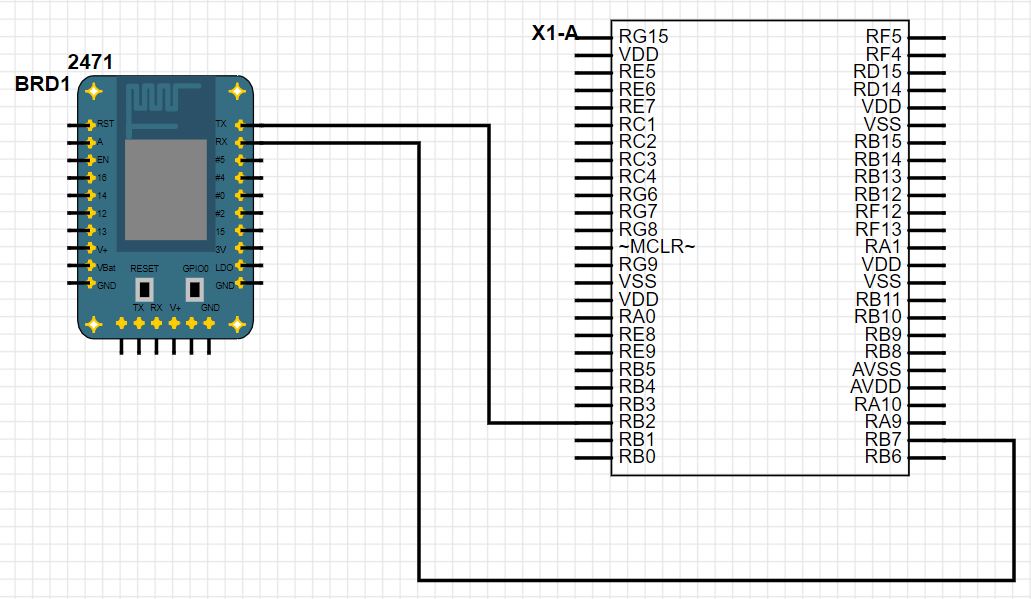


Fig 1: UART Communication Between ESP8266 and Pic24

The wiring for communication involves only two ports, the Tx and Rx ports of each device. The Tx port transmits data to the Rx port of the second device. Tx always goes to Rx, and Rx to Tx, that way the other receives what the first is transmitting. The Pic24 has two UART ready communication (so 4 ports, or 4 pins). Fig 1-1 shows the first UART with the ESP8266. In addition to the wiring diagram, the PicKit 3 Debugger was used to interface with the Pic24, MPLABX IDLE software was used to program the Pic24 through the PicKit3. Arduino software was used to monitor the UART communication going into the Elegoo Uno via the second UART.

### 

**Command**

**Board**

**ESP8266**

**Networked**

**Computer**

**PIC24F08KL302**

**Wifi Module**

**TCP**

**UART**

**SPI**

**Fig 2: Communications Block Diagram**

### ESP8266:

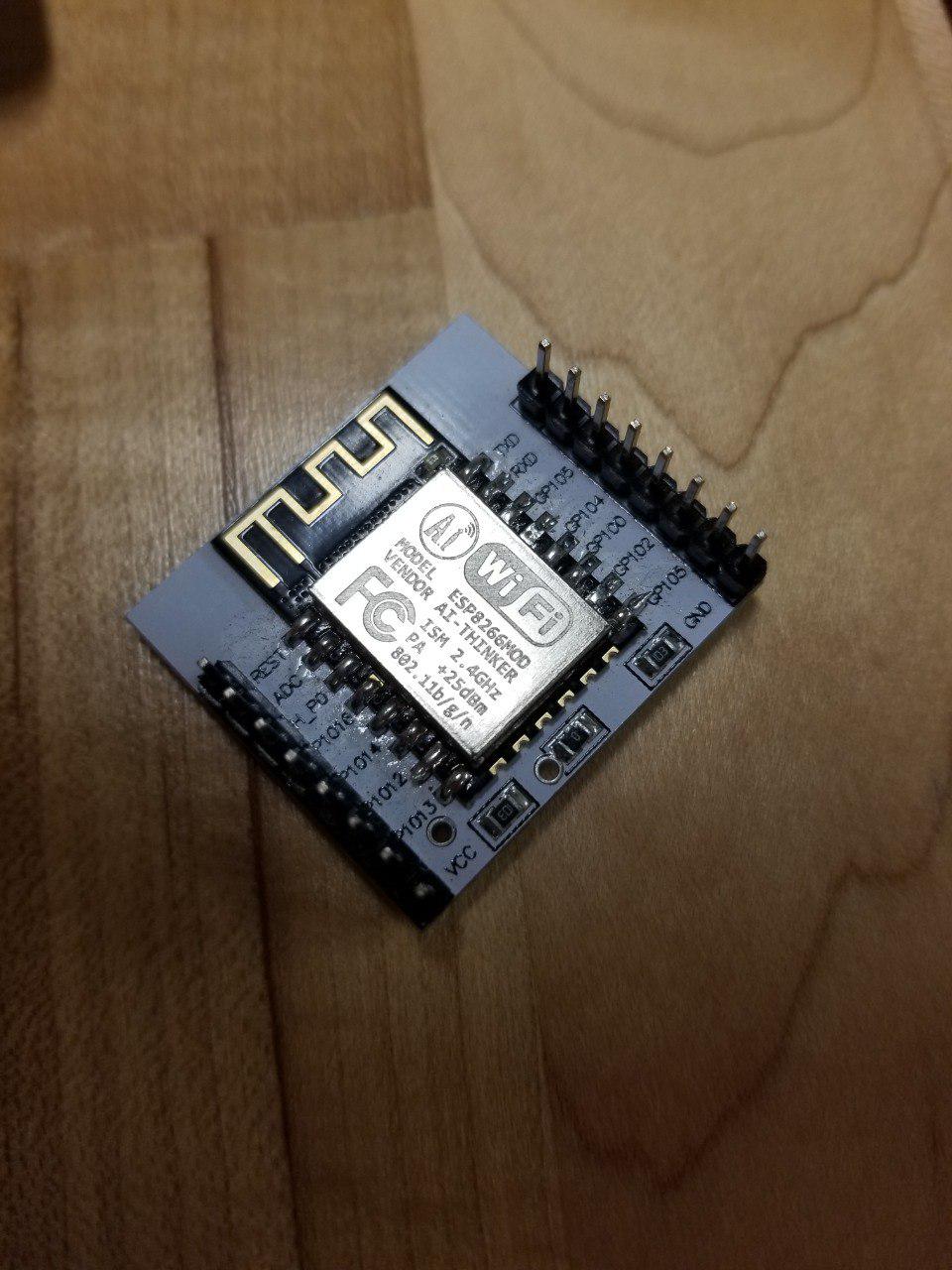
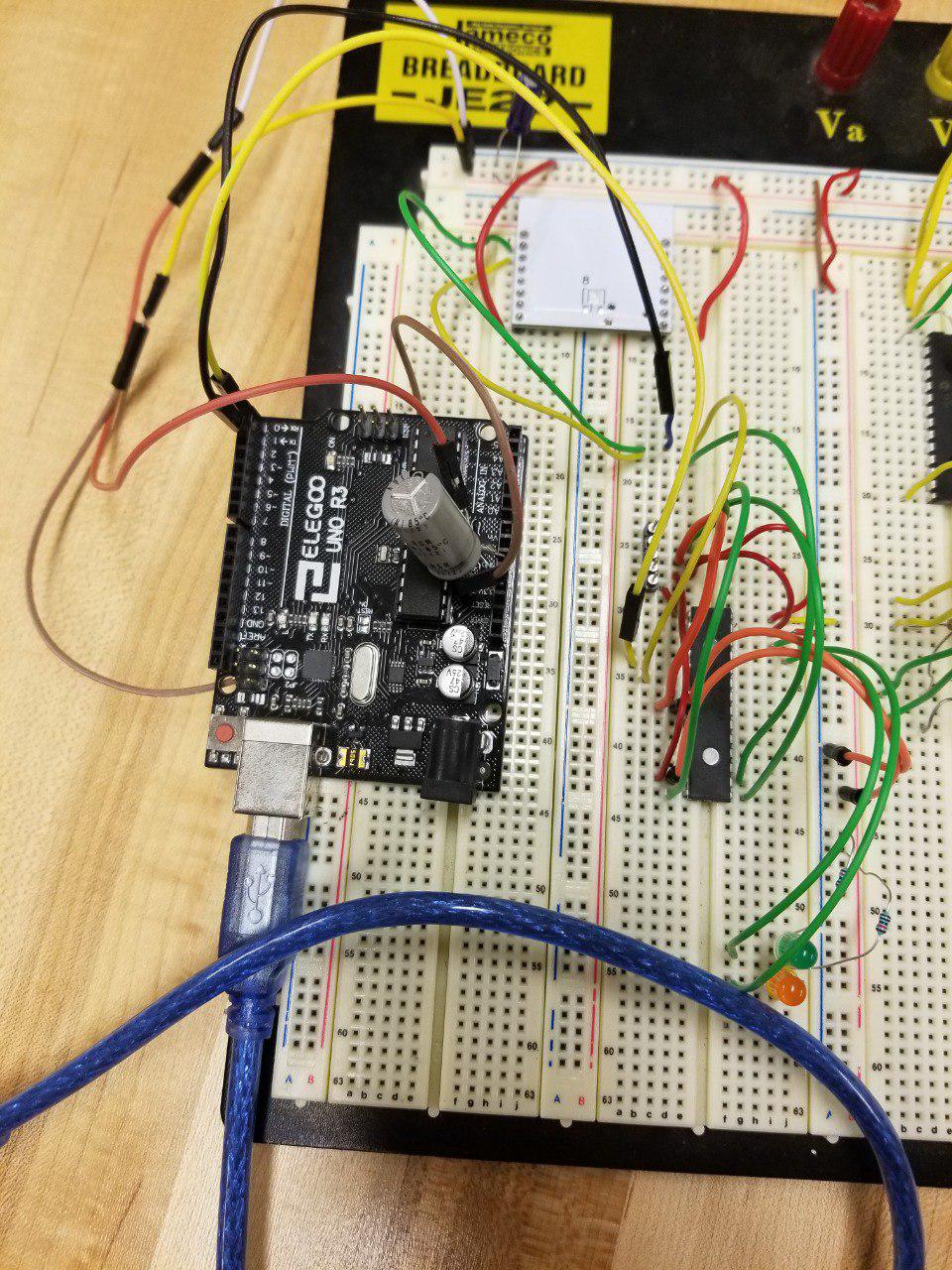


Fig 3-1: ESP8266

The ESP8266 is a 16-bit single core microprocessor capable of establishing multiple TCP and SSH connections through Wifi. It has the ability to broadcast its own network, or connect itself to an existing network. It is how the meters will communicate wirelessly to an admin computer. It communicates information through UART. Though it has the capabilities for SPI communication, it was documented previously that the baud rate was not reliable, and data was unpredictably lost during communication.

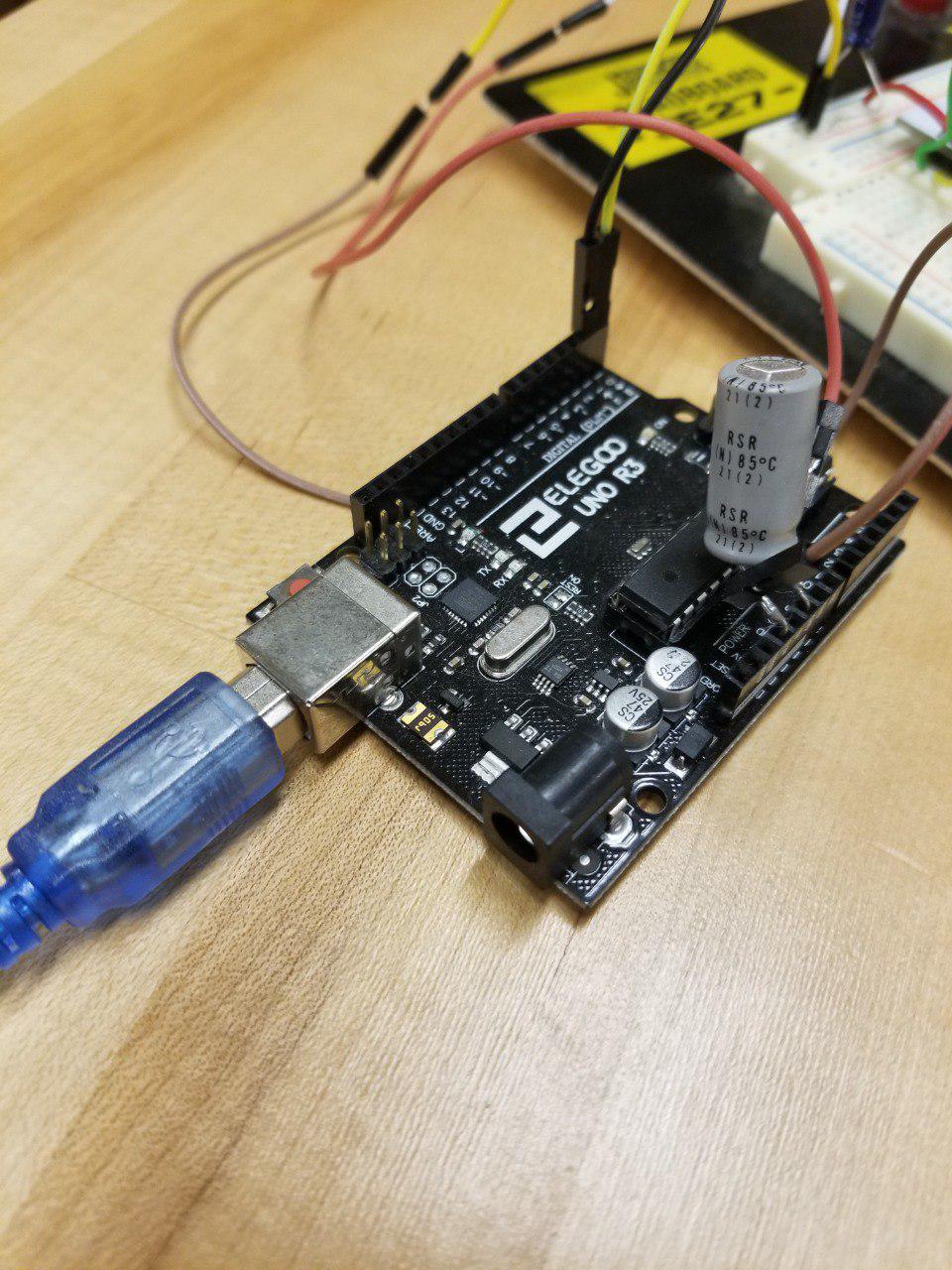
### PIC24F16KL302:



**Fig 3-2: PIC24F08KL302**

The PIC24F16KL302 is another 16-bit microprocessor, programmable by a PicKit 3 debugger via computer. It has a much more stable SPI communication protocol, and is much easier to transmit SPI inside of the code using circular buffers. The Pic24 is capable of both UART and SPI, making it the perfect go-between for information to be passed between the ESP and the command board. Because of its two UART ports, data is echoed back out through a second port into an Elegoo Uno, which can be viewed on the arduino software serial monitor from a computer. This gives the programmer a way to sniff and see if data is being lost in transmission or not.

### Elegoo Uno:



**Fig 3-3: Elegoo Uno**

The Elegoo Uno behaves exactly like an Arduino Uno, which in our case is useful for supplying a single 5V supply to power both the Pic24 and the ESP8266 as well as UART ports to sniff the transmission between the two chips. This microcontroller will not be implemented in the finalized design, it is used for debugging purposes only. It gives the user minimum setup (plugging in one usb) while maximizing function.

### 

### 

**Operation Procedure (See Fig. 2)**

When the ESP and Pic24 initialize, the Pic begins a set of prewritten instructions that configure the ESP for wireless TCP connection. Though these instructions could be programmed into the ESP, it will be exponentially easier to debug once on a printed circuit board in the meter. The instructions begin by first resetting the ESP to ensure a fresh start, then feeding it the current WiFi SSID and password for the network it will be sending information through. Right now that information is stored inside of the Pic, but a future goal is to have the WiFi network reprogrammable on the user interface. Once the pic receives a confirmation through the ESP chip (it prints “WIFI CONNECTED, GOT IP”) the ESP prints its local IP address. This is useful because it is the address the admin will use to establish a TCP connection. The Pic then sets the ESP to receive multiple TCP communications, and toggles off then on the TCP connection of the ESP. This removes any erroneous bugs from a previous cycle. Once all that has been completed, the ESP is ready to start receiving communication! The SPI connection does not require any setup as it is acting as slave to the command board.

## **Future Plans**

The main goal right now is to finish off the SPI communication into the command board of the meter, and turning the breadboard prototype into a printed PCB design. Another short-term goal is to have users see visible WiFi networks, set the SSID, and set the password of the WiFi network through the front-end user interface. Without losing perspective, our general goal is to take this basic wireless communication and expand upon it. The module has developed into software and hardware sides. Progress has made in the creation of user friendly monitoring software for administrator computers. A hurdle is what language to choose for writing the software. The current languages theorized were python or java. Within the software, plans have been made to have a login to create yet another layer of security. Once the software has been finished, it would be satisfactory to have a network of meters seen on a single login. A future plan is to design the software in such a way that admins can easily see a map of meters data all presented in a single readable format.

# **Conclusion**

From this module we get data communication between a wireless computer and a Pic24 integrated circuit. This will enable us to communicate with the command board and create a tunnel of communication from the command board to a remote computer. This is helpful because it adds another layer of security and increases ease of access.

# **References**

<https://www.microchip.com/wwwproducts/en/PIC24F08KL302>

<https://www.espressif.com/sites/default/files/documentation/4a-esp8266_at_instruction_set_en.pdf>