WiFi & Client Mode

ECE 476 Advanced Embedded Systems

Jake Glower - Lecture #34

Please visit Bison Academy for corresponding
lecture notes, homework sets, and solutions

Introduction:

The Pico has two WiFi modes

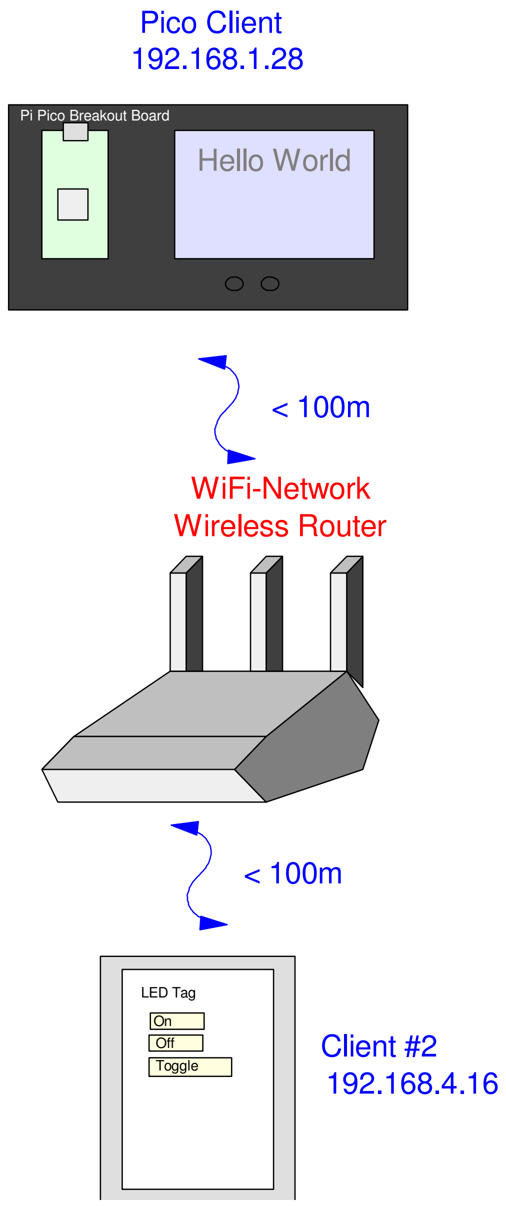
- AP: Pico is the WiFi server
  - Last lecture
- WLAN: Pico is a client
  - WiFi network must already exist
  - This lecture

When operating as a client

- The Pico has an address
  - 192.168.1.28
  - varies

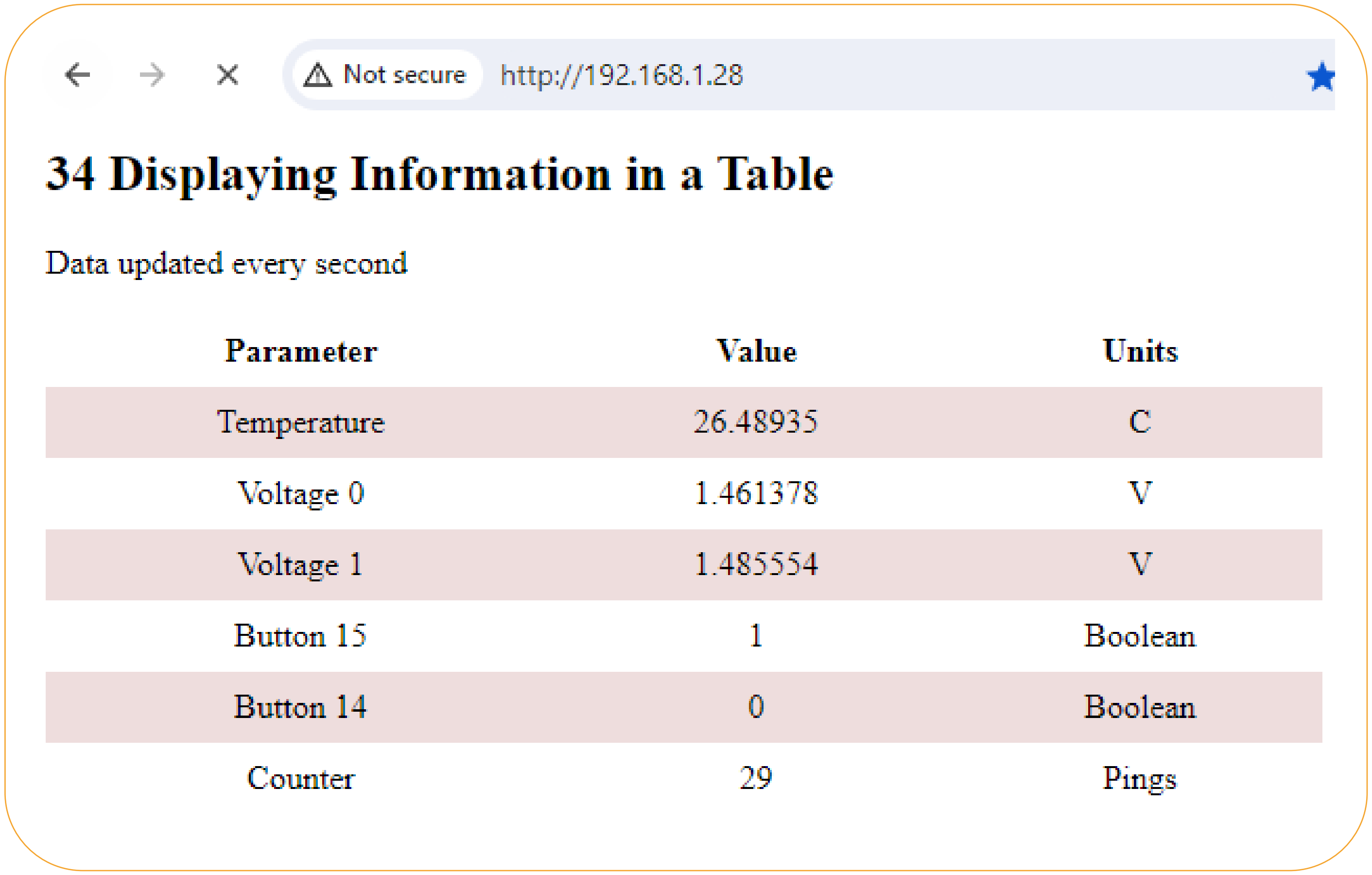
Anyone on this router can access the Pico

- Receive data from the Pico
  - This lecture
- Send data to the Pico
  - Next lecture



Goal of this lecture:

At the end of this lecture, a web page will be created allowing you to see the status of your Pico board as follows:



34 Displaying Information in a Table

Data updated every second

Parameter	Value	Units
Temperature	26.48935	C
Voltage 0	1.461378	V
Voltage 1	1.485554	V
Button 15	1	Boolean
Button 14	0	Boolean
Counter	29	Pings

Connecting to a Router

Step 1: Turn on the Pico's WiFi.

Option #1: Stand-Alone AP network

- Last lecture

```
wlan = network.AP_IF(network.STA_IF)
```

Option #2: Connect as a client to an existing WiFi network

- This lecture

```
wlan = network.WLAN(network.STA_IF)
```

Connecting to a WiFi Network (code)

```
import network, rp2, time

rp2.country('US')
wlan = network.WLAN(network.STA_IF))
wlan.active(True)

ssid = 'xxxx'
password = 'xxxx'

wlan.connect(ssid, password)

while( (wlan.isconnected() == 0) and (wlan.status() > 0)):
    print('Waiting to connect')
    time.sleep(1)

if(wlan.status() == 3):
    print('connection successful')
    print(wlan.ifconfig())
```

shell

```
Waiting to connect
Waiting to connect
Waiting to connect
Connection successful
('192.168.1.28', '255.255.255.0', '192.169.1.1', '192.168.1.1')
```

More Options

active()

```
wlan.active()                 return True if the network is active
wlan.active('up')             activate the network interface
wlan.active('down')           deactivate the network interface
```

connect(ssid, password)

```
wlan.connect(ssid, password) try to connect to a WiFi network
```

disconnect()

```
wlan.disconnect()             disconnect from the current network
```

scan()

```
wlan.scan()                   scan for networks
                              returns names of networks
```

More Options (cont'd)

status()

```
wlan.status()       returns the status of the network
                    -3  failed to connect due to password
                    -2  failed to connect - no such ssid
                    -1  failed to connect for other reasons
                    0   idle, no connection, no activity
                    1   trying to connect
                    3   connection successful
```

isconnected()

```
wlan.isconnected()  true if connected
                    false if not connected
```

ifconfig()

```
x = ifconfig()               returns four parameters
                    IP address
                    subnet mask
                    gateway server
                    DNS server
```

More Options (cont'd)

config()

```
wlan.config(channel=11)
wlan.config('ssid')
wlan.config('channel')
```

config(pm)

```
wlan.config(pm = 16)         disable power management
wlan.config(pm = 10555714)   maximum performance
wlan.config(pm = 17)         balanced performance vs. power
```

WiFi Example: UR Request

Once connected to the web, you can open and read web pages.

Example: read the contents of BisonAcademy.com

```
import network
import urequests

rp2.country('US')
wlan = network.WLAN(network.STA_IF)
wlan.active(True)

ssid = 'xxxx'
password = 'xxxx'
wlan.connect(ssid, password)

r = urequests.get("https://BisonAcademy.com")
print(r.content)
r.close()
```

UR Request (cont'd)

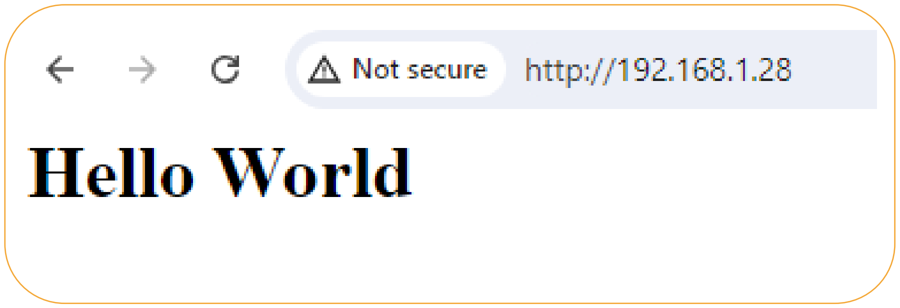
The html code of the web page is returned

- Not entirely sure how this is helpful

```
<!DOCTYPE html><html lang="en"><head><script
src="/gdpr/gdprscript.js?buildTime=1722004568&hasRemindMe=true&
stealth=false"></script><title>BISON ACADEMY -
Home</title><meta property="og:site_name" content="BISON
ACADEMY" />
<meta property="og:title" content="BISON ACADEMY" />
<meta property="og:description" content="Lecture notes,
homework sets, and solutions for courses taught in the
Department of Electrical and Computer Engineering at North
Dakota State University." /><meta property="og:image"
content="http://bisonacademy.com/uploads/3/4/4/0/34406028/glaci
er2_orig.jpg" />
<meta property="og:url" content="http://bisonacademy.com/" />
<link rel="icon" type="image/png"
href="//www.weebly.com/uploads/reseller/assets/1001-favicon.ico
" />
```

Creating a Web Page: Hello World

Starting with everyone's favorite example, let's create a web page that says *Hello World*



Hello World

html code:

- same we used before:

```
<html>
<body>
<h1>Hello World</h1>
</body>
</html>
```

Main Routine

import netman

- from Pepe80.com

Set up connection

- ssid
- password

Read in the web page

Open a socket

```
import netman
import socket
from machine import Pin

ssid = 'xxxxx'
password = 'xxxxx'
country = 'US'

wifi_connection =
netman.connectWiFi(ssid,password,country)

def web_page():
    f = open("HelloWorld.html","rt")
    x = f.read()
    x = x.replace('\r\n',' ')
    return(x)

# Open socket
addr = socket.getaddrinfo('0.0.0.0', 80)[0][-1]
wlan = socket.socket()
wlan.bind(addr)
wlan.listen(1)

print('listening on', addr)
```

Main Routine (cont'd)

The main routine then simply

- Waits for a ping in *wlan.accept()*
- Then echos back the html code with *cl.send(response)*
- Then closes the current ping

The shell window

- shows the connection
- shows who pinged the Pico

```
while(1):
    cl, addr = wlan.accept()
    print('client connected from', addr)
    request = cl.recv(1024)

    response = web_page()

    cl.send(response)
    cl.close()
```

Shell

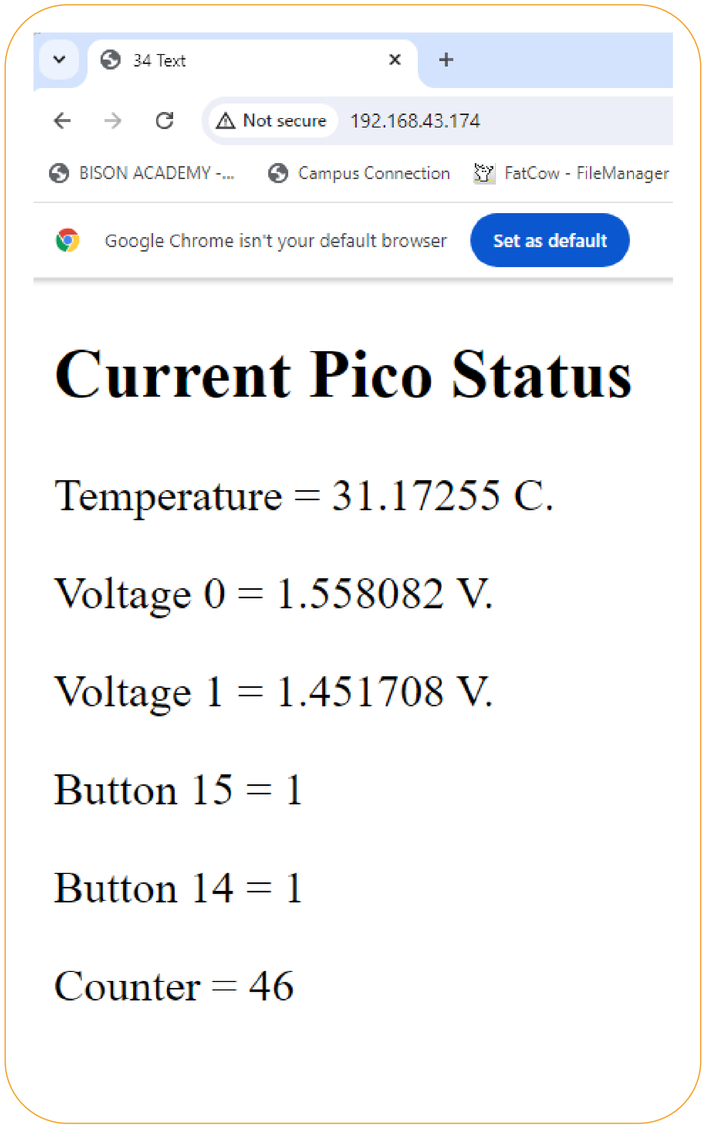
```
MPY: soft reboot
waiting for connection...
waiting for connection...
waiting for connection...
connected
ip = 192.168.1.28
listening on ('0.0.0.0', 80)
client connected from ('192.168.1.3', 52527)
client connected from ('192.168.1.3', 52528)
client connected from ('192.168.1.3', 52529)
:
:
```

Displaying Information in Text Format

Each ping, the Pico updates the web page

- Automatic: Once per second
- F5: Refresh

By modifying information in the web page, you can see what's happening in the Pico's world



Current Pico Status

Temperature = 31.17255 C.

Voltage 0 = 1.558082 V.

Voltage 1 = 1.451708 V.

Button 15 = 1

Button 14 = 1

Counter = 46

html Code

Set up a page

refresh

- page automatically refreshes every second

Use dummy variables

- aaaaa
- bbbbb
- These will be replaced with current data

```
<!DOCTYPE html>
<html>

<head>
    <title>34 Text</title>
    <meta http-equiv="refresh" content="1">
</head>

<body>
    <h2>Current Pico Status</h2>
    <p>Temperature = aaaaa C.</p>
    <p>Voltage 0   = bbbbb V.</p>
    <p>Voltage 1   = ccccc V.</p>
    <p>Button 15   = ddddd </p>
    <p>Button 14   = eeeee </p>
    <p>Counter     = fffff </p>
</body>

</html>
```

note: This is one solutions

- Other (better) ways to do this probably exist

web_page()

Similar to before

- Pass data to be displayed
- Replace dummy variables.

```
def web_page(Temp, V0, V1, B15, B14, N):
    f = open("34 Text.html","rt")
    x = f.read()
    x = x.replace('\r\n',' ')
    x = x.replace('aaaaa', str(Temp))
    x = x.replace('bbbbb', str(V0))
    x = x.replace('ccccc', str(V1))
    x = x.replace('ddddd', str(B15))
    x = x.replace('eeeee', str(B14))
    x = x.replace('fffff', str(N))
    return(x)
```

Main Loop

The main loop then

- Waits for a ping (which will happen every second),
- Collects data, then
- Replies with the updated web page

```
a2d0 = ADC(0)
a2d1 = ADC(1)
a2d4 = ADC(4)
k = 3.3 / 65520
B15 = Pin(15, Pin.IN)
B14 = Pin(14, Pin.IN)

N = 0
while(1):
    cl, addr = wlan.accept()
    print('client connected from', addr)
    request = cl.recv(1024)

    N += 1
    V0 = a2d0.read_u16()*k
    V1 = a2d1.read_u16()*k
    a4 = a2d4.read_u16()
    Temp = 0.02927*(14940 - a4)

    response = web_page(Temp, V0, V1,
B15.value(), B14.value(), N)

    cl.send(response)
    cl.close()
```

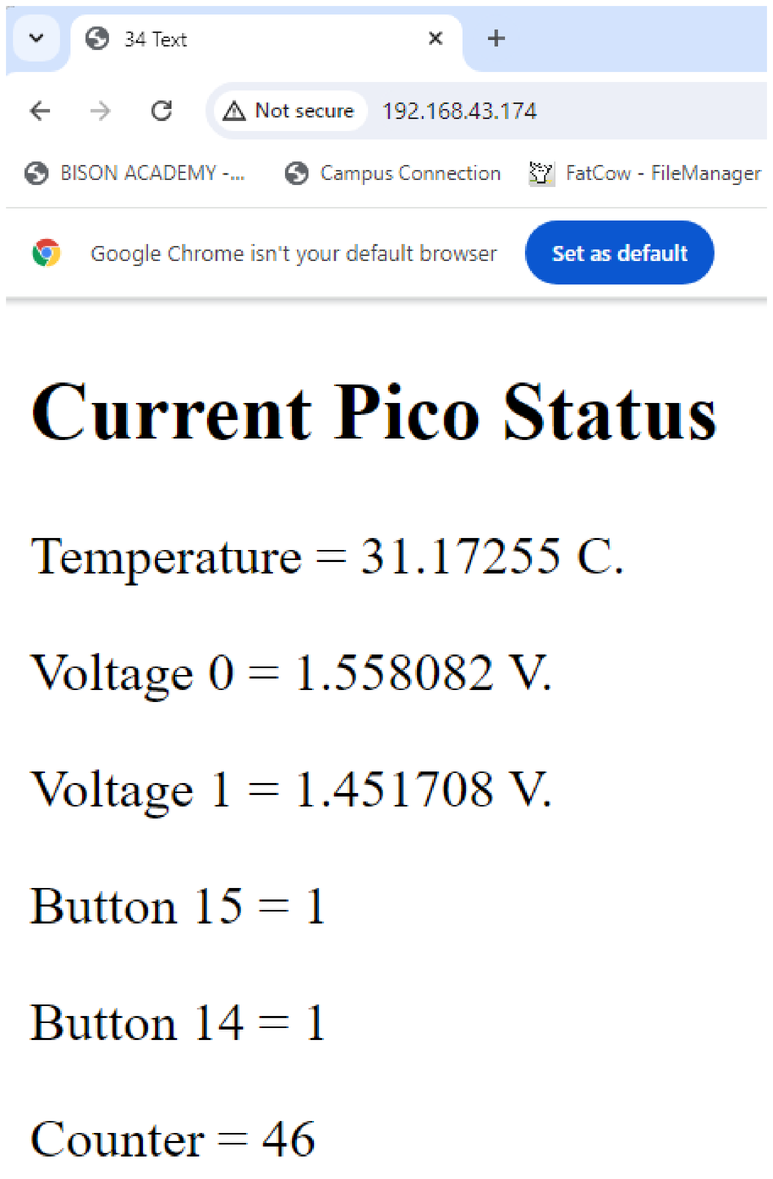
Result

Connect to 192.168.43.174

- Pi-Pico

Status of Pico can be seen

Data is updated every second



Current Pico Status

Temperature = 31.17255 C.

Voltage 0 = 1.558082 V.

Voltage 1 = 1.451708 V.

Button 15 = 1

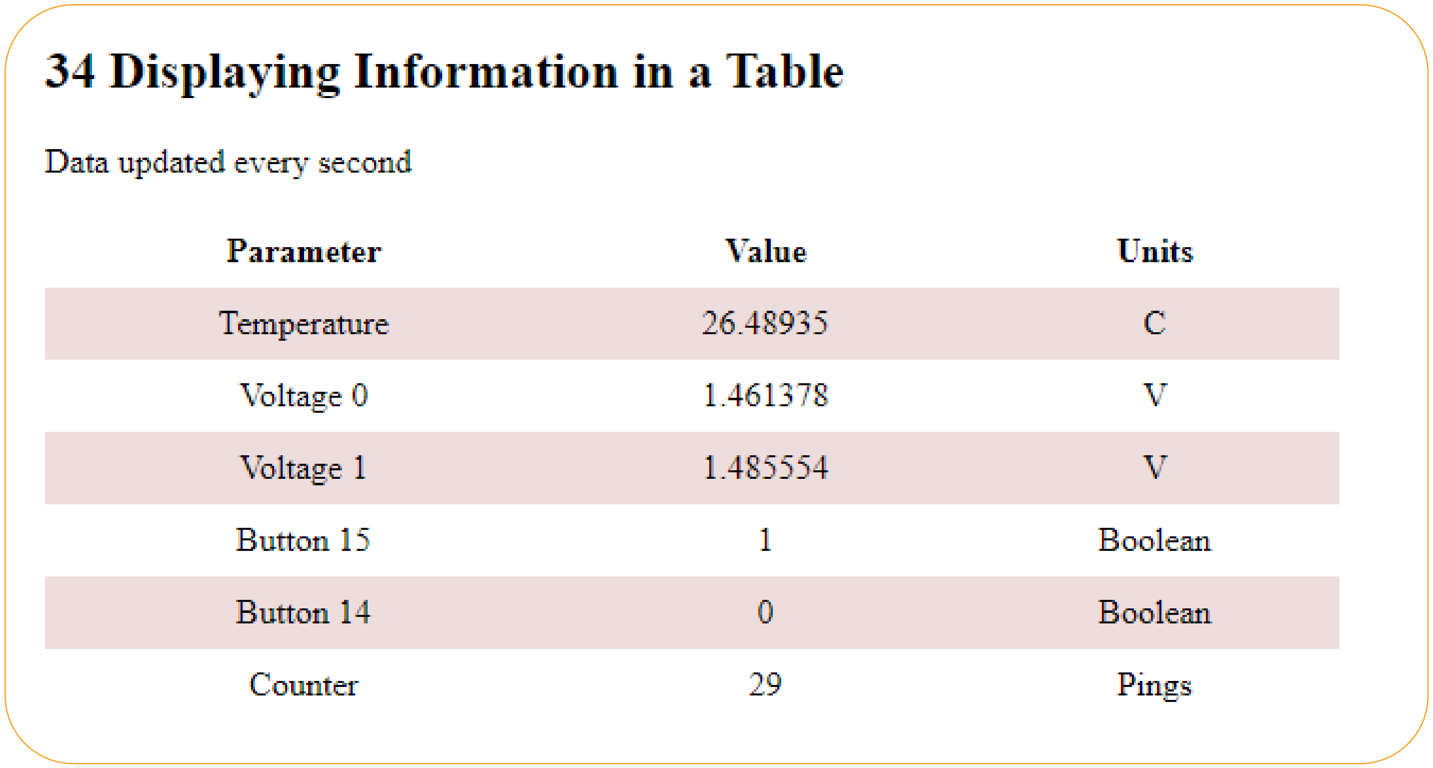
Button 14 = 1

Counter = 46

Displaying Data in Table Format

Same information

- Prettier display
- Update every second



34 Displaying Information in a Table

Data updated every second

Parameter	Value	Units
Temperature	26.48935	C
Voltage 0	1.461378	V
Voltage 1	1.485554	V
Button 15	1	Boolean
Button 14	0	Boolean
Counter	29	Pings

Goal: Display data in a table which is updated every second

html code: Table display

- Use dummy variables for the data
- Everything else remains unchanged

```
<!DOCTYPE html><html>
<head>
  <title>34 Table</title>
  <meta http-equiv="refresh" content="1">
  <style>
    table {  border-collapse: collapse;   width: 80%; }
    th, td {  text-align: center;  padding: 8px; }
    tr:nth-child(even) { background-color: #EEDDDD; }
    </style>
  </head>
<body>
  <h2>34 Displaying Information in a Table</h2>
  <p>Data updated every second</p>
  <table>
    <tr> <th>Parameter</th>   <th>Value</th>   <th>Units</th> </tr>
    <tr> <td>Temperature</td> <td> aaaaa </td> <td>C</td> </tr>
    <tr> <td>Voltage 0</td>   <td> bbbbb </td> <td>V</td> </tr>
    <tr> <td>Voltage 1</td>   <td> ccccc </td> <td>V</td> </tr>
    <tr> <td>Button 15</td>   <td> ddddd </td> <td>Boolean</td> </tr>
    <tr> <td>Button 14</td>   <td> eeeee </td> <td>Boolean</td> </tr>
    <tr> <td>Counter</td>   <td> fffff </td>   <td>Pings</td>   </tr>
    </table>
  </body>
</html>
```

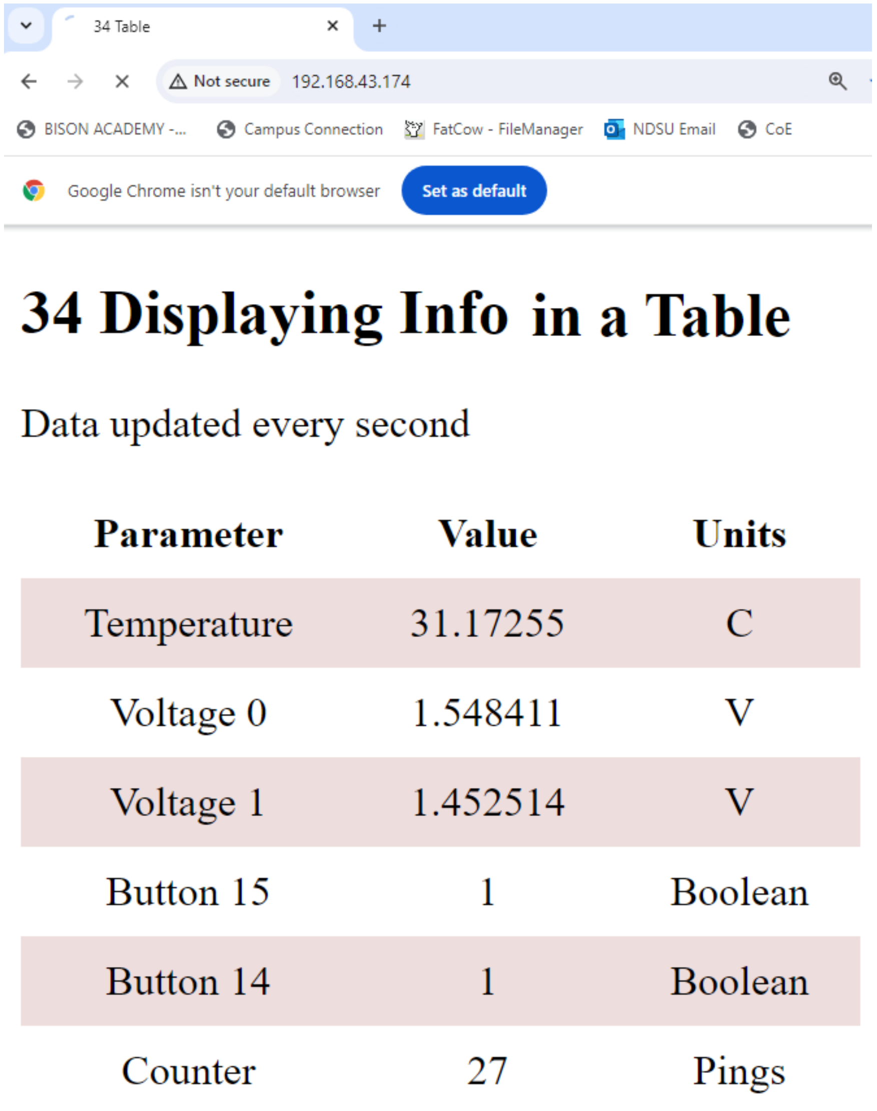
Result

Connect to 192.168.43.174

- Pico

Status of Pico can be seen

- Data is updated every second



34 Displaying Info in a Table

Data updated every second

Parameter	Value	Units
Temperature	31.17255	C
Voltage 0	1.548411	V
Voltage 1	1.452514	V
Button 15	1	Boolean
Button 14	1	Boolean
Counter	27	Pings

Summary

If a WiFi network already exists

- The Pico can be connected as a client

Once connected, other clients on the WiFi network can see what's happening to the Pico

- Connect to the Pico's URL address
- Data is updated every ping

References

- pepe80.com
- https://www.w3schools/tags/att_input_type