

El-Metkoul Karim

Progetto TPS

Sistema per Ridurre l'utilizzo dello smartphone e altre apparecchiature elettroniche

Obiettivo:

Realizzare un sistema embedded che permetta il monitoraggio del tempo di utilizzo dello smartphone attraverso un server HTTP e una striscia al Led RGB.

Il tempo di utilizzo viene monitorato da un Sensore a ultrasuoni **HC-SR04**.

La pagina web riporta un grafico con il tempo di utilizzo a seconda della fascia oraria riportando un log con l'ora e il tempo di utilizzo stimato.

La pagina web che viene visualizzata da Arduino è dotata di css per la realizzazione del grafico. Per questioni di comodità la pagina HTML è stata direttamente caricata sulla scheda ESP8266 mediante l'utilizzo di un addon che riesce a dedicare zone di memoria per l'inserimento di immagini o altri script.

Il sistema rileva l'orario e la fascia di tempo, nel caso il tempo di utilizzo del dispositivo è eccessivo compare un warn sulla pagina Con il messaggio che dice "hai superato il tempo di utilizzo consigliato" invece quando il dispositivo viene prelevato nel periodo adibito alla scuola Arduino invia un pacchetto Json al philips hue Bridge collegato in locale che va a cambiare colore della scatola in rosso, invece se il dispositivo è posto nella base apposita la scatola torna al colore originario quindi verde.

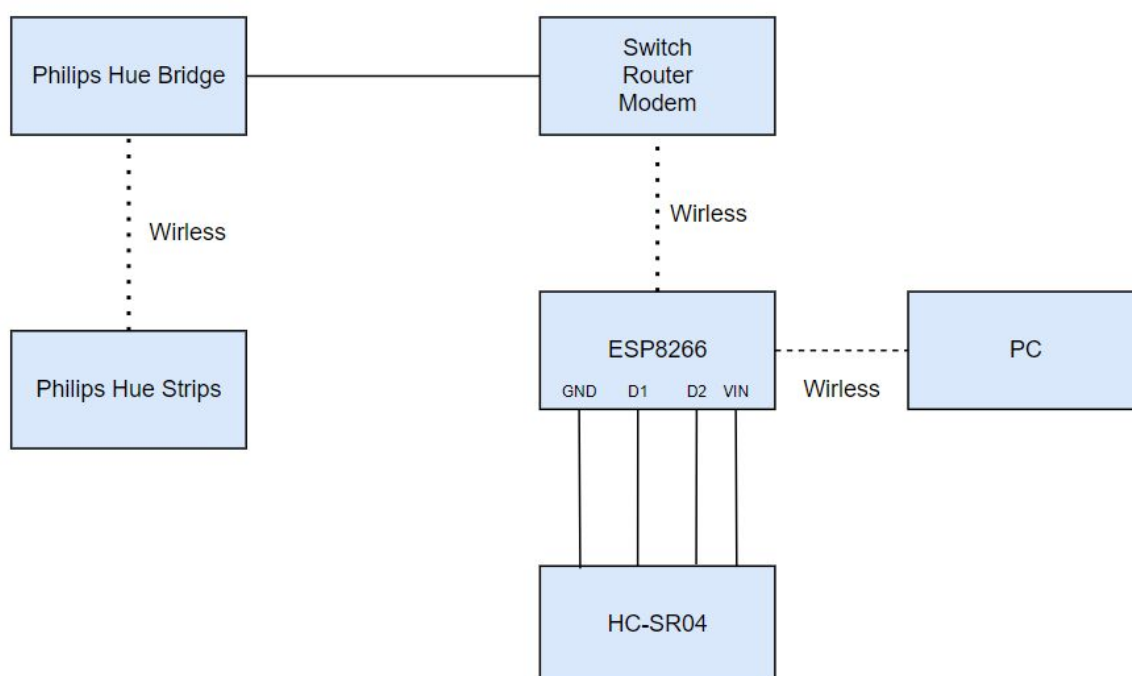
Lista dei Componenti

- 1x Esp8266 ("NodeMcu Lua")
- 1x Philips Hue Bridge
- 1x Philips Hue Strips
- 1x Sensore **HC-SR04**
- 4x Jumper
- 1x Cavo Ethernet
- 2x Breadboard
- 1x Cavo MicroUSB
- 2x Alimentatori

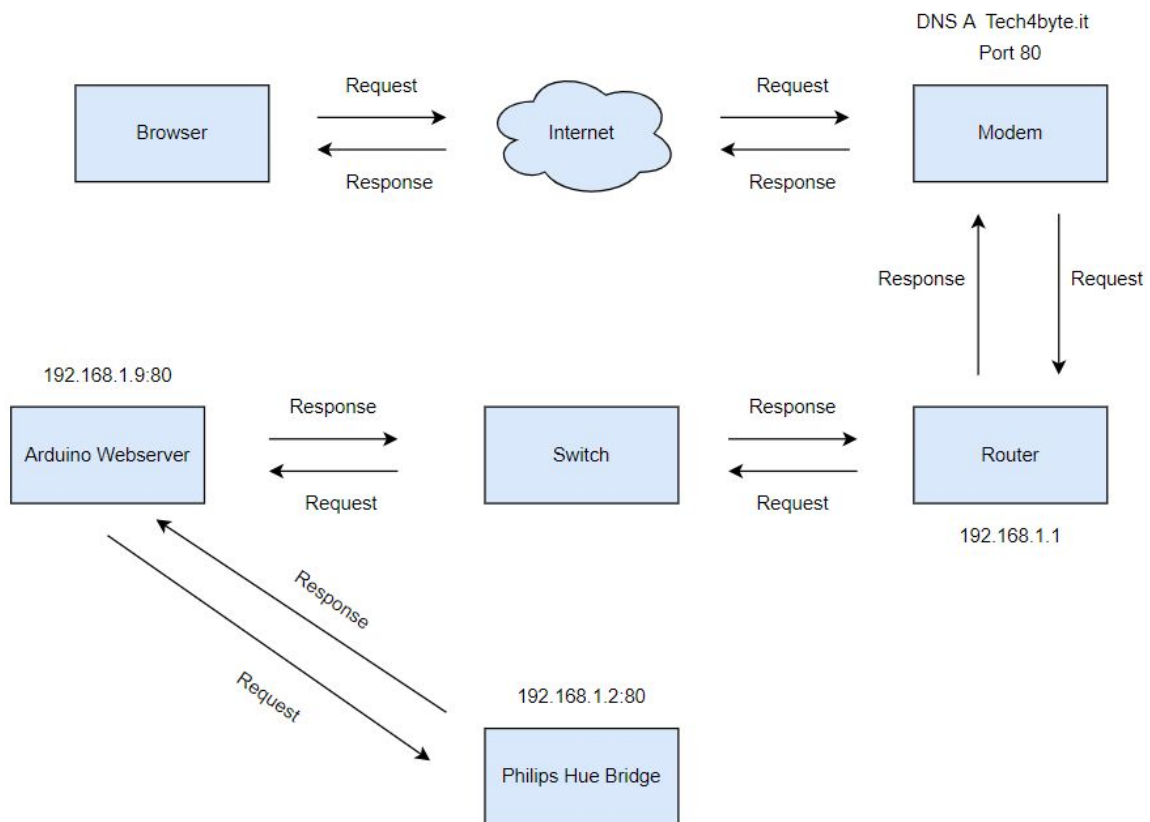
Software utilizzati

- Arduino IDE 1.8.12
- Atom
- Google Chrome 88.0.4324.104
- NodeMcu Flasher
- Fing

Schema fisico del circuito



Schema logico di rete



Flowchart

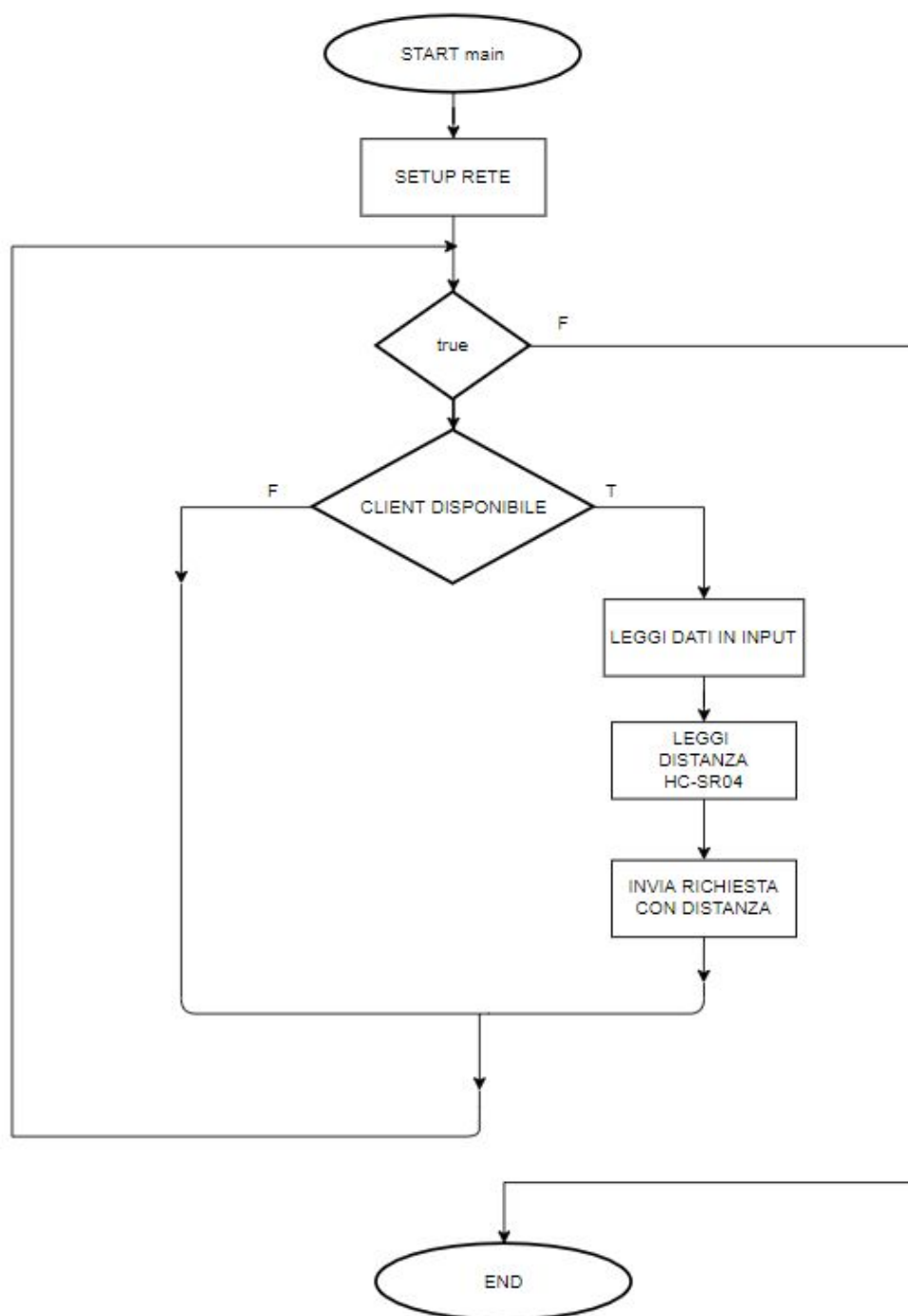
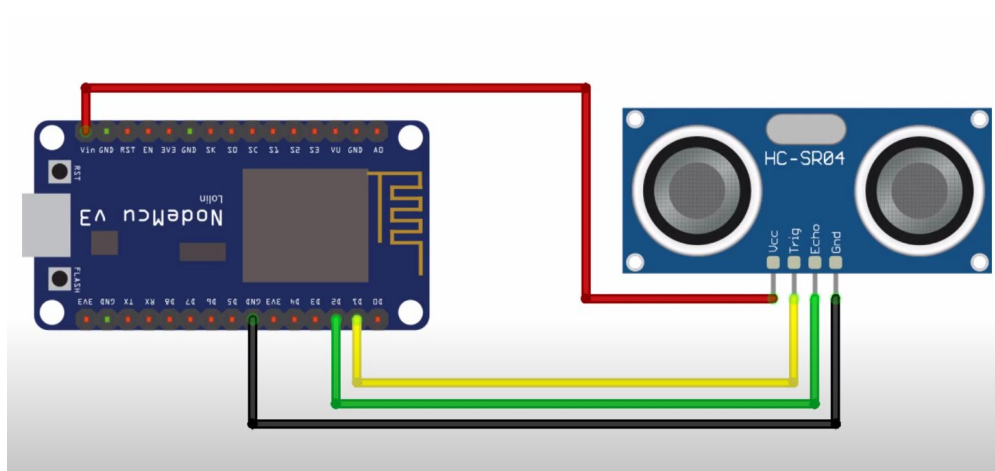
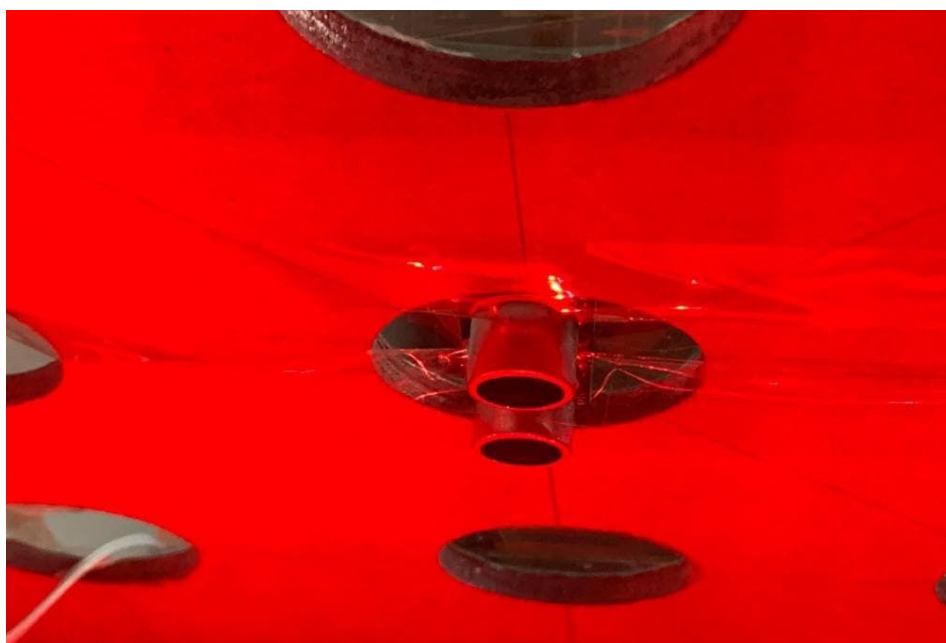
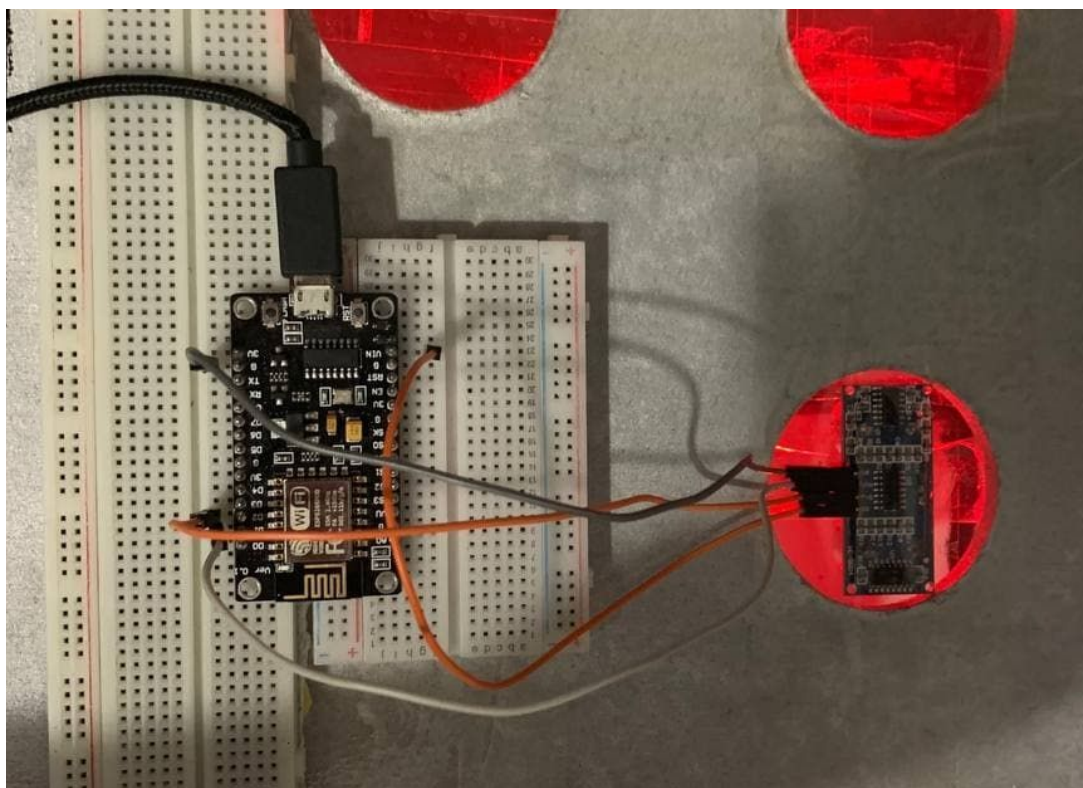


Foto del circuito:



Codice sorgente

```

1  // Carica Libreria Wi-Fi
2  #include <ESP8266WiFi.h>
3
4  // Credenziali accesso rete
5  const char* ssid      = "Tech4Byte";
6  const char* password = "*****";
7
8  // imposto porta di comunicazione
9  WiFiServer server(80);
10

```

```

14
15 // Assegno output ai Pin GPIO
16 const int output5 = 5;
17 const int output4 = 4;
18
19

```

Configurazione dei parametri di rete.

```

21
22 const char* endpoint = "http://192.168.0.3/api/MhK4WdkWiu5aqY3JtLGyC1Sai78vuNh3TqyZQvSS/lights/1/state"; //Chiave connessione Philips Hue Bridge
23
24 void setup(void)
25 {
26   Serial.begin(9600);
27   WiFi.begin(ssid, password);
28   while (WiFi.status() != WL_CONNECTED)      // Conesso alla rete
29   {
30     delay(500);
31     Serial.print(".");
32   }
33   Serial.println("");
34   Serial.println("Connected");
35 }
36

```

Definisco Variabile per assegnare chiave di accesso al bridge e verifico se la connessione è stata stabilita

```
28 void loop()
29
30
31 {
32     TrunOffLight();
33     delay(2000);
34     TrunOnLight();
35     delay(2000);
36     {
37
38
39         if ( distance > 36){
40
41             //imposto colore rosso
42             String message = "{\"on\":true, \"sat\":254, \"bri\":150,\"hue\":";
43             delay(2000);
44         }
45
46         if ( distance < 34 ){
47
48             //imposto colore verde
49             String message = "{\"on\":true, \"sat\":254, \"bri\":150,\"hue\":20000";
50             delay(2000);
51
52         }
```

Spengo e accendo la striscia per controllare il corretto funzionamento, e imposto il colore a seconda della distanza rispetto alla base.

API Debug tool

Generated Hex Color Codes at in

Get Start

← → ↻

Non sicuro | 192.168.1.2/debug/clip.html?com...

🔍

CLIP API Debugger

URL:

GET

PUT

POST

DELETE

Message Body:

```
{ "on": true, "sat": 254, "bri": 254, "hue": 20000 }
```

Command Response:

```
[
  {
    "success": {
      "/lights/1/state/on": true
    }
  },
  {
    "success": {
      "/lights/1/state/hue": 20000
    }
  },
  {
    "success": {
      "/lights/1/state/sat": 254
    }
  },
  {
    "success": {
      "/lights/1/state/bri": 254
    }
  }
]
```

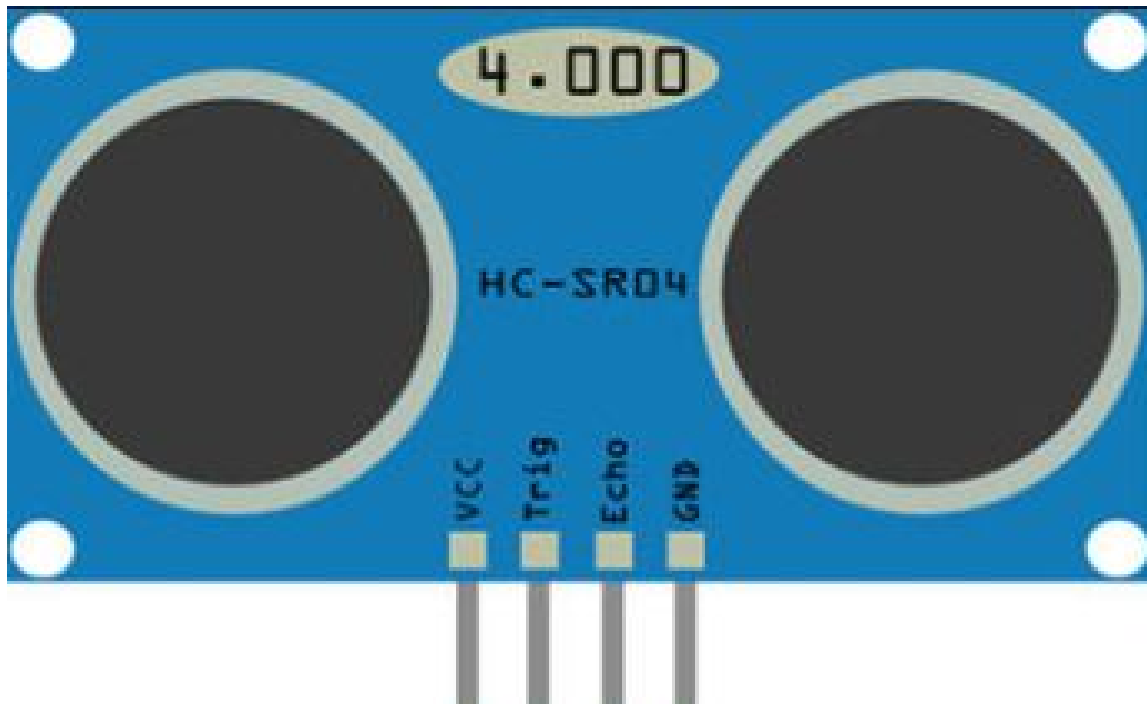
Il bridge presenta un debugger per veicolare la striscia al Led RGB, il linguaggio JSON è stato utilizzato nella IDE per il controllo dei colori. per ogni dispositivo/software è associato un codice di verifica univoco.

Il codice sorgente, completo con tutti i file necessari per il corretto funzionamento del progetto è disponibile nelle note della relazione alla fine del documento.

Problemi riscontrati e test

Sensore HC-SR04

Il Sensore HC-SR04, prodotto dalla SparkFun è dotato di 4 pin che vengono usati per l'alimentazione e per la lettura della distanza.



Se l'ordine dei Pin non è corretto si presentano diversi problemi, del tipo se si inverte il Pin VCC e GND il sensore inizia a produrre calore invece se vengono invertiti il pin TRING e ECHO Arduino non riesce a interpretarli.

Calcolare La Distanza

```
long durata = pulseIn( echoPort, HIGH );
long distanza = 0.034 * durata / 2;           //calcolo per distanza
Serial.print("distanza: ")
if ( durata > 38000 ) {                       //controlla se la distanza supera i 3800ms
  Serial.println("Fuori portata ");           //se maggiore è fuori portata
}
else {
  Serial.print(distanza);                     //esprimi distanza
  Serial.println(" cm ");                     //in cm
}
```

formula della spazio: $s = v \cdot t$

la velocità del suono pari a 343 m/s (che espressa in microsecondi diventa 0,0343 m/uS), si ottiene:

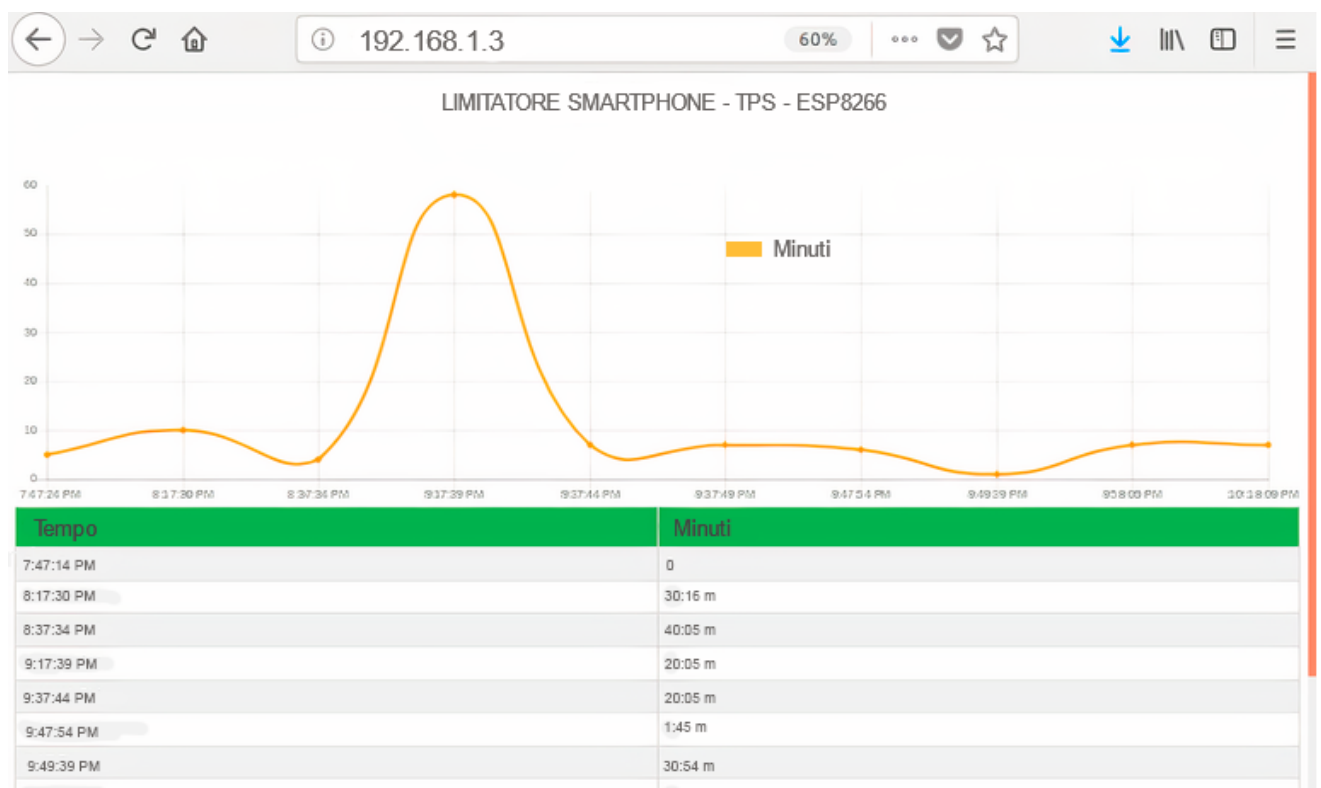
$$s = 0,0343 \cdot t$$

Il tempo ottenuto deve essere diviso in 2 perchè il tempo è diviso in trasmissione e ricezione

$$s = 0,0343 \cdot (t/2)$$

Programmazione in HTML e CSS della pagina web

Grafico tempo di utilizzo



La pagina HTML registra i log quando nella scatola viene riposto o rimosso il dispositivo dalla superficie di testing, Il grafico rappresenta il tempo di utilizzi in minuti e in basso viene riportato una tabella con i log di rimozione e inserimento con i minuti.

Port forwarding sul router

Il forwarding di una porta è una procedura che varia da modem a modem. Ad Esempio nel mio caso il mio Firmware è: XS_3.7.04.06 Vodafone Power Station Per effettuare il port forwarding bisogna selezionare l'ip del dispositivo e la porta o le porte che si desidera aprire.

Aggiungi triggering porte

Servizio

TCP
▼

Dispositivo

Nessun dispositivo specifico
▼

Indirizzo IP locale

192

168

1

3

Tipo porte

☒
Porta

☐
Intervallo porte

Porta pubblica

80

Porta locale



80

Salva

Annulla

Configurazione DNS

L'indirizzo che abbiamo assegnato dall'ISP non è statico, dunque l'ip può cambiare molto spesso e per questo motivo è comodo usare un DNS, così non è necessario impararsi dei numeri a memoria ma occorre solo impararsi l'indirizzo del proprio DNS, ad esempio con CloudFlare si può impostare un record di tipo A che varia ogni volta che varia il nostro ip domestico.

Type	Name	Content	TTL	Proxy status	
A	tech4byte.it	37.179.33.86	Auto	 Proxied	Edit ▶
A	www	37.179.2.136	Auto	 Proxied	Edit ▶

Il sito è raggiungibile ovunque in qualsiasi momenti al seguenti indirizzo:

Sito: <https://tech4byte.it>

File Progetto: <https://github.com/Ferrar65/Phone-Limitator-ESP8266>

Philips Hue JSON Tutorial: <https://developers.meethue.com/develop/get-started-2/>