Micro Controladores
Prática 3 - Comunicação Serial

Átila Camurça Alves

[1]Instituto Federal do Ceará (IFCE)

Abstract. *This practice refers to the creation of a circuit that will achieve to make serial communication. In this case a PIC18F4550 on the SanUSB platform communicating with an ESP8266.*

Resumo. *Esta prática refere-se a criar um circuito que irá realizar comunicação serial. No caso o PIC18F4550 na plataforma SanUSB comunicando-se com um ESP8266.*

1. Introdução

Para a prática 3 será necessário montar um circuito que irá realizar comunicação serial com equipamento externo, no caso o PIC18F4550 na plataforma SanUSB comunicando-se com um ESP8266.

2. Objetivo

Enviar um valor inteiro para o ESP8266 através de uma URL para que o ESP8266 identifique o valor e envie através de porta serial o valor para o PIC18F4550 fazendo com que ele interrompa e inicie o processo de alternar LEDs para um sinal de pedestres e automóveis.

3. Comunicação

Ao receber uma requisição HTTP, com uma URL do tipo `http://172.24.2.95/?val=10`, o ESP8266 obtém o seguinte *payload*:

```
GET /?val=10 HTTP/1.1
Host: 172.24.2.95
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
```

Logo na primeira linha está o valor que desejamos enviar ao PIC.

Abaixo um trecho do codigo fonte (Linguagem Lua):

```
-- a simple http server
srv = net.createServer(net.TCP)
srv:listen(80, function(conn)
    conn:on("receive", function(sck, payload)
        if string.match(payload, "val=") then
            tempo = string.match(payload, '%d+')
            print(tempo)
        end

        sck:send("HTTP/1.0 200 OK\r\n"
            .. "Content-Type: text/html\r\n\r\n<h1>"
            .. "Iniciando semáforo com tempo de "
            .. tempo
            .. " segundos<br>Transmitido via ESP8266</h1>")
    end)
end)
```

A ideia é que quando o payload conter a String `val=` (linha 5), então o tempo é obtido através do *regex* (Regular Expression) `%d+`, indicando que é um valor inteiro, um dígito ou mais (linha 6). Finalmente o ESP envia o valor para a porta serial (linha 7) através de um `print`.

4. Circuito

As Figuras 1, 2 e 3 mostram a evolução do circuito.

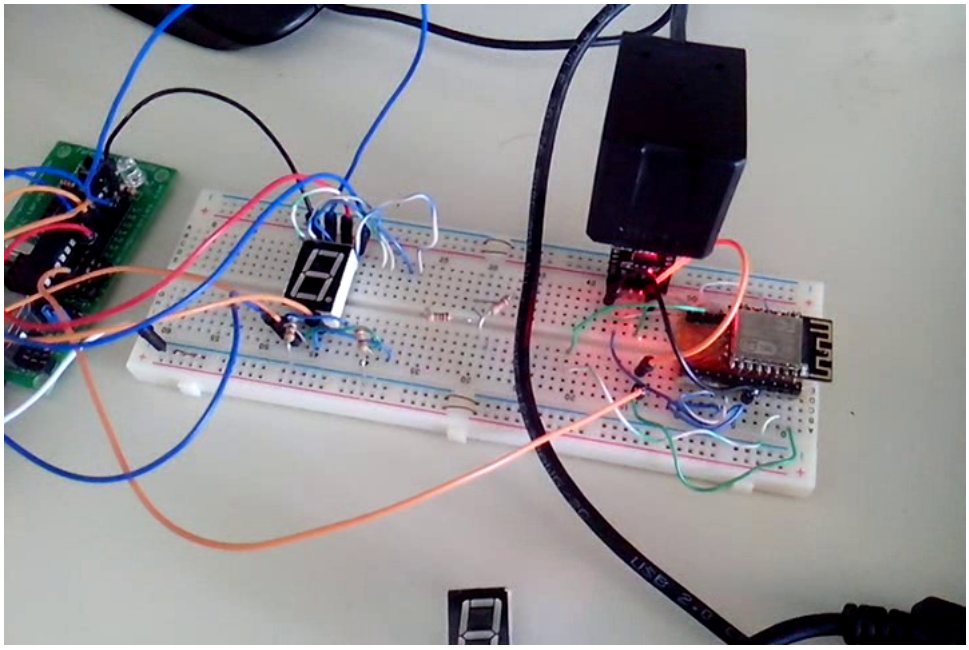

Figura 1. Montagem Inicial apenas com 1 display de 7 segmentos

5. Recepção serial

Para receber o valor pela serial foi usado a opção de verificar se houve interrupção e ler o valor da porta serial. Abaixo um trecho do código:

```
char char_byte;
char comando[2];
int pos = 0;
int valor_recebido = 0;
void interrupt interrupcao() {
    if (serial_interrompeu) {
        serial_interrompeu = 0;
        char_byte = ReadUSART();
        // recebe bytes até '\n'
        if (char_byte != '\n') {
            comando[pos] = char_byte;
            pos++;
        } else {
            valor_recebido = 1;
            pos = 0;
        }
    }
}
```

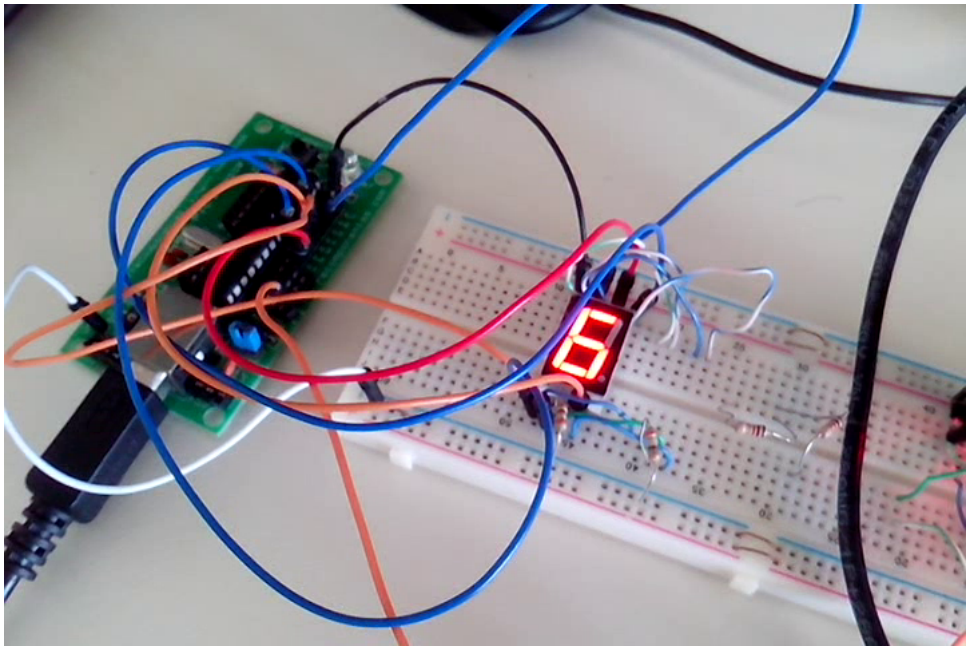

Figura 2. Envio de comando apenas para ligar, ainda sem indicar valor

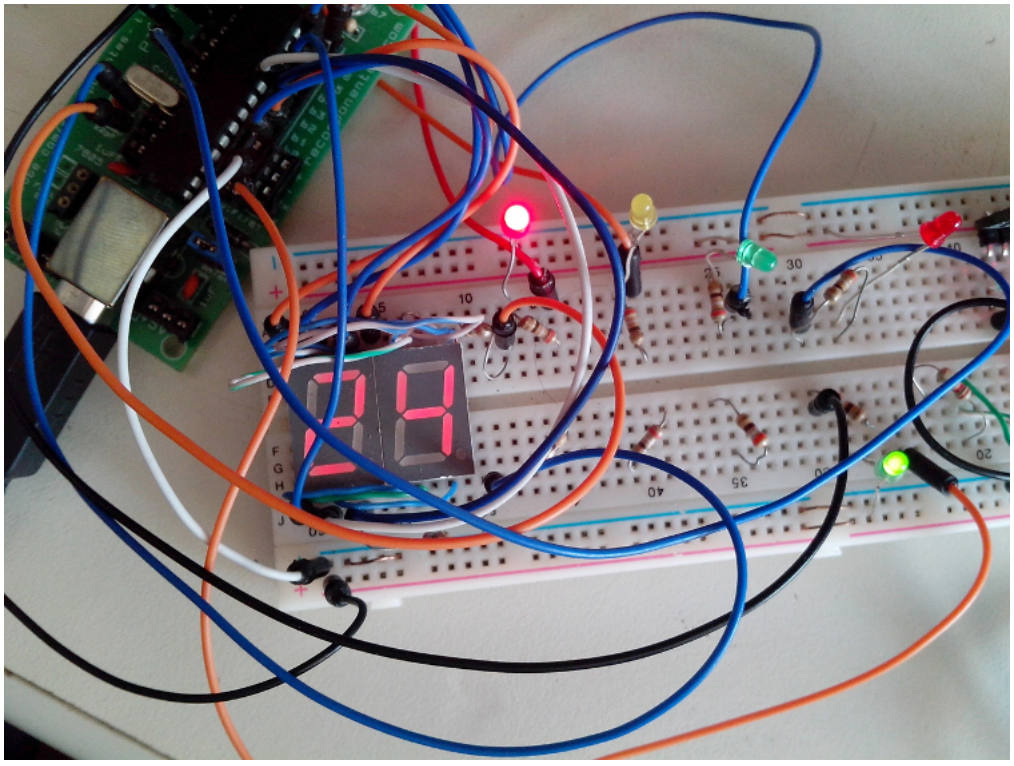

Figura 3. Circuito Final, com semáforo e valor passado via porta serial com 2 displays de 7 segmentos