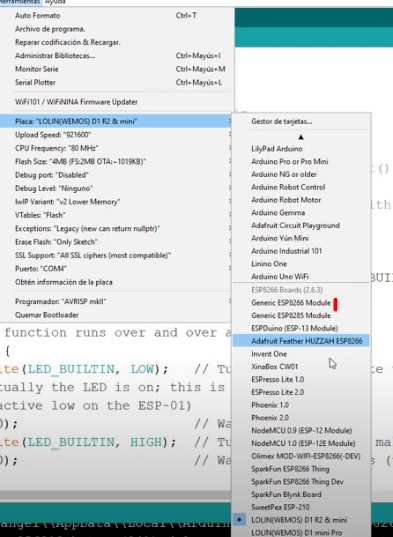
TRABAJO FINAL PROGRAMACION DISTRIBUIDA

Para instalar el WEMOS D1 segui los pasos del video <https://www.youtube.com/watch?v=9mkPMXblPJI>, primero instalamos el driver para que la pc detecte la placa, luego en el arduino agregamos el paquete de ESP8266 e instalamos el mismo gestor de tarjeta.

En mi caso uso el Generic ESP8266 Module en vez del WEMOS D1.



Luego se probó el ejemplo que hay en el video para ver si funcionaba la placa correctamente.

Ahora ya llegamos a la parte de los experimentos de la practica, los cuales trabajaban con un servidor y cliente enviandose mensajes y calculando el tiempo que tardaba en enviarse y llegar.

Para hacer lo mismo con la placa y calcular lo que sería “tiempo real” de comunicación primero seguí los pasos del siguiente video

<https://www.youtube.com/watch?v=_Kgm1jGAFt8>

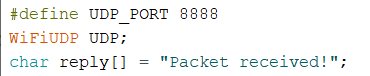
(ya no, ahora use esta pagina <https://siytek.com/esp8266-udp-send-receive/> y esta <https://www.luisllamas.es/esp8266-protocolo-udp/>)

En el mismo se aprendio a crear un cliente servidor mediante una comunicación por UDP/IP

Primero incluimos los paquetes que necesitaremos

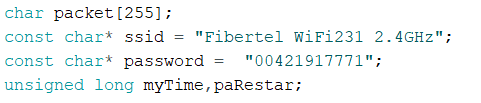


Definimos el puerto UDP, instancia de la clase UDP y la respuesta para cuando recibamos el mensaje



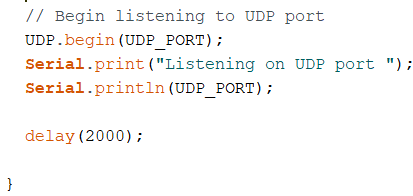
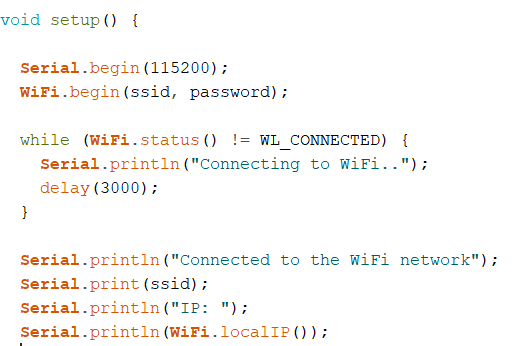
Por un lado tendremos las constantes que se necesitan para la conexión wifi, básicamente el wifi que se usa y la contraseña.

Por otro el buffer que es para el mensaje que enviaremos y los long que se usarán para calcular el tiempo que tarda desde el envío hasta que llega.

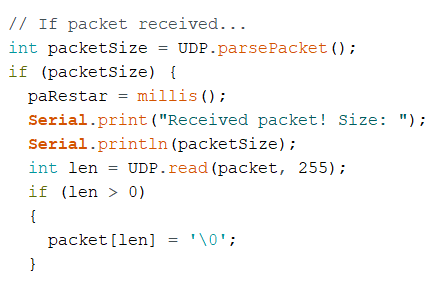


El Serial.begin(115200) es el upload speed que tenemos para poder ver el monitor serial.

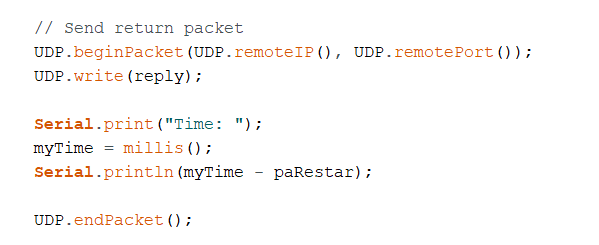
Se inician las conexiones con wifi y UDP se conecta al puerto.



En el void loop se hace lo siguiente, si se recibe el paquete UDP se comienza el conteo del tiempo de comunicación, imprimimos el tamaño del paquete, se lee.



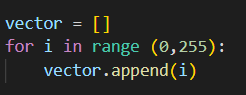
Ahora enviamos un paquete avisando que le leyo y finalizado esto podemos calcular el tiempo total de comunicación.



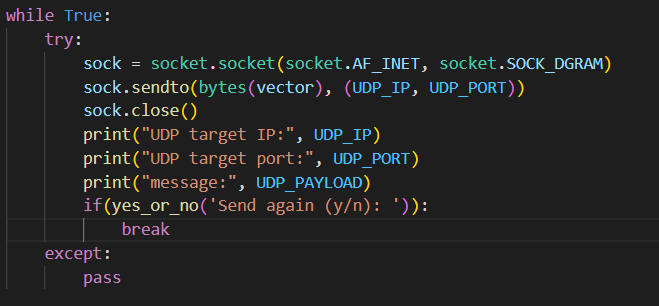
Ahora para poder enviar el mensaje y recibir la respuesta se creo un codigo en python que se conectara al mismo puerto y a la IP que se indica cuando ejecutamos el arduino.



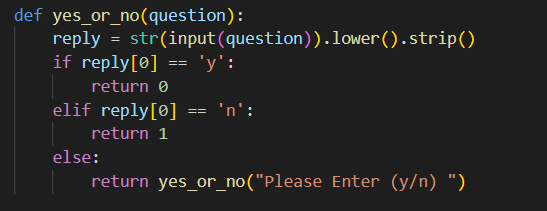
El mensaje sera un vector de 255



Después se utilizó un while true donde se envía paquetes con los bytes del vector creado y se creo una funcion yes or no que permite volver a enviar mensajes



En la funcion se chequea que si ponemos “y” se vuelva a enviar y si ponemos “n” termine



Explicar las diferencias y similitudes entre ambos contextos de experimentos (PC y PC+ESP8266-Wemos). Específicamente lo referido a los tamaños de mensajes y reintentos necesarios y los tiempos de comunicaciones de la práctica 1 (podés directamente copiarlos de lo que entregaste en su momento). En particular, podés comentar también la relación entre los tiempos de comunicaciones con la capacidad de transporte de datos de la capa física con ESP8266-Wemos (que es WiFi).

Tamaños de mensajes y reintentos necesarios diferencias:

Los tamaños de mensajes que utilice tanto acá como en el ejercicio de la practica deberían ser los mismos con el vector de 255 en ambos.

Relacion entre los tiempos de comunicaciones con la capacidad de transporte de la capa fisica con Wemos

En cuanto a los tiempos de comunicación podemos observar que los de la practica rondaban entre 10 y 17 o un poco más y menos en el caso de C, pero con la placa tan solo tarda entre 0 y 1 milisegundo

