PoloPoints

Versión	Fecha	Autor	Descripción de cambios
1.00	23/04/2023	Pedro Parodi	Versión inicial
1.01	6/5/2023	Pedro Parodi	Agregado descanso entre chukkers
1.10	24/1/2024	Pedro Parodi	Actualización según APP y manual de armado

Índice

1 Modo de uso del tablero

La operación comienza con el encendido del cartel. Luego, el dispositivo ESP32 inicia un access point para conectarse a través de una interfaz WiFi y automáticamente se configuran los números del cartel a sus valores de inicio. Por lo general, estos valores son: puntajes en 0, chukker en 1, períodos de 7’ 00’’ (con un agregado de 30’’ de tiempo extendido) e intervalo de descanso de 3’.

El usuario que administre el cartel deberá loguearse al access point con los siguientes datos:

- **SSID:** PoloPoints
- **PASSWORD:** 12345678

Y luego acceder al sitio web que le permite administrar el cartel:

- **DOMINIO:** http://polopoints.local (accesible a través de un navegador web)

La información del cartel físico y lo que muestre la aplicación deben ser idénticas en todo momento.

Ilustración 1. Imagen representativa del tablero y aplicación.

1.1 Dinámica del juego

El modo de juego es el siguiente: el usuario da comienzo al partido iniciando el temporizador desde la aplicación móvil. Durante el transcurso del *chukker*, iniciar/detener el temporizador o incrementar/decrementar puntajes estará a cargo del usuario.
Finalizado los 7’ del *chukker*, sonará una campana y el juego continua con 30’’ adicionales de tiempo extendido (total 7’ 30’’). Transcurrido este tiempo, sonará nuevamente la campana y este evento da por finalizado el *chukker* actual y se pasa a un intervalo de descanso de 3’.
Luego del intervalo, el número de *chukker* se incrementa automáticamente y habilitará que el juego continue y repetir asi el ciclo del juego.
El juego finaliza cuando el usuario lo desee, no es necesario prefijar una cantidad de *chukkers* desde la aplicación.

1.2 Consideraciones para el uso

Si bien la aplicación administra el tablero con poca intervención del usuario, es posible en todo momento iniciar/detener/modificar el reloj, modificar puntajes o el número del *chukker*.

El botón **Reiniciar** reinicia el *chukker* actual, es decir, lleva el reloj al valor 07’ 00’’. Independientemente si el juego se encuentra en intervalo de descanso o tiempo extendido. Las modificaciones realizadas a los puntajes o al número de *chukker* no se ven afectadas, conservando las operaciones realizadas por el usuario.

El botón **Reiniciar tablero** lleva el tablero a sus valores de inicio. Salvo modificaciones realizadas por el usuario, esto corresponde a puntajes en cero, *chukker* en 1 y reloj en 07’ 00’’. Con 30’’ de tiempo extendido y 3’ de intervalo de descanso.

1.2.1 Actualizar valores del temporizador

Valor actual

Es posible modificar el valor actual del reloj, independientemente si corresponde al reloj del *chukker* en proceso, el de tiempo extendido o bien del intervalo de descanso. Esto dependerá justamente del estado en el que se encuentre el juego, facilmente identificable por la etiqueta en pantalla de la aplicación.

Ilustración 2. Etiquetas de la aplicación según el estado del juego.

Para ello, simplemente se debe frenar el reloj (en caso de que esté activo) y mediante los botones **+** y **–** ajustarlo al valor deseado tanto en minutos como en segundos. Al iniciar nuevamente el reloj, comenzará desde el valor configurado.

Esto es de utilidad, por ejemplo, si se quisiera agregar 20 segundos al *chukker* actual por demoras en el juego. O bien para agregar un minuto más de descanso entre *chukkers*.

Valor default de *chukker* / tiempo extendido / intervalo de descanso

Es posible modificar la duración defualt de cada *chukker*, del tiempo extendido o bien del intervalo de descanso. Para ello es necesario que el juego se encuentre en el estado que se desee modificar. Es decir, que si se quisiera modificar el valor *default* del tiempo extendido, necesariamente el juego deberá estar transitando este evento. Se puede ayudar de la siguiente tabla:

Valor a modificar	Etiqueta de referencia
Duración *Chukker*	CHUKKER 2
Duración Tiempo Extendido	TIEMPO EXTENDIDO
Duración Intervalo de Descanso	INTERVALO

Tabla 1. Referencia de etiquetas según el *timer* que se desee modificar.

Identificado el estado del juego, es posible modificar el valor deseado de la siguiente forma*:*

- Frenar el reloj, en caso de que esté activo.
- Ajustar el valor mediante los botones **+** y **–** para minutos y segundos.
- Presionar el botón **Guardar**.
- La aplicación debe mostrar un mensaje, confirmando el cambio.

2 Armado y montaje de la electrónica

2.1 Materiales para el armado

- ESP32-DevKitC con módulo ESP32-WROOM-32UE (versión con antena externa).

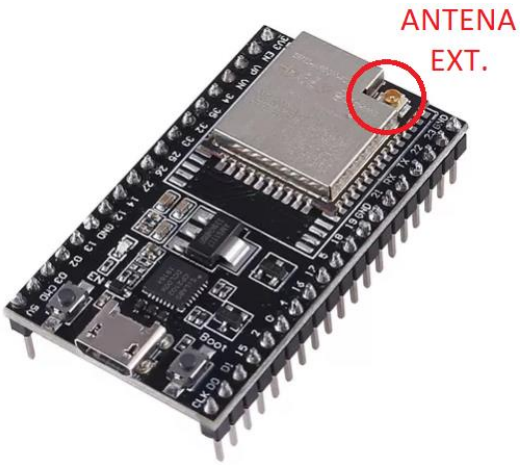


Ilustración 3. ESP32-based kit con antena externa.

- Antena WiFi 2.4GHz 12dbi.
- Conector pigtail ufl – SMA female (conexión antena-ESP32).
- Kit del tablero.
  - Segmentos LEDs.
  - Placa controladora general.
  - 1 placa de comando por cada dígito (los puntos no cuentan con placa de comando propia).
- Convertidor DC-DC 12V a 5V.
  - Step Down DC-DC **XI7015**: último módulo probado de ML, pero hay alternativas.
- *Transceiver* RS232-UART.
  - Probados módulos comerciales con IC MAX3232.
- Amplificador de audio 60W.
  - Amp Audio 60W Mono Clase D **TPA3118**: último módulo probado de ML, pero hay alternativas.
- 2 parlantes plásticos de 4''.
  - Modelo probado: 4'' 15W_{RMS} (máx. 50W) 4Ω.
- Gabinete del producto.
- Materiales y herramientas para armado y soldado.

2.2 Instructivo

Los pasos para el armado del tablero son:

1. Programar ESP32 (ver ANEXO I: PROGRAMAR ESP32).
2. Programar placa controladora (circuito LEDs).
   a. Utilizar la herramienta provista por fabricante.
   b. Programar 1 Grupo – 9 Words y brillo al 100%.

3. Setear el nivel de tensión de 5V del convertidor DC-DC.
    a. Alimentar con 12V la entrada.
    b. Medir con múltimetro a la salida.
    c. Configurar el preset para salida de $5V_{DC}$.
4. Armado de cables.
    a. Especial cuidado en caso de utilizar el cable serial RS232 que provee el fabricante de tablero de LEDs. Al menos en primeros modelos, el cable rojo se correspondía con línea de masa o GND.
5. Ensamble.

El ensamble y conexionado debe resultar de la siguiente figura:

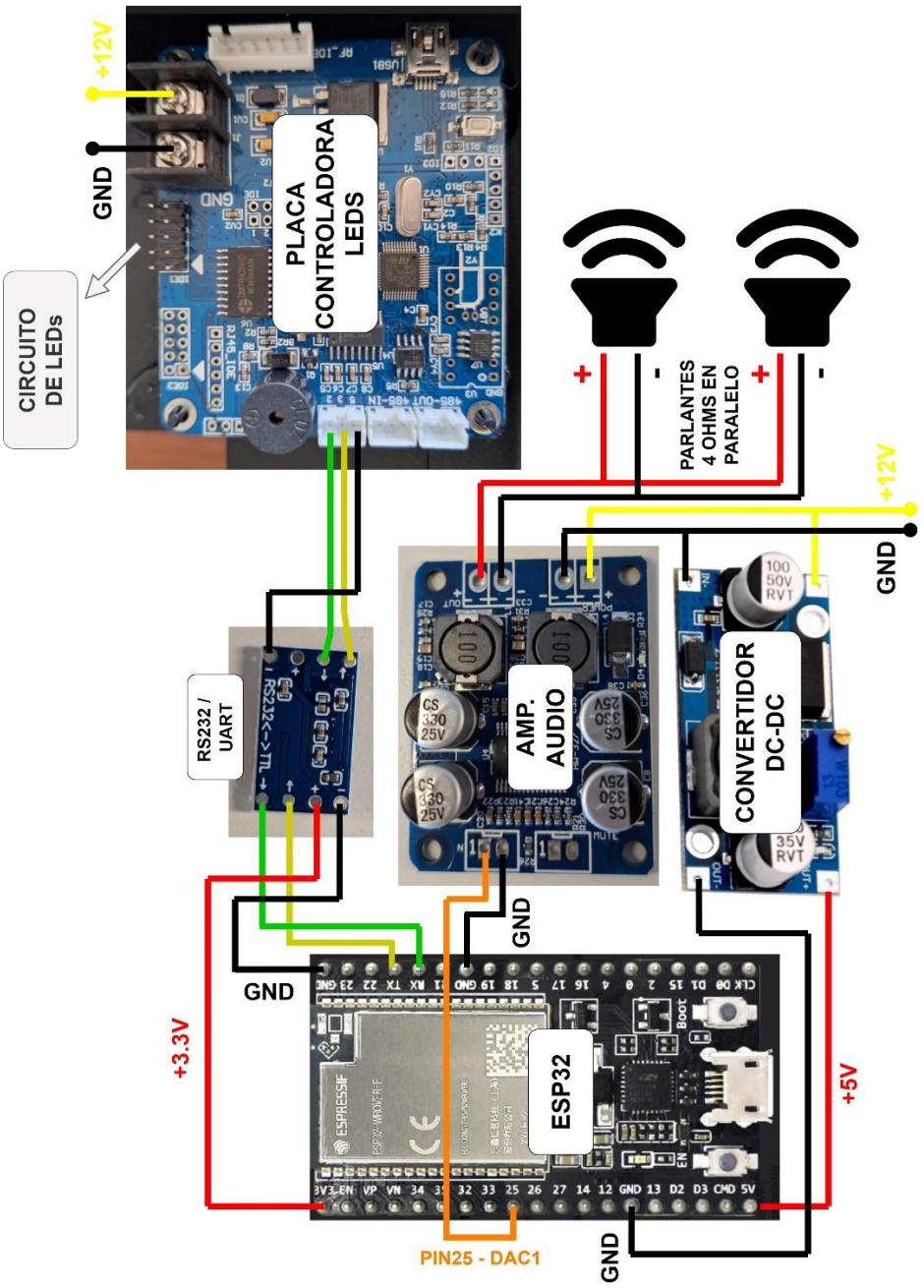


Ilustración 4. Diagrama de conexión.

Consideraciones:

- El circuito de LEDs debe conectarse en el siguiente orden, partiendo de la placa controladora y respetando la orientación que marcan las flechas del conector de cada placa y conector:
    1. PUNTAJE VISITANTE UNIDAD
    2. PUNTAJE VISITANTE DECENA
    3. *TIMER* SEGUNDOS UNIDAD
    4. *TIMER* SEGUNDOS DECENA (incluir 1er punto del *timer*)
    5. *CHUKKER*
    6. *TIMER* MINUTOS UNIDAD
    7. *TIMER* MINUTOS DECENA (incluir 2do punto del *timer*)
    8. PUNTAJE LOCAL UNIDAD
    9. PUNTAJE LOCAL DECENA

3 Software (*front-end*)

3.1 Web Server

El dispositivo ESP32 genera una red WiFi y funciona como access point. Los datos de la red son los siguientes:

- SSID: “Polo Points”
- Password: 12345678
- DOMINIO: polopoints.local

Se implementa un servidor con una serie de *endpoints* para administrar las variables y funcionalidad del tablero. Para ello, se hace uso de los siguientes comandos disponibles, que serán parámetros parte de la URL en las consultas al servidor:

- INC_SCORE_LOCAL = 1 → incrementar puntaje del local
- INC_SCORE_VISITOR = 2 → incrementar puntaje del visitante
- DEC_SCORE_LOCAL = 3 → decrementar puntaje del local
- DEC_SCORE_VISITOR = 4 → decrementar puntaje del visitante
- INC_CHUKKER = 5 → incrementar chukker (período)
- DEC_CHUKKER = 6 → decrementar chukker (período)
- START_*TIMER* = 7 → iniciar el temporizador
- STOP_*TIMER* = 8 → detener el temporizador
- RESET_*TIMER* = 9 → resetear el temporizador a valor de inicio (*default*)
- SET_CURRENT_*TIMER* = 10 → modificar el valor actual del temporizador
- SET_*DEFAULT_TIMER* = 11 → modificar duración de *chukker* del temporizador
- SET_*EXTENDED_TIMER* = 12 → modificar duración de tiempo extendido
- SET_*HALFTIME_TIMER* = 13 → modificar duración de intervalo de descanso
- RESET_ALL = 14 → resetear el tablero a valores de inicio (*default*)

Todas las solicitudes programadas en el servidor son de tipo GET. Algunas de ellas solo responden con un STATUS_CODE en función de éxito o fracaso de la acción deseada.
Otras, en caso de éxito, responden con los valores actualizados del tablero. Los datos del tablero son:

- Puntaje local
- Puntaje visitante
- Chukker

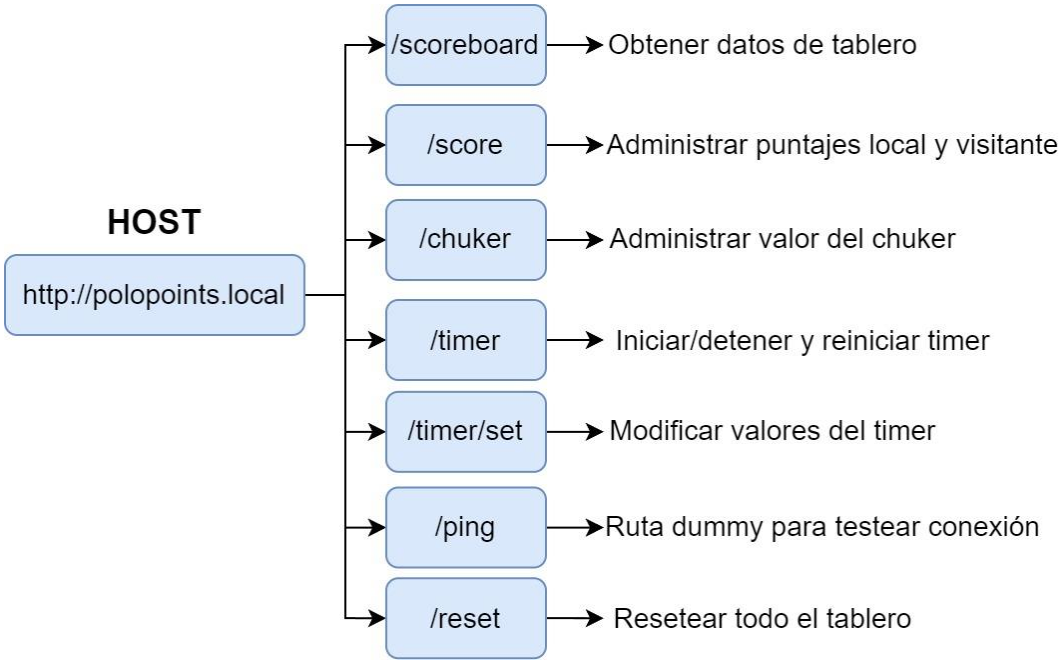
- Reloj (minutos)
- Reloj (segundos)
- Estado del timer
- Estado de la partida

El dispositivo ESP32 envía los datos como una variable de tipo *string* que concatena todos ellos, con el siguiente formato:
“puntaje-local,puntaje-visitante,valor-chukker,valor-*timer*-mm,valor-*timer*-ss,estado-*timer,*estado-partida”.

Donde el estado del temporizador (*timer*) puede ser:
- Detenido: 0
- En ejecución: 1

El estado de la partida puede ser:
- En progreso: 0
- Tiempo extendido: 1
- En descanso: 2

3.1.1 Endpoints



Resumen de *endpoints* implementados.

Obtener valores de tablero

Solicitud para obtener el estado actual de las variables del tablero: puntajes, chukker, valor actual del *timer*, estado del *timer* y estado de la partida.

Ejemplo de solicitud:

```
GET /scoreboard
Host: http://polopoints.local
```

Ejemplo de respuesta:

```
Content-type: “text/plain”
Body: { “0,0,1,7,0,0,0” }
```

Status code	Description
200 – Ok	Aceptado.

Puntajes

Solicitudes para administrar los puntajes de local y visitante. Cada solicitud permite incrementar o decrementar un determinado puntaje, definido por el parámetro que se incorpore en la URL. La solicitud devuelve el estado actual del tablero. No requiere que el *timer* se encuentre detenido.
El valor del parámetro ‘cmd’ se corresponderá con:

- INC_SCORE_LOCAL = 1
- INC_SCORE_VISITOR = 2
- DEC_SCORE_LOCAL = 3
- DEC_SCORE_VISITOR = 4

Ejemplo de solicitud (**incrementar** puntaje de **local**):

```
GET /score?cmd=1
Host: http://polopoints.local
```

Ejemplo de respuesta:

```
Content-type: “text/plain”
Body: { “1,0,3,5,22,1,0” }
```

Status code	Description
202 – Accepted	Aceptado.
404 – Bad Request	Falta parámetro o es incorrecto. Solicitud cancelada.

Chukker

Solicitudes para administrar el chukker. El mismo *endpoint* permite incrementarlo o decrementarlo, definido por el parámetro que se incorpore en la URL. La solicitud devuelve el estado actual del tablero. No requiere que el *timer* se encuentre detenido.
El valor del parámetro ‘cmd’ se corresponderá con:

- INC_CHUKKER = 5
- DEC_CHUKKER = 6

Ejemplo de solicitud (**incrementar** chukker):

```
GET /score?cmd=5
Host: http://polopoints.local
```

Ejemplo de respuesta:

```
Content-type: “text/plain”
Body: { “1,0,2,5,22,1,0” }
```

Status code	Description
202 – Accepted	Aceptado.
404 – Bad Request	Falta parámetro o es incorrecto. Solicitud cancelada.

Timer

Iniciar/detener

Solicitud para iniciar/detener el temporizador que solo devuelve como información un STATUS_CODE. No se considera como error solicitar, por ejemplo, iniciar el *timer* cuando ya se encuentra activo. Tampoco lo será detener el *timer* cuando esté detenido.
El valor del parámetro ‘cmd’ se corresponderá con:

- START_*TIMER* = 7
- STOP_*TIMER* = 8

Ejemplo de solicitud (**iniciar** *timer*):

```
GET /timer?cmd=7
Host: http://polopoints.local
```

Respuesta:

Status code	Description
202 – Accepted	Aceptado.
404 – Bad Request	Falta parámetro o es incorrecto. Solicitud cancelada.

Reiniciar

Solicitud para detener el temporizador y llevarlo al valor *default*. El valor *default* es 7’ 00’’ salvo que sea modificado a través de otra solicitud. La solicitud, en caso de éxito, devolverá el estado actual del tablero.
El valor del parámetro ‘cmd’ será siempre RESET_*TIMER* = 9.

Ejemplo de solicitud:

```
GET /timer?cmd=9
Host: http://polopoints.local
```

Ejemplo de respuesta:

```
Content-type: “text/plain”
Body: { “1,3,2,7,0,0,0” }
```

Status code	Description
202 – Accepted	Aceptado.
404 – Bad Request	Falta parámetro o es incorrecto. Solicitud cancelada.

Modificar valores

Solicitud para el temporizador que permite modificar el valor actual, duración de *chukker*, duración de tiempo extendido o duración de intervalo de descanso. Se define entre estas cuatro opciones mediante el parámetro 'cmd' de la solicitud. Los parámetros 'mm' y 'ss' representarán los valores a asignar de minutos y segundos, respetivamente.
En caso de éxito, la solicitud devolverá el estado actual del tablero. Es necesario que el temporizador se encuentre detenido para estas operaciones, caso contrario la solicitud devolverá error. Tampoco será valido intentar configurar el valor del *timer* a 00:00 o bien configurarlo cuando ya llegó a 00:00 (*timer* finalizado, ocurre luego del descanso).

- SET_CURRENT_*TIMER* = 10
- SET_*DEFAULT_TIMER* = 11
- SET_EXTENDED_TIMER = 12
- SET_HALFTIME_TIMER = 13

Valor actual [1]

Ejemplo de solicitud (modificar valor **actual** del *timer*):

```
GET /timer/set?mm=2&ss=30&cmd=10
Host: http://polopoints.local
```

Ejemplo de respuesta:

```
Content-type: "text/plain"
Body: { "0,1,1,2,30,0,0" }
```

Status code	Description
202 – Accepted	Aceptado.
404 – Bad Request	Falta parámetro o es incorrecto. Solicitud cancelada.

Duración de chukker

Ejemplo de solicitud (modificar valor ***default* de *chukker***):

```
GET /timer/set?mm=9&ss=0&cmd=11
Host: http://polopoints.local
```

Ejemplo de respuesta:

```
Content-type: "text/plain"
Body: { "0,1,1,9,0,0,0" }
```

Status code	Description
202 – Accepted	Aceptado.
404 – Bad Request	Falta parámetro o es incorrecto. Solicitud cancelada.

[1] Notar que el valor actual bien podría corresponderse con cualquiera de las sitaciones: *timer* corresponde a *chukker*, tiempo extendido o intervalo de descanso.

Valor del tiempo extendido

Ejemplo de solicitud (modificar valor **del tiempo extendido** del *timer*):

```
GET /timer/set?mm=9&ss=0&cmd=12
Host: http://polopoints.local
```

Ejemplo de respuesta:

```
Content-type: “text/plain”
Body: { “0,1,1,9,0,0,1” }
```

Status code	Description
202 – Accepted	Aceptado.
404 – Bad Request	Falta parámetro o es incorrecto. Solicitud cancelada.

Valor del intervalo de descanso

Ejemplo de solicitud (modificar valor **del intervalo de descanso** del *timer*):

```
GET /timer/set?mm=9&ss=0&cmd=13
Host: http://polopoints.local
```

Ejemplo de respuesta:

```
Content-type: “text/plain”
Body: { “0,1,1,9,0,0,2” }
```

Status code	Description
202 – Accepted	Aceptado.
404 – Bad Request	Falta parámetro o es incorrecto. Solicitud cancelada.

Resetear tablero

La solicitud frena el *timer*, en caso de que se encuentre activo, y lleva todos los valores del tablero a su valor de inicio. Salvo modificaciones, estos valores son:

- Puntaje local: 0
- Puntaje visitante: 0
- Chukker: 1
- *Timer*:
    - Duración Chukker: 7’ 00’’
    - Tiempo extendido: 30’’
    - Intervalo: 3’ 0’’

La solicitud devuelve los datos actualizados del tablero.

Ejemplo de solicitud:

```
GET /reset
Host: http://polopoints.local
```

Ejemplo de respuesta:

```
Content-type: “text/plain”
Body: { “0,0,1,7,00,0,0” }
```

Status code	Description
202 – Accepted	Aceptado.

Ping

URL dummy para testear conexión (podría utilizarse cualquiera de las otras rutas).

Ejemplo de solicitud:

```
GET /ping
Host: http://polopoints.local
```

Ejemplo de respuesta:

Status code	Description
202 – Accepted	Aceptado.

3.2 Diagrama de flujo

La aplicación en el dispositivo ESP32 puede dividirse en dos bloques sencillos: servidor web y máquina de estados que administra el tablero.
El servidor web es de tipo asincrónico, por lo que corre de fondo como un hilo independiente y administra y ejecuta todas las solicitudes que recibe el *host* (ESP32). Las solicitudes HTTP permiten actuar sobre los valores y estado del tablero. Cada vez que hay un cambio en algún dato (por ejemplo, un puntaje) a través de una solicitud, la misma solicitud informa mediante un *flag* a la máquina de estados principal de la aplicación para denotar que los LEDs del tablero deben actualizarse.
Por otro lado, la máquina de estados alterna entre tres estados:

- INIT: inicialización de tablero con valores iniciales. Se ejecuta por única vez al comienzo del programa.
- IDLE: estado de reposo. Queda a la espera de que pase 1 segundo (con *timer* activo) o que se reciba alguna solicitud HTTP que modifique algún valor del tablero. Luego invoca al siguiente estado para actualizar el tablero.
- REFRESH_SCOREBOARD: actualizar tablero (LEDs). Luego, retorna a estado de reposo IDLE.

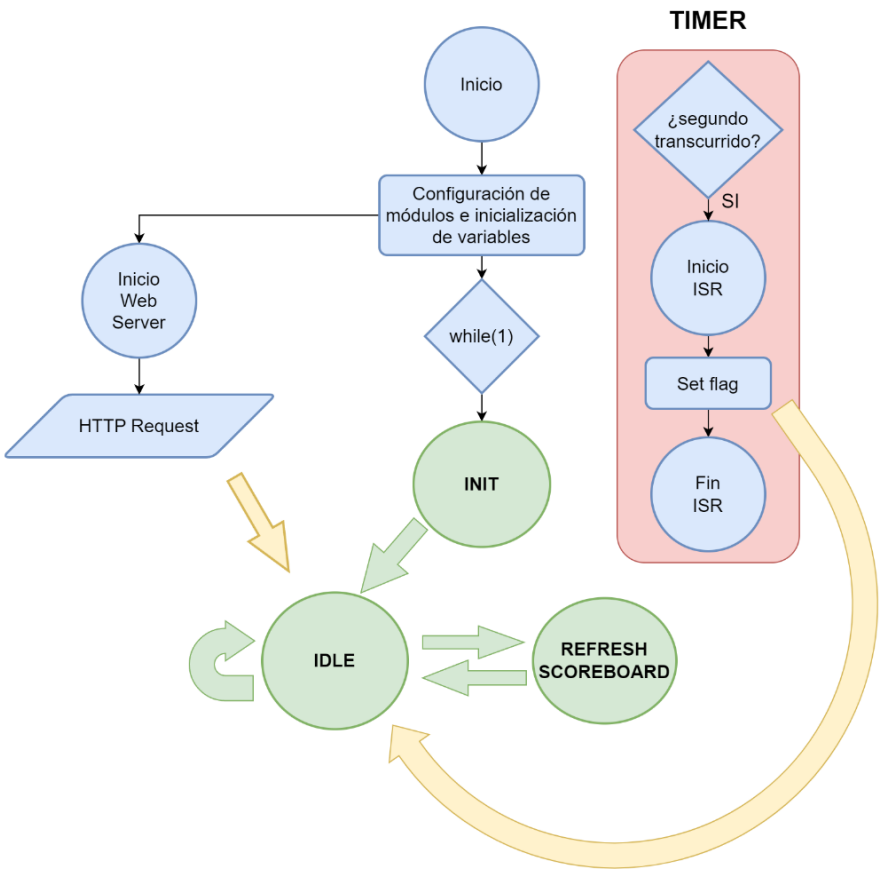


Diagrama de flujo del programa.

ANEXO I: PROGRAMAR ESP32

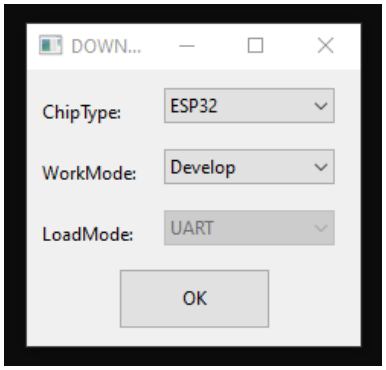
Para programar el ESP32 sin usar un IDE como Visual Studio Code o Arduino IDE, descargar la herramienta **Flash Download Tools** de ExpressIf. Para este documento, se utilizó la versión v3.9.5.

Link: https://www.espressif.com/en/support/download/other-tools

Descomprimir la herramienta **Flash Download Tools** y en la carpeta **bin** copiar los archivos binarios necesarios:

- bootloader.bin
- partitions.bin
- firmware.bin
- spiffs.bin

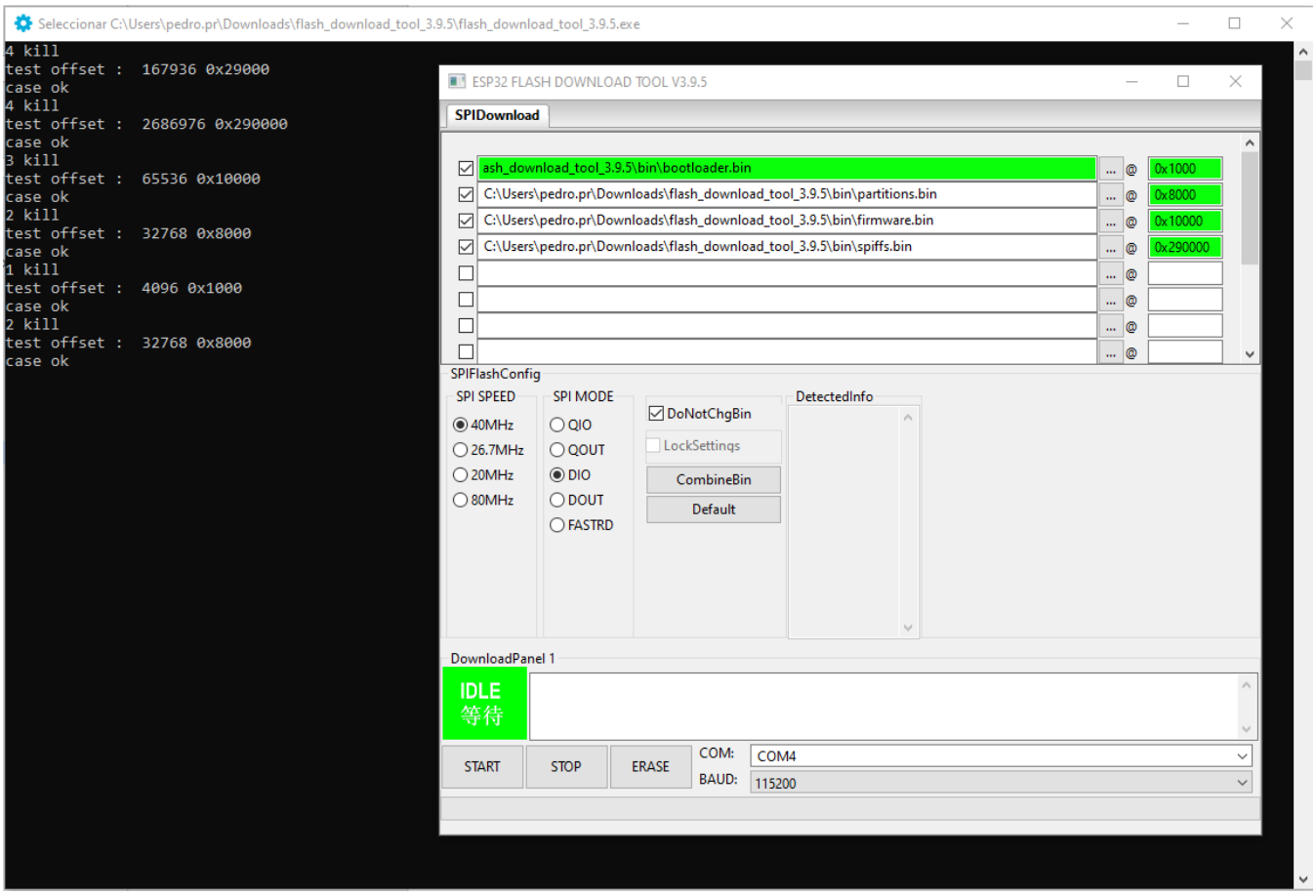
Abrir la herramienta desde su ejecutable **.exe** y especificar **ChipType → ESP32**.



Dar **OK** y en la siguiente ventana se debe cargar cada archivo binario a grabar en el dispositivo ESP32 y asignar la dirección de memoria para el grabado de cada uno de ellos:

- bootloader.bin → 0x1000
- partitions.bin → 0x8000
- firmware.bin → 0x10000
- spiffs.bin → 0x290000

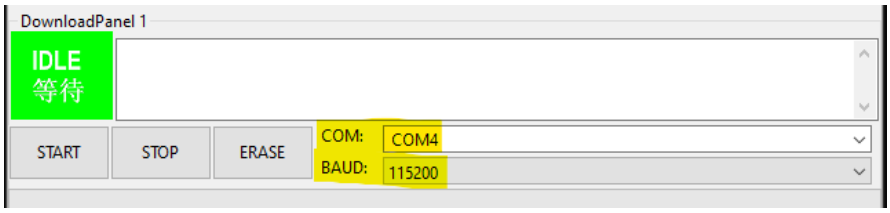
Respetar el orden de los archivos y no olvidar dar click al *checkbox* de cada línea.



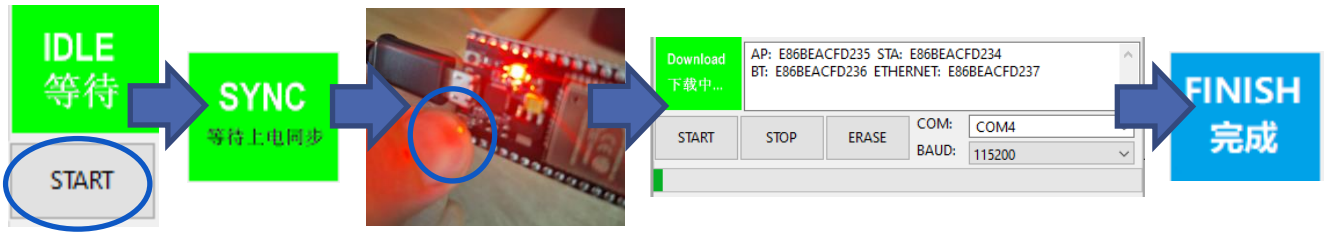
A medida que se cargan los archivos y se establece la dirección de memoria de inicio para el grabado, la terminal de la herramienta **Flash Download Tool** mostrará notificaciones de estado y la línea se coloreará con verde.

En este punto ya se encuentra todo listo para grabar la aplicación en el ESP32.

Conectar el ESP32 a la PC (si no se hizo previamente) y establecer el canal de comunicación **COM**, para este ejemplo se uso **COM4**. Dejar el Baudrate en valor default de 115200 bps.



Dar al botón **START** de la herramienta Flash Download Tool y mantener presionado el botón **BOOT** del dispositivo ESP32 hasta que ver el mensaje **Download** en la herramienta y observar una barra de carga.



Al terminar, la terminal además mostrará un mensaje “is stub and send flash finish”.

El dispositivo ESP32 ya se encuentra programado y listo para su uso. En caso de precisar corroborar el correcto grabado de la aplicación previo a su montaje, reiniciar el módulo mediante el boton de **RESET** y probar las funcionalidad de la aplicación.