Reconversión de una consola analógica para control de un rotor.

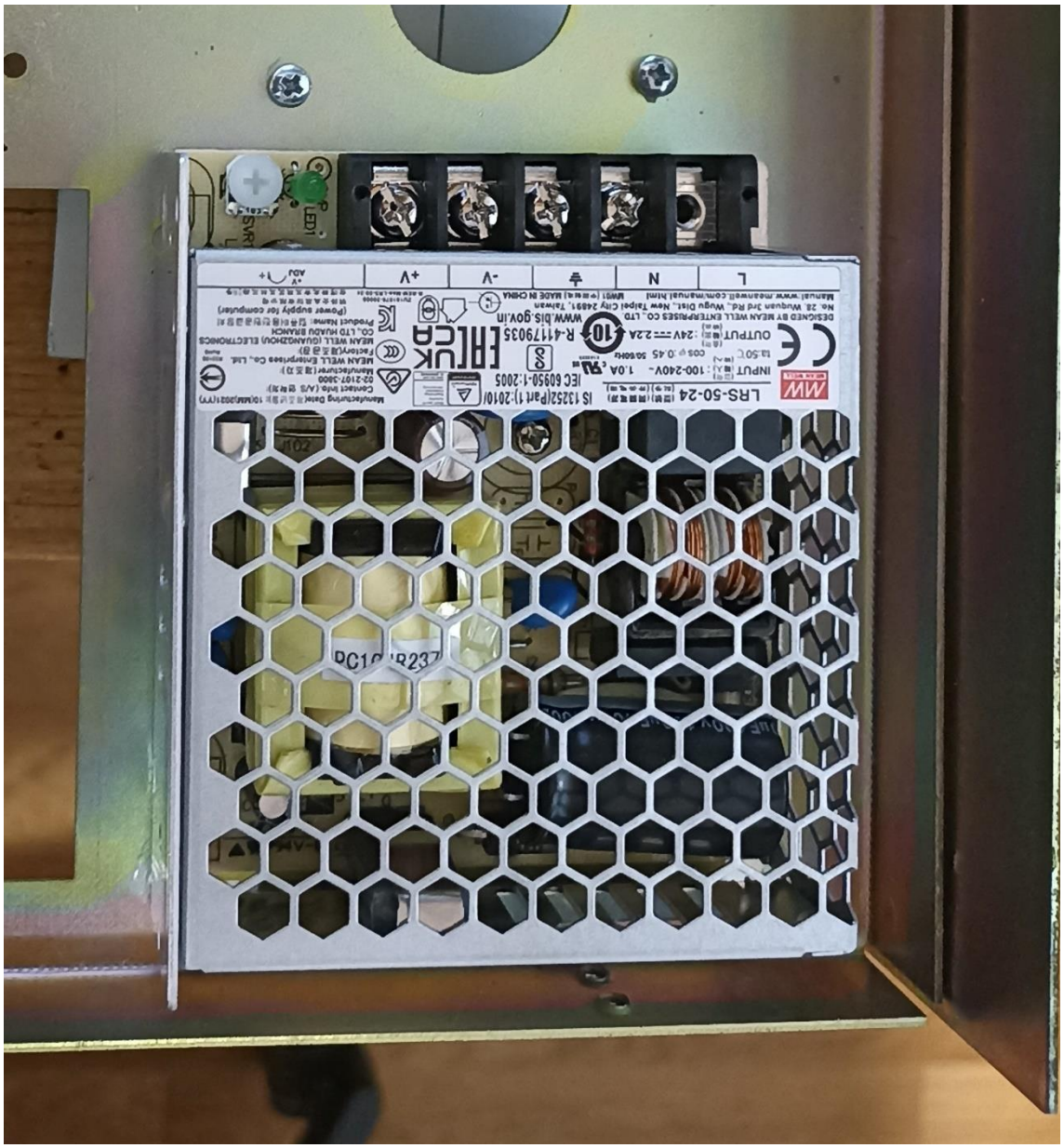
En este trabajo se describe la reconstrucción de una consola analógica CDE para el control de un rotor Yaesu G450C. Se han retirado los componentes y el voltímetro calibrado que indica la dirección del haz de la antena y se han reutilizado la caja y los mandos.

El rotor Yaesus G450C es apto para soportar antenas de VHF y de HF ligeras como, por ejemplo, una Hexbeam. Dispone de un motor de corriente continua alimentado a 20-22Vcc con un consumo inferior a 600mA.

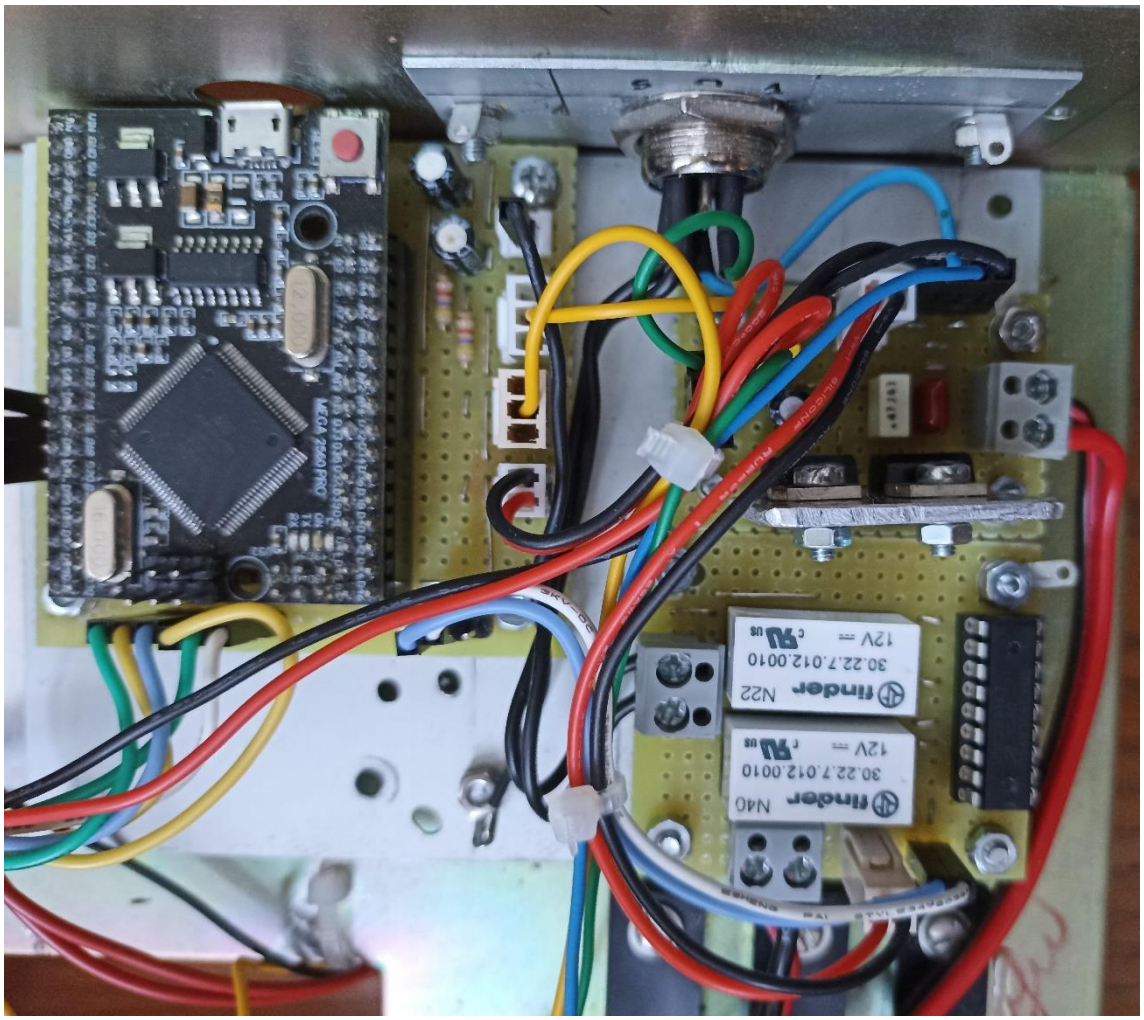
Modificaciones en el hardware. -

La consola dispone de tres pulsadores; dos para los mandos de dirección y el tercero en el centro que desbloquea el freno eléctrico, se utilizará para iniciar el giro hacia una dirección seleccionada previamente. Se ha instalado una pantalla gráfica en el hueco que deja el instrumento de señalamiento enmarcada por un bisel impreso en 3D. Se sustituye el potenciómetro de ajuste del instrumento por otro destinado a la selección previa de giro.

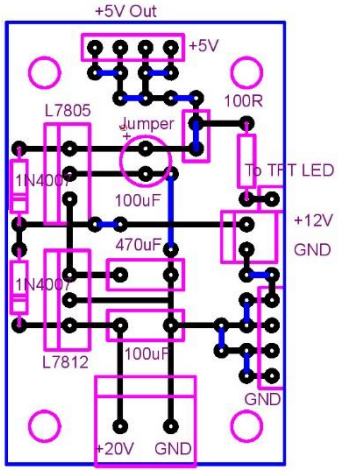
La fuente de alimentación analógica se sustituye por una Meanwell de la serie LRS-50 que se sujeta en la parte inferior de chasis. Se trata de una fuente conmutada de 24V que se puede ajustar para que entregue algo más de 22V/2A.



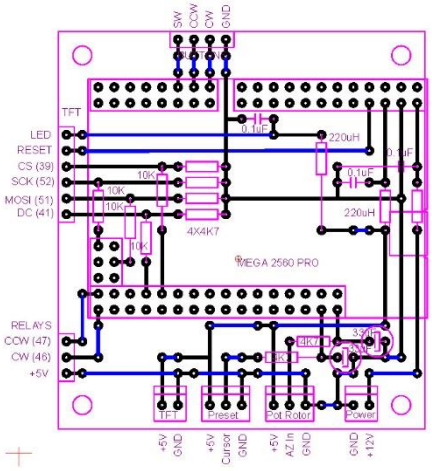
Los componentes se distribuyen en tres módulos que se instalan sobre una placa de aluminio en la parte posterior del chasis.



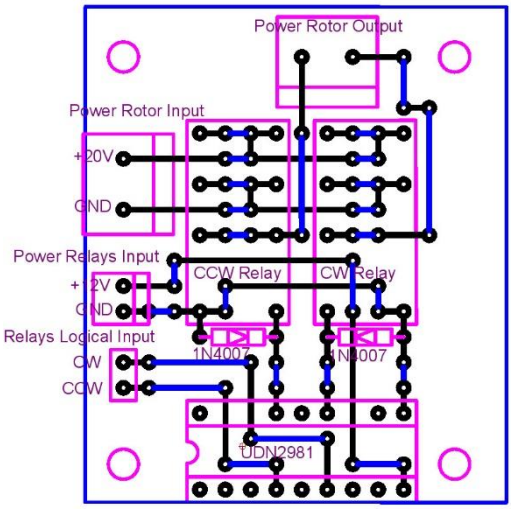
Un regulador de tensión proporciona salidas de 12Vcc para alimentar los relés y 5Vcc para el mCU, los sensores y la pantalla. El rotor utiliza los 20V de la fuente. La pantalla y la placa del mCU llevan un regulador AMS1117 de 3V3.



El mCU es un ATmega2560 instalado en una placa clónica de Arduino MEGA, dispone de memoria y capacidad de computación sobradas para gestionar el software que dirige rotor y la pantalla gráfica. Esta placa incluye varios puertos serie por hardware necesarios para un eventual control posterior desde un PC. El ATmega2560 dispone de un ADC de 10 bit estable y lineal, con suficiente sensibilidad para procesar un rango de tensión de 0 a 5Vcc.



El módulo de los relés está compuesto por dos Finder de la serie 30.22 de 12V de muy bajo consumo, capaces de conmutar hasta 125Vcc/2A. El controlador es un conmutador bipolar de ocho canales fabricado por Allegro.



Los circuitos están realizados con placas estándar perforada con orificios a 0,1'' (2,54mm). Las pistas están realizadas con alambre de cobre plateado de 0,5mm. Este tipo de placas permite cruzar las pistas como en un circuito impreso de doble cara. En el dibujo, las pistas que discurren por el lado inferior, lado del cobre, están marcadas en negro y las del lado superior, lado componentes, están macadas en azul. Algunos cambios de lado en las pistas tienen por objeto facilitar la sujeción del alambre para la soldadura. Esta forma de realizar circuitos impresos tiene la ventaja de la sencillez y la economía a cambio de una cuidadosa ejecución.

La pantalla TFT de 2,8'' incluye un controlador ILI9341 para el que existen diferentes librerías que permiten gestionar colores y dibujos básicos como líneas, triángulos y rectángulos en un panel de 320 x 240 pixeles.

El proyecto que se describe dispone de las prestaciones básicas de una consola analógica como la que se suministra de fábrica con el rotor y a las que se añaden un preselector de giro y una pantalla digital gráfica. La consola es apta para cualquier rotor que requiera los mandos manuales de giro y el control para un rotor que disponga de un motor de corriente continua.

El costo aproximado de los componentes se describe en la tabla. El precio de la MEGA puede variar hasta 32,00€ comprado en Amazon. El UDN2981 no se fabrica, pero se pueden encontrar piezas en Ali Express o sustituir por un array de Darlington como el ULN2003, incluso por un módulo de relés chino con optoacopladores o bien utilizar transistores tipo 2N2222 como conmutador. En estos casos sería necesario modificar el módulo de relés.

Componente	Procedencia	Importe
MEGA 2560 PRO	Ali Express	12,00€
Finder 30.22 (2)	Comercio local	12,00€
L78xx (2) + 1N4007 (4)	Comercio local	3,00€
UDN2981	Ali Express	2,00€
Conectores	Comercio local	4,00€
Placa circuito impreso	Comercio local	10,00€
Componentes pasivos	Comercio local	5,00€
Total		48,00€

Al tratarse de un proyecto de hardware y software libre, es posible añadir nuevas prestaciones modificando las entradas y salidas necesarias para el control automático del freno, control de la velocidad de arranque y parada o la selección de una dirección de aparcamiento. Estas modificaciones permitirían controlar rotores de mayor envergadura como las series 800 y 1000 de Yaesu o Hy Gain. Además, con la programación adecuada, es posible comunicar la consola con un PC para utilizar software de control de rotores vía puerto serie, Ethernet o WiFi añadiendo a la placa del mCU los correspondientes módulos de comunicación.

Para la interconexión de los circuitos se han utilizado terminales Berg (Dupont) en las puertas I/O digitales, Molex KK245 polarizados en los circuitos de corriente y clemas para las tensiones de 20Vcc.

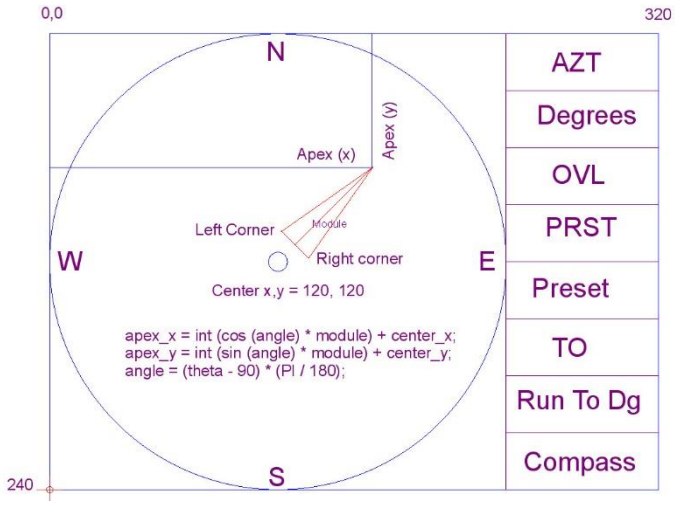
Funcionamiento. -

El motor del rotor, de corriente continua, gira en el sentido de las agujas del reloj (CW) o en sentido contrario (CCW) en función de la polarización. Dos relés se ocupan de suministrar la corriente polarizada. En reposo, ambos relés derivan los dos polos del motor a masa. Cuando se activa el relé CW o el CCW, polarizan la corriente en un sentido o en otro. Es posible utilizar un rotor cuyo motor se alimente con corriente alterna modificando la salida de los contactos de los relés.

La comunicación entre el mCU y la pantalla gráfica se realiza a través de un puerto SPI. Esta característica proporciona una velocidad de comunicación suficiente para las necesidades gráficas de la pantalla. El nivel lógico diferente del Atmega2560 (TTL, 5V) y del controlador de la pantalla (3V3), se resuelve con divisores de tensión resistivos en las salidas lógicas (SCK, MOSI, CS y DC) del puerto SPI.

Los relés se pueden activar con los pulsadores poniendo el nivel lógico de la salida correspondiente a 1. También se activan cuando se presiona el pulsador de inicio de giro si el mCU ha detectado un cambio en la dirección de preselección y ésta es diferente a la posición del rotor. En este caso, un algoritmo detecta dirección e inicia el giro que terminará cuando la dirección seleccionada y la actual coincidan o se presione de nuevo el pulsador de inicio.

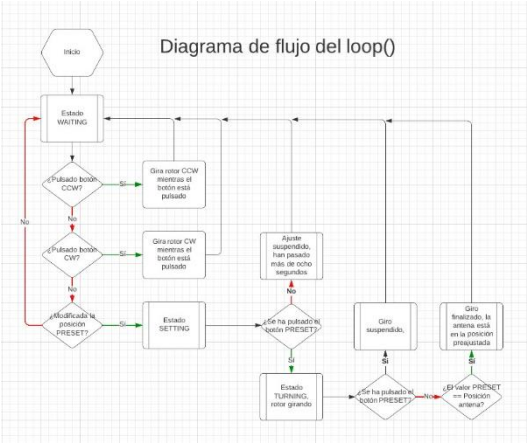
La pantalla gráfica se divide en dos segmentos, el principal dibuja una esfera con una aguja que indica en tiempo real la posición del rotor. Esto permite al operador una mejor información espacial de la posición de la antena que una pantalla matricial.



El segundo segmento indica la posición en tiempo real del rotor en grados y la posición en la rosa de los vientos de una brújula. Unas flechas en las esquinas inferiores indican la dirección de giro del rotor cuando está en marcha. También indica la dirección de preselección durante la selección y la dirección seleccionada cuando se inicia el giro.

Software. –

La programación está realizada en C++ para Arduino. La función principal, loop (), gestiona los eventos en tres modos diferentes.



WAITTING, es este modo se gestionan dos posibles eventos,

Si se pulsa el mando CCW o el mando CW, mientras el mando esté pulsado, el motor gira en sentido horario (CW) o en sentido contrario (CCW), las pantallas gráfica y matricial indican el giro y el sentido del giro.

Si se ajusta el potenciómetro de preselección de giro, la pantalla indica la dirección seleccionada y el modo pasa a,

SETTING, este modo permite seleccionar una dirección de destino. Si transcurren más de ocho segundos desde el inicio sin iniciar el giro al destino, el modo pasa automáticamente a WAITTING y se borra la pantalla de preselección (PRST).

Si se presiona el pulsador de inicio, el pulsador central, antes de que transcurran ocho segundos desde el inicio, se borra la pantalla de selección y pasa la pantalla de destino (“TO”) y el modo pasa a,

TURNING, en este modo, un algoritmo detecta el camino más corto a la dirección de destino teniendo en cuenta si es necesario el sobregiro que permite los 450° del rotor. La pantalla gráfica y matricial siguen el giro y una flecha indica la dirección. El rotor gira hasta que la dirección de la antena sea igual a la dirección preseleccionada. En este momento el modo pasa a WAITING y se borran las indicaciones de giro de la pantalla.

En caso de presionar de nuevo el pulsador de inicio antes de que termine el giro, el rotor se detiene y suspende el giro pasando a modo WAITTING.

Desde el punto de vista de programación, los principales retos que se han planteado son: la estabilidad del indicador de rumbo y el diseño gráfico del indicador utilizando los recursos gráficos de la librería.

La mayoría de los rotores ligeros y medios, como el G450C, utilizan como sensor de dirección una tensión variable mediada por un potenciómetro que actúa como divisor. Esta tensión variable, en una consola analógica, se utiliza para indicar el rumbo mediante un miliamperímetro calibrado o un motor sincronizado con el rotor.

El principal problema que plantea diseñar un instrumento tan sensible como la entrada de un mCU, es la inestabilidad de la tensión que suministra un potenciómetro con irregularidades o impurezas en la pista que roza el cursor o variaciones causadas por la temperatura de los componentes. Esta inestabilidad introduce pequeños cambios que pueden alterar lectura y provocar oscilaciones en la aguja gráfica de la pantalla en determinadas posiciones, incluso

cuando la antena está estática. Este problema no afecta a la precisión y su defecto principalmente es cosmético.

Para paliar el problema, es necesario que la fuente de referencia, de 5Vcc, esté bien filtrada y estabilizada. Además, es necesario añadir un filtro de software que consiste en una función de lectura de la entrada analógica que devuelva la media de un centenar de lecturas consecutivas.

Por otra parte, la pantalla únicamente recibe cambios de imagen cuando el rotor está en marcha. Cuando el rotor está estático no se procesan cambios en la dirección de la antena debido a una posible inestabilidad.

Otro reto que se ha necesitado superar es la programación de la posición y movimiento de la aguja en un plano de 320 x 240 pixeles. El eje de giro de la aguja está situado en las coordenadas 120,120px. El giro completo es de 360° con un grado de precisión, mientras el giro del rotor es de 450°.

Se han revisado numerosos proyectos gráficos, incluidos relojes analógicos en los que la división máxima de la circunferencia es de 60°. Pero en los trabajos consultados, las agujas han sido dibujadas como una línea. En este proyecto se ha utilizado un triángulo isósceles como aguja indicadora del rumbo. La aguja se desplaza situando el ápice del ángulo en un punto de una circunferencia de radio inferior al de las indicaciones del rumbo. Calcular las coordenadas del punto de referencia (el ápice) es sencillo utilizando la formula trigonométrica.

apex_x = cos (theta) * module + center_x;

apex_y = sin (theta) * module + center_y;

Donde apex_x, apex_y son las coordenadas del ápice en el plano, module es la longitud de la aguja, la hipotenusa del triángulo que forma con los cuadrantes de la circunferencia y center_x, center_y las coordenadas del eje en el plano de giro (120, 120px).

theta es el ángulo en radianes que indica la dirección del haz de la antena desfasado en -90° para compensar el eje del giro en el plano que plantea la posición de la pantalla. La mayor dificultad consiste en calcular las coordenadas de los ángulos opuestos en el triángulo isósceles.

La función de la librería <Adafruit_ILI9341> que permite dibujar un triángulo relleno con color es, tft.fillTriangle (apex_x, apex_y, right_corner_x, right_corner_y, left_corner_x, left_corner_y, color)

El programa (sketch) está escrito en C++, se puede descargar del repositorio del GitHub, consiste en el archivo principal rotator45C.ino, display.ccp contine los métodos de la clase display y display.h la cabecera. rotator.ccp los métodos de la clase rotator y rotator.h la cabecera.

Todas las funciones y variable están autodefinidas por lo que se omiten los comentarios, los abordajes de las funciones y métodos están comentado al principio de cada bloque.

Para poner en marcha el proyecto, únicamente es necesario descargar el sketch a la IDE de Arduino, compilarlo y descargarlo en la memoria del mCU

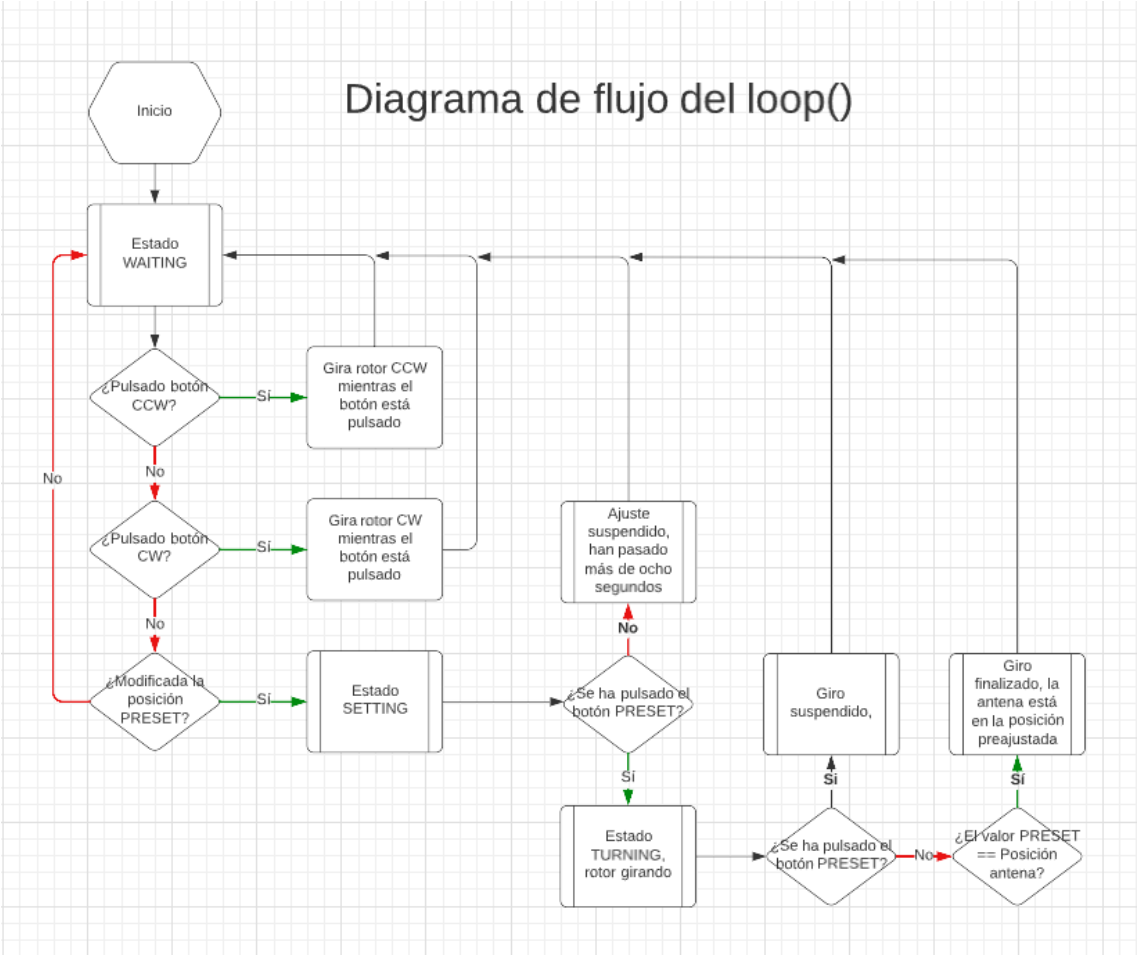


Diagrama de flujo del loop()
Inicio
Estado WAITING
¿Pulsado botón CCW?
Sí
Gira rotor CCW mientras el botón está pulsado
No
¿Pulsado botón CW?
Sí
Gira rotor CW mientras el botón está pulsado
No
¿Modificada la posición PRESET?
Sí
Estado SETTING
No
¿Se ha pulsado el botón PRESET?
Ajuste suspendido, han pasado más de ocho segundos
No
Sí
Estado TURNING, rotor girando
¿Se ha pulsado el botón PRESET?
Sí
Giro suspendido,
No
¿El valor PRESET == Posición antena?
Sí
Giro finalizado, la antena está en la posición preajustada