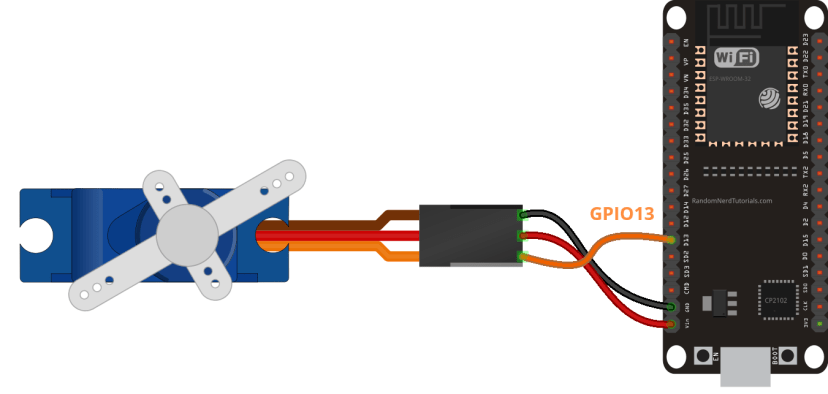
Notas: poner codigos, explicar materiales, explicar problemas, imágenes? Videos? (Version a supersucio, version final overleaf)

Pasos realizados:

1. Montaje fisico del dispositivo a programar)
2. Investigar el funcionamiento de los servos, una vez descubierto, ver como programarlos en arduino(Poner programa y explicar)
3. Explicar cambios respecto al primer programa(HTML, como se conecta al wifi…)
4. Ver todas las pobsibles soluciones encontradas y poner cual se ha usado finalmente
5. Explicar las barras y el estado
6. Imagen que contiene motor

   Descripción generada automáticamenteComo dispositivo principal se ha usado la mano robótica 5 grados de libertad QDS-1601, esto incluye toda la estructura de la propia mano además de 5 servo motores que van a ser los que muevan los dedos
7. Seguidamente se ha procedido a una primera prueba de la misma, procediendo con la prueba individual de los servos seguido de una prueba colectiva.

La conexión de los servos funciona de la siguiente manera

Para esta primera prueba se ha usado el siguiente programa:

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja

En el se incluye la librería ESP32Servo para controlar los servomotores. Esta librería permite manejar servos de manera sencilla mediante el uso de la función attach (para asociar un pin al servo) y write (para definir el ángulo de giro del servo).

Se crean objetos tipo Servo para cada dedo de la mano mecánica: meñique, anular, medio, índice y pulgar.

Mediante las directivas #define, se asignan los pines GPIO correspondientes a cada servo del ESP32.

Estas variables (outputXXState) permiten registrar el estado actual de cada servo, facilitando el control y seguimiento de sus posiciones. Aunque no se utilizan directamente en este fragmento, son útiles para proyectos más avanzados que involucren interacción dinámica.

En la función setup, se inicializan los servos asociándolos a sus respectivos pines GPIO mediante attach.

Cada servo comienza con un ángulo inicial que define la posición de reposo de los dedos. Por ejemplo:

Los dedos meñique y anular empiezan completamente abiertos (ángulo 0).

Los otros dedos empiezan ligeramente cerrados (ángulo 15).

Esta función cierra la mano mecánica parcialmente:

Dedos meñique, anular y medio se cierran a 15°.

Dedos índice y pulgar se cierran completamente a 0°.

Se utiliza un retraso de 3 segundos (delay(3000)) para mantener esta posición antes de continuar.

Abre completamente la mano:

Dedos meñique, anular y medio se abren a 0°.

Dedos índice y pulgar vuelven a 15°.

También se mantiene esta posición durante 3 segundos.

Esta función mueve los dedos de forma secuencial:

Cierra los dedos uno por uno con un retraso de 1 segundo entre cada movimiento.

Mantiene todos cerrados durante 2 segundos.

Luego, abre los dedos uno por uno, en el orden inverso.

Se utiliza un retraso final de 3 segundos.

El programa ejecuta continuamente la función dedo\_1a1, realizando los movimientos secuenciales de apertura y cierre de los dedos de la mano mecánica.

1. Inclusión de conectividad Wi-Fi

Se incorpora la biblioteca WiFi.h para conectar el ESP32 a una red inalámbrica. Además, se definen las credenciales de la red Wi-Fi con las variables:

cpp

Copiar

Editar

const char\* ssid = "Livebox6-5853";

const char\* password = "Do6ZEsZvt7vS";

En la función setup, se configura la conexión al Wi-Fi, mostrando el estado de la conexión y la dirección IP del dispositivo en el monitor serie:

cpp

Copiar

Editar

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.println("WiFi connected.");

Serial.println("IP address: ");

Serial.println(WiFi.localIP());

2. Implementación de un servidor web

Se utiliza la clase WiFiServer para crear un servidor en el puerto 80:

cpp

Copiar

Editar

WiFiServer server(80);

El servidor permite recibir solicitudes de un cliente (como un navegador web) y responder con una página HTML que contiene botones para controlar la mano mecánica.

3. Generación de la interfaz HTML

En el código se añade una respuesta HTTP que genera una página web. Esta página contiene botones para ejecutar las acciones de:

Cerrar la mano.

Abrir la mano.

Mover los dedos uno a uno.

La página se crea dentro del bloque:

cpp

Copiar

Editar

client.println("<!DOCTYPE html><html>");

client.println("<head><meta name=\"viewport\" content=\"width=device-width, initial-scale=1\">");

client.println("<style>.button { ... }</style></head>");

client.println("<body><h1>ESP32 Control de Servos</h1>");

client.println("<p><a href=\"/mano\_cerrada\"><button class=\"button\">Cerrar Mano</button></a></p>");

client.println("<p><a href=\"/mano\_abierta\"><button class=\"button\">Abrir Mano</button></a></p>");

client.println("<p><a href=\"/dedo\_1a1\"><button class=\"button\">Mover Dedo 1 a 1</button></a></p>");

client.println("</body></html>");

4. Manejo de solicitudes HTTP

El servidor identifica la acción solicitada mediante el análisis de la cabecera HTTP. Dependiendo de la URL enviada por el cliente (/mano\_cerrada, /mano\_abierta o /dedo\_1a1), ejecuta la función correspondiente para mover los servos:

cpp

Copiar

Editar

if (header.indexOf("GET /mano\_cerrada") >= 0) {

mano\_cerrada();

Serial.println("Ejecutando: Mano Cerrada");

} else if (header.indexOf("GET /mano\_abierta") >= 0) {

mano\_abierta();

Serial.println("Ejecutando: Mano Abierta");

} else if (header.indexOf("GET /dedo\_1a1") >= 0) {

dedo\_1a1();

Serial.println("Ejecutando: Mover Dedo 1 a 1");

}