Sprint #2

Fecha de desarrollo: 9 de abril / 12 de abril – 19 de abril / 22 de abril

Equipo: Juan Andrés Abella Ballén (Scrum Máster), Santiago Cortes Tovar (Product Owner), Tomás Montañez Piñeros (Developer)

Para este sprint no se realizaron modificaciones en el product backlog, por lo tanto, se cuenta con el siguiente product backlog:

Product Backlog		
	Peso	Dias
Construcción de la estructura de la CNC	1	1
Ensamblaje y acoplamiento de los motores, motobomba y guias en la estructura	1	1
Implementar el sistema de control de los actuadores del sistema	2	2
Implementación del electroiman	1	1
Implementación de sensor de humedad	1	1
Implementación de la camara	2	2
Implementacion de Solucion NLPE	5	7
Implementación del microfono	1	1
Entrenamiento de la Inteligencia Artificial	5	8
Implementar protocolo de pruebas	3	4
Comunicación entre microcontroladores y demás electrónica	2	2
Generación de red local con dirección IP	2	2
Programar riego por zonas	4	5
Programar sistema de monitoreo del estado de las plantas	4	5
Diseñar HMI y de alertas	1	1
Enviar datos y recibir datos desde conexión IP	2	2
	37	45

Para el entregable de este sprint se planeó el siguiente sprint backlog:

	Sprint 2							
Objetivo	Implementacion de sensores y demas para el control del sistema			_			PLANNING	
	Sprint Backlog	Peso	Dias			Santiago	Tomas	Juan
1	Implementación del electroiman	1	1	Conectar el electroiman a su actuador y colocarlo en el cabezal	0,1	0,1		
1	implementación del electrolman			Programar un programa de pruebas y probar	0,9	0,9		
2	Implementación de sensor de humedad	1	1	Conectar a la ESP32 y colocarlo en la estructura	0,1		0,1	
2	implementacion de sensor de numedad		1	Programar un programa de pruebas y probar	0,9		0,9	
3	2		1	Conectar a la ESP32 y colocarlo en la estructura	0,1			0,1
3	Implementación del microfono	1	1	Programar un programa de pruebas y probar	0,9			0,9
4	Implementación de la camara	2	٠	Conectar a la ESP32 y colocarlo en la estructura	0,1			0,1
4	implementación de la camara			Programar un programa de pruebas y probar	1,9	0,64	0,64	0,62
				Conectar las dos ESP32	0,1			0,1
5	Implementación de la camara	2	2	Verificar la comunicación	0,4	0,2	0,2	
				Implementar botones de control de parada de emergencia e incio	1,5	0,49	0,49	0,52
TOTAL SEMANA 2			7		7	2,33	2,33	2,34

Se cuenta con el objetivo general de este sprint el cual es:

Ajuste e implementación sensores y de componentes necesarios para el movimiento de la estructura acoplando diferentes piezas diseñadas para permitir los movimientos en el eje X, Y e Z contando con la puesta en funcionamiento de los tres motores.

El criterio de éxito para el sprint #2 el movimiento de los tres motores junto con su respectivo acople en el montaje físico.

Descripción detallada del product Backlog para el Sprint #2 y avances Sprint #1:

- Ensamblaje y acoplamiento de los motores, motobomba y guías en la estructura: Verificación de la puesta física de cada uno de los sensores y actuadores y todo lo relacionado con la funcionalidad del movimiento para permitir el desplazamiento del cabezal en los tres ejes. Además de garantizar una ubicación idónea para los sensores.
- Implementar el sistema de control de los actuadores del sistema:
 Programación del sistema de control en el microcontrolador para verificar su funcionamiento adecuado para el desplazamiento en el montaje físico.
- Implementación del sensor de humedad: Código y programación para adquirir datos de humedad.
- Implementación del micrófono: Código y programación para obtener señales de audio.
- Implementación de la cámara: Código y programación para obtener imágenes para procesamiento.
- Comunicación entre microcontroladores y demás electrónica: Código para comunicar los microcontroladores verificando el funcionamiento adecuado de su implementación junto con las demás implementaciones de sensores y actuadores.

AVANCE DE SPRINT 1

Antes de dar inicio al Sprint 2, el equipo continuó trabajando en el diseño y ensamblaje del montaje físico de la estructura CNC, lo que implicó ajustar algunas de las tareas originalmente planificadas. Como se muestra en la tabla, se reorganizó el trabajo para enfocarse en completar los elementos pendientes del Sprint 1:

	Sprint 1			(3 de Abril- 8 de Abril)								
Objetivo	Construir la estructura física y conectar los componentes principales del CNC						PLANNING		MONITORING			
	Sprint Backlog	Peso	Dias			Santiago	Tomas	Juan	Santiago	Tomas	Juan	
				Construcción de la base	0,4	0,2	0,2		0,2	0,2		
1	Construcción de la estructura de la CNC	1	1	Construcción de la estructura del cabezal	0,4			0,4			0,4	
				Construcción de la zona donde se va a almacenar la electrónica	0,2	0,2			0,1			
		1		Colocar las guias y varillas roscadas por cada eje	0,5		0,5			0		
2	2 Ensamblaje y acoplamiento de los motores, motobomba y guias en la estructura		1	Colocar los motores con cada varilla roscada	0,4	0,2		0,2	0,2		0	
				Poner la motobomba	0,1			0,1			0	
				Conectar los motores DC al puente H	0,1	0,1			0,1			
3	Implementar el sistema de control de los actuadores del sistema	2	2	Conectar el motor paso a paso al driver	0,1		0,1			0,1		
3	Implementar el sistema de control de los actuadores del sistema	2	2	Conectar la motobomba al mosfet	0,1			0,1			0	
				Probar cada conexión	1,7	0,6	0,55	0,55	0,1	0,5	0,5	
	TOTAL SEMANA 1	4	4		4	1,3	1,35	1,35	0,7	0,8	0,9	
				-					53,85%	59,26%	66,67%	

Se evidencian avances en el montaje estructural, con distribución del trabajo entre los tres integrantes del equipo:

	Sprint 1	(9 de Abril / 12 de Abril - 19 de Abril / 22 de Abril)										_
Objetivo	Construir la estructura física y conectar los componentes principales del CNC						PLANNING			MONITORING		L
	Sprint Backlog	Peso	Dias			Santiago	Tomas	Juan	Santiago	Tomas	Juan	
				Construcción de la base	0,4	0,2	0,2		0,2	0,2]
1	Construcción de la estructura de la CNC	1	1	Construcción de la estructura del cabezal	0,4			0,4			0,4	1
				Construcción de la zona donde se va a almacenar la electrónica	0,2	0,2			0,2			
				Colocar las guias y varillas roscadas por cada eje	0,5		0,5			0,2		П
2	Ensamblaje y acoplamiento de los motores, motobomba y guias en la estructura 1	1	1	Colocar los motores con cada varilla roscada	0,4	0,2		0,2	0,2		0	
				Poner la motobomba	0,1			0,1			0,05	
				Conectar los motores DC al puente H	0,1	0,1			0,1]
3	Implementar el sistema de control de los actuadores del sistema	2	,	Conectar el motor paso a paso al driver	0,1		0,1			0,1		
3	implementar el sistema de control de los actuadores del sistema	- 2	-	Conectar la motobomba al mosfet	0,1			0,1			0	
				Probar cada conexión	1,7	0,6	0,55	0,55	0,3	0,55	0,55	
	TOTAL SEMANA 1	4	4		4	1,3	1,35	1,35	1	1,05	1	
									76,92%	77,78%	74,07%	Γ

Para lo cual se puede expresar la finalización de la construcción de la estructura de la CNC, completando en su mayoría el ensamblaje y acoplamiento de los motores, motobomba, guías y los diferentes dispositivos en la maqueta física y con ello haciendo que cada uno de los integrantes complete satisfactoriamente una parte de sus tiempos y pesos de trabajo.

Con ello se tiene que se pasó de un desarrollo del 60% al 76.25% y para ser completado solo se debe completar el montaje de la CNC.

DESARROLLO SPRINT 2

Durante este segundo sprint, el enfoque principal estuvo en la **implementación**, **conexión y pruebas de los dispositivos electrónicos** que forman parte del sistema CNC. Según lo planteado en el Sprint Backlog, las tareas se centraron en validar la comunicación entre los microcontroladores y los distintos sensores que se integrarán al diseño.

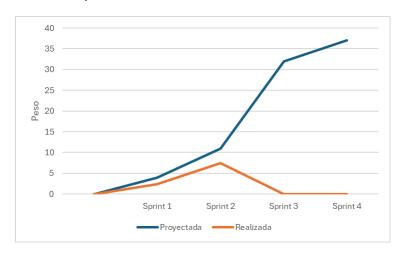
En esta etapa se trabajó en:

- La conexión y prueba del **sensor de humedad y temperatura**.
- La verificación del funcionamiento del micrófono como entrada de datos.
- La integración de la cámara, considerando calidad de captura y compatibilidad.
- La **comunicación entre los microcontroladores** y cada uno de los dispositivos mencionados.

Estas actividades permitieron avanzar significativamente en la preparación electrónica del sistema, sentando las bases para su integración final con la estructura física. Los resultados de este sprint se resumen en la siguiente tabla:

	Sprint 2]		(9 de Abril / 12 de Abril - 19 de Abril / 22 de Abril)								
	Implementacion de sensores y demas para el control del	1										1
Objetivo	sistema						PLANNING			MONITORING		
	Sprint Backlog	Peso	Dias			Santiago	Tomas	Juan	Santiago	Tomas	Juan	
1		-	-		0,1	0,1						1
-		-	1	Programar un programa de pruebas y probar	0,9	0,9						
2	Implementación de sensor de humedad	1	1	Conectar a la ESP32 y colocarlo en la estructura	0,1		0,1		0,05			
2	implementacion de sensor de numedad	1	1	Programar un programa de pruebas y probar	0,9		0,9		0,6		0,3	
3	Implementación del microfono	1	1	Conectar a la ESP32 y colocarlo en la estructura	0,1			0,1			0,05	
3	Implementacion del microlono	1	1	Programar un programa de pruebas y probar	0,9			0,9	0,2	0,1	0,6	
4	Implementación de la camara	2	2	Conectar a la ESP32 y colocarlo en la estructura	0,1			0,1	0,05			
4	implementacion de la camara		-	Programar un programa de pruebas y probar	1,9	0,64	0,64	0,62	0,2	0,2	0,2	
				Conectar las dos ESP32	0,1			0,1	0,1			
5	Comunicación entre microcontroladores y demás electrónica	2	2	Verificar la comunicación	0,4	0,2	0,2			0,3		
				Implementar botones de control de parada de emergencia e incio	1,5	0,49	0,49	0,52	0,4	0,6	0,5	L
	TOTAL SEMANA 2	7	7		7	2,33	2,33	2,34	1,6	1,2	1,65	
									68,67%	51,50%	70,51%	

En el desarrollo de este Sprint se aumentó un 16% más en el primer Sprint, por lo que se lleva un 76% completo, por otra parte, las tareas correspondientes a este segundo Sprint se realizo un trabajo total de 4.45 días, lo que corresponde a un 40% completo. Teniendo en cuanta lo anterior se tiene la siguiente curva con respecto al avance del proyecto a través de los Sprints:



El presente documento resume el desarrollo y avance correspondiente al Sprint 2 del proyecto CNC, incluyendo tareas completadas, decisiones técnicas y distribución del trabajo entre los integrantes. Se entrega este informe como constancia del progreso alcanzado hasta la fecha, quedando a la espera de observaciones y recomendaciones por parte del responsable del seguimiento.

A continuación, se firma por parte del equipo desarrollador y el evaluador del proyecto como evidencia de revisión y validación:

Firma de Scrum Master	tattatz
Firma de Product Owner	Santiago Cortes Fovar
Firma de Developer	The state of the s

Firma de Cliente

Monorso

Anexos

Programa de ESP32 para control de motores y recolección de datos del DHT11

#include "DHT.h"	const int INTERVALO = 30; // Intervalo en ms para el cambio de velocidad										
#define DHTPIN 5 // Pin donde está conectado el sensor DHT	int paso = 1; // Incremento/decremento de velocidad										
#define DHTTYPE DHT22 // Tipo de sensor DHT (DHT11,	void setup() {										
DHT22, etc.)	// Configuración motor 1										
DHT dht(DHTPIN, DHTTYPE);	pinMode(motorPin1, OUTPUT);										
// Definición de pince para el 1 200N	pinMode(motorPin2, OUTPUT);										
// Definición de pines para el L298N	// Configuración motor 2										
const int motorPin1 = 6; // Pin PWM para controlar velocidad en una dirección	pinMode(motorPin3, OUTPUT);										
const int motorPin2 = 7; // Pin PWM para controlar	pinMode(motorPin4, OUTPUT);										
velocidad en dirección opuesta	// Configuración step motor										
	pinMode(bob1, OUTPUT);										
// Definición de pines para el L298N	pinMode(bob2, OUTPUT);										
const int motorPin3 = 8; // Pin PWM para controlar velocidad en una dirección	pinMode(bob3, OUTPUT);										
const int motorPin4 = 9; // Pin PWM para controlar velocidad en dirección opuesta	pinMode(bob4, OUTPUT);										
	// Configuracion dht22										
// Definición de pines para step motor	dht.begin();										
const int bob1 = 10;											
const int bob2 = 11;	// Inicializar motor detenido										
const int bob3 = 12;	digitalWrite(motorPin1, LOW);										
const int bob4 = 13;	digitalWrite(motorPin2, LOW);										
const int mis = 2.5; // 5 10 15											
	digitalWrite(motorPin3, LOW);										
// Variables para el control del motor	digitalWrite(motorPin4, LOW);										
int velocidad1 = 0, velocidad2 = 0; // Valor PWM (0-255)											
bool direccion1 = true, direccion2 = true; // true = adelante, false = atrás	// Iniciar comunicación serial										
bool modoAutomatico = false;	Serial.begin(115200);										
char step = 'p';	Serial.println("Control de Motor L298N");										
	Serial.println("Comandos disponibles:");										
// Variables para el modo automático	Serial.println("A+velocidad - Giro en sentido horario (0-255)");										
unsigned long tiempoAnterior = 0;	Serial.println("B+velocidad - Giro en sentido antihorario (0-255)");										

```
Serial.println("C+velocidad - Giro en sentido horario (0-
                                                                       }
255)");
                                                                       // Leer temperatura y humedad
 Serial.println("D+velocidad - Giro en sentido antihorario
                                                                       float h = dht.readHumidity();
(0-255)");
                                                                       float t = dht.readTemperature();
 Serial.println("E - Giro en sentido horario (step motor)");
                                                                       Serial.print("Humedad: ");
 Serial.println("F - Giro en sentido antihorario (step
motor)");
                                                                       Serial.print(h);
 Serial.println("S - Detener motor");
                                                                       Serial.print("%, Temperatura: ");
 Serial.println("M - Activar/desactivar modo automático");
                                                                       Serial.print(t);
                                                                       Serial.println("°C");
                                                                      }
void loop() {
 // Leer comandos del serial
                                                                       void procesarComando(String comando) {
 if (Serial.available() > 0) {
                                                                       comando.trim(); // Eliminar espacios en blanco
  String comando = Serial.readStringUntil('\n'); // Leer
hasta nueva línea
                                                                       if (comando.length() > 0) {
 procesarComando(comando);
                                                                         char primerCaracter = comando.charAt(0);
                                                                         step = primerCaracter;
                                                                         // Detener modo automático si se recibe cualquier
 // Control automático si está activo
                                                                       comando
 if (modoAutomatico) {
                                                                         if (primerCaracter != 'M' && modoAutomatico) {
 modoVariacionAutomatica();
                                                                         modoAutomatico = false;
}
                                                                         Serial.println("Modo automático desactivado");
 if (step == 'E' || step == 'e') {
  stepdireccion(bob1, bob2, bob3, bob4, mis);
                                                                         switch (primerCaracter) {
 Serial.println("Step motor girando en sentido horario");
                                                                         case 'A':
 } else if (step == 'F' || step == 'f') {
                                                                         case 'a':
 stepdireccion(bob4, bob3, bob2, bob1, mis);
                                                                          // Giro en sentido horario
 Serial.println("Step motor girando en sentido
                                                                           if (comando.indexOf('+') != -1) { // devuelve la posición
antihorario");
                                                                       del primer '+' encontrado
 } else if (step == 'P' || step == 'p'){
                                                                           velocidad1 =
                                                                       comando.substring(comando.indexOf('+') + 1).toInt(); //
  digitalWrite(bob1, LOW);
                                                                       extraer velocidad
  digitalWrite(bob2, LOW);
                                                                           velocidad1 = constrain(velocidad1, 0, 255); // Limitar
  digitalWrite(bob3, LOW);
                                                                      velocidad entre 0 y 255
  digitalWrite(bob4, LOW);
                                                                           moverMotor1(true, velocidad1); // mover motor hacia
                                                                       adelante
  Serial.println("step motor detenido");
                                                                           Serial.print("Motor 1 girando hacia adelante con
 } else {
                                                                      velocidad: ");
  Serial.println("Comando no reconocido");
                                                                           Serial.println(velocidad1);
```

```
}
                                                                        case 'C':
   break;
                                                                        case 'c':
   case 'B':
                                                                         // Giro en sentido horario para motor 2
   case 'b':
                                                                          if (comando.indexOf('+') != -1) {
   // Giro en sentido antihorario
                                                                          velocidad2 =
                                                                      comando.substring(comando.indexOf('+') + 1).toInt();
   if (comando.indexOf('+') != -1) {
                                                                          velocidad2 = constrain(velocidad2, 0, 255);
    velocidad1 =
comando.substring(comando.indexOf('+') + 1).toInt();
                                                                          moverMotor2(true, velocidad2); // mover motor hacia
                                                                      adelante
    velocidad1 = constrain(velocidad1, 0, 255);
                                                                           Serial.print("Motor 2 girando hacia adelante con
    moverMotor1(false, velocidad1);
                                                                      velocidad: ");
    Serial.print("Motor 1 girando hacia atrás con
                                                                          Serial.println(velocidad2);
velocidad: ");
    Serial.println(velocidad1);
                                                                          break;
   break;
                                                                        case 'D':
                                                                        case 'd':
   case 'S':
                                                                          // Giro en sentido antihorario para motor 2
   case 's':
                                                                          if (comando.indexOf('+') != -1) {
   // Detener motor
                                                                          velocidad2 =
    detenerMotor();
                                                                      comando.substring(comando.indexOf('+') + 1).toInt();
    Serial.println("Motores detenidos");
                                                                          velocidad2 = constrain(velocidad2, 0, 255);
    break;
                                                                          moverMotor2(false, velocidad2);
                                                                           Serial.print("Motor 2 girando hacia atrás con
                                                                      velocidad: ");
   case 'M':
                                                                          Serial.println(velocidad2);
   case 'm':
                                                                         }
   // Activar/desactivar modo automático
                                                                          break;
    modoAutomatico = !modoAutomatico;
    if (modoAutomatico) {
                                                                        default:
    Serial.println("Modo automático activado");
                                                                          Serial.println("Comando no reconocido");
    velocidad1 = 0; // Iniciar desde velocidad 0
                                                                          break;
    paso = 1; // Iniciar incrementando
   } else {
                                                                       }
    Serial.println("Modo automático desactivado");
    detenerMotor();
                                                                      void moverMotor1(bool direc, int vel) {
    break;
                                                                       if (direc) {
```

```
// Dirección 1 (adelante)
  analogWrite(motorPin1, vel);
                                                                     void modoVariacionAutomatica() {
 analogWrite(motorPin2, 0);
                                                                       unsigned long tiempoActual = millis(); // Obtener tiempo
 } else {
 // Dirección 2 (atrás)
                                                                       // Cambiar velocidad cada INTERVALO milisegundos
 analogWrite(motorPin1, 0);
                                                                       if (tiempoActual - tiempoAnterior >= INTERVALO) {
 analogWrite(motorPin2, vel);
                                                                       tiempoAnterior = tiempoActual; // Actualizar tiempo
                                                                      anterior
 direccion1 = direc;
                                                                       // Cambiar dirección de incremento/decremento en los
                                                                      límites
 velocidad1 = vel;
                                                                       if (velocidad1 >= 255) {
                                                                        paso = -1; // Comenzar a decrementar
                                                                        } else if (velocidad1 <= 0) {
void moverMotor2(bool direc, int vel){
                                                                        paso = 1; // Comenzar a incrementar
 if (direc) {
                                                                        // Cambiar dirección de giro cuando llega a 0
 // Dirección 1 (adelante)
                                                                        direccion1 = !direccion1;
 analogWrite(motorPin3, vel);
 analogWrite(motorPin4, 0);
 } else {
                                                                        // Actualizar velocidad
 // Dirección 2 (atrás)
                                                                        velocidad1 += paso;
 analogWrite(motorPin3, 0);
                                                                        velocidad1 = constrain(velocidad1, 0, 255);
 analogWrite(motorPin4, vel);
 }
                                                                        // Aplicar al motor
                                                                        moverMotor1(direccion1, velocidad1);
 direccion2 = direc;
 velocidad2 = vel;
                                                                        // Mostrar información cada 10 pasos para no saturar el
                                                                      puerto serial
                                                                        if (velocidad1 % 10 == 0) {
void detenerMotor() {
                                                                        Serial.print("Auto - Dirección: ");
 analogWrite(motorPin1, 0);
                                                                        Serial.print(direccion1?"Adelante":"Atrás");
 analogWrite(motorPin2, 0);
                                                                        Serial.print(", Velocidad: ");
 analogWrite(motorPin3, 0);
                                                                        Serial.println(velocidad1);
 analogWrite(motorPin4, 0);
 // Reiniciar variables
                                                                      }
 velocidad1 = 0;
                                                                     }
 velocidad2 = 0;
```

```
void stepdireccion(int pin1, int pin2, int pin3, int pin4, int
                                                                        digitalWrite(pin1, LOW);
ms){
                                                                        digitalWrite(pin2, LOW);
 Serial.println("step motor");
                                                                        digitalWrite(pin3, HIGH);
 digitalWrite(pin1, HIGH);
                                                                        digitalWrite(pin4, LOW);
 digitalWrite(pin2, LOW);
                                                                        delay(ms);
 digitalWrite(pin3, LOW);
                                                                        digitalWrite(pin1, LOW);
 digitalWrite(pin4, LOW);
                                                                        digitalWrite(pin2, LOW);
 delay(ms);
                                                                        digitalWrite(pin3, HIGH);
 digitalWrite(pin1, HIGH);
                                                                        digitalWrite(pin4, HIGH);
 digitalWrite(pin2, HIGH);
                                                                        delay(ms);
 digitalWrite(pin3, LOW);
                                                                        digitalWrite(pin1, LOW);
 digitalWrite(pin4, LOW);
                                                                        digitalWrite(pin2, LOW);
 delay(ms);
                                                                        digitalWrite(pin3, LOW);
 digitalWrite(pin1, LOW);
                                                                        digitalWrite(pin4, HIGH);
 digitalWrite(pin2, HIGH);
                                                                        delay(ms);
 digitalWrite(pin3, LOW);
                                                                        digitalWrite(pin1, HIGH);
 digitalWrite(pin4, LOW);
                                                                        digitalWrite(pin2, LOW);
 delay(ms);
                                                                        digitalWrite(pin3, LOW);
 digitalWrite(pin1, LOW);
                                                                        digitalWrite(pin4, HIGH);
 digitalWrite(pin2, HIGH);
                                                                        delay(ms);
 digitalWrite(pin3, HIGH);
 digitalWrite(pin4, LOW);
 delay(ms);
```

Programa de ESP32 para implementación de micrófono

```
#include <driver/i2s.h>
                                                                   .communication_format = I2S_COMM_FORMAT_I2S,
                                                                   .intr_alloc_flags = ESP_INTR_FLAG_LEVEL1,
#define I2S_WS 25 // Word Select (WS)
                                                                   .dma_buf_count = 4,
#define I2S_SD 32 // Serial Data (SD)
                                                                   .dma_buf_len = 64,
#define I2S_SCK 33 // Serial Clock (SCK)
                                                                   .use_apll = false,
                                                                   .tx_desc_auto_clear = false,
void setupI2S() {
                                                                   .fixed_mclk = 0
 const i2s_config_t i2s_config = {
                                                                  };
  .mode = i2s_mode_t(I2S_MODE_MASTER |
I2S_MODE_RX),
                                                                  const i2s_pin_config_t pin_config = {
  .sample_rate = 16000,
                                                                   .bck_io_num = I2S_SCK,
  .bits_per_sample = I2S_BITS_PER_SAMPLE_32BIT,
                                                                   .ws_io_num = I2S_WS,
  .channel_format = I2S_CHANNEL_FMT_ONLY_LEFT,
                                                                   .data_out_num = I2S_PIN_NO_CHANGE,
```

```
.data_in_num = I2S_SD
                                                                       size_t bytesRead;
 };
                                                                       // Leer datos del micrófono
 i2s_driver_install(I2S_NUM_0, &i2s_config, 0, NULL);
                                                                       i2s_read(I2S_NUM_0, (void*)sampleBuffer,
                                                                      sizeof(sampleBuffer), &bytesRead, portMAX_DELAY);
 i2s_set_pin(I2S_NUM_0, &pin_config);
 i2s\_zero\_dma\_buffer(I2S\_NUM\_0);
                                                                       int samples = bytesRead / sizeof(int32_t);
                                                                       for (int i = 0; i < samples; i++) {
void setup() {
                                                                        // Convertir a valores más pequeños para graficar
 Serial.begin(115200);
                                                                      fácilmente
 setupI2S();
                                                                        int value = sampleBuffer[i] >> 14; // Reduce resolución
                                                                      (ajusta si lo ves muy plano)
 delay(500);
                                                                        Serial.println(value);
                                                                                                   // Una sola línea para que el
                                                                      Serial Plotter funcione bien
void loop() {
                                                                      }
int32_t sampleBuffer[64];
```

Codigo para ESP32 para implementación de cámara

```
// Non-FIFO OV7670 Video on TFT with ESP32
                                                              // SSCB_SDA(SIOD) -> 21(ESP32)
                                                              // SSCB_SCL(SIOC) -> 22(ESP32)
#define MODE QVGA // 320 X 240
                                                              // RESET
                                                                           -> 3.3V
//#define MODE QQVGA // 160 x 120
                                                              // PWDN
                                                                                     -> GND
//#define MODE QCIF // 176 x 144 (crop)
                                                              // HREF
                                                                                     -> NC
//#define MODE QQCIF // 88 x 72 (crop)
                                                               const camera_config_t cam_conf = {
                                                               .D0 = 36,
#define COLOR RGB565
                                                               .D1 = 39,
//#define COLOR YUV422
                                                               .D2 = 34,
                                                               .D3 = 35,
#define ROTATION 1 // 0~4
                                                               .D4 = 32,
                                                               .D5 = 33,
#include <Wire.h>
                                                               .D6 = 25,
#include <SPI.h>
                                                               .D7 = 26,
#include <OV7670.h>
                                                               .XCLK = 27,
#include <Adafruit_GFX.h>
                                                                .PCLK = 14,
#include <Adafruit_ILI9341.h>
                                                               .VSYNC = 13,
                                                               .xclk_freq_hz = 10000000, // XCLK 10MHz
// OV7670 pins
                                                                .ledc_timer = LEDC_TIMER_0,
```

```
. ledc\_channel = LEDC\_CHANNEL\_0
                                                                  Serial.printf("cam MID = %X\n\r", cam.getMID());
};
                                                                  Serial.printf("cam PID = %X\n\r", cam.getPID());
// TFT pins
                                                                  switch (MODE) {
#define TFT_DC 16
                                                                   case QVGA:
#define TFT_CS 5
                                                                    w = 320;
#define TFT_RST 17
                                                                    h = 240;
#define TFT_MISO 19
                                                                    break;
#define TFT_MOSI 23
                                                                   case QQVGA:
#define TFT_CLK 18
                                                                    w = 160;
                                                                    h = 120;
Adafruit_ILI9341 tft = Adafruit_ILI9341(TFT_CS, TFT_DC,
                                                                    break;
TFT_RST);
                                                                   case QCIF:
OV7670 cam;
                                                                    w = 176;
uint16_t *buf, w, h;
                                                                    h = 144;
                                                                    break;
void setup() {
                                                                   case QQCIF:
 setCpuFrequencyMhz(240);
                                                                    w = 88;
                                                                    h = 72;
 Serial.begin(9600);
                                                                  }
                                                                 }
 Wire.begin();
 Wire.setClock(400000);
                                                                 void loop(void) {
 tft.begin();
                                                                  for (uint16_t y = 0; y < h; y++) {
 tft.setRotation(ROTATION);
                                                                   buf = cam.getLine(y + 1);
 tft.fillScreen(0);
                                                                   tft.drawRGBBitmap(0, y, buf, w, 1);
                                                                  }
 esp_err_t err = cam.init(&cam_conf, MODE, COLOR);
                                                                 }
 if (err != ESP_OK) Serial.println("cam.init ERROR");
 cam.setPCLK(2, DBLV_CLK_x4);
 cam.vflip(false);
```