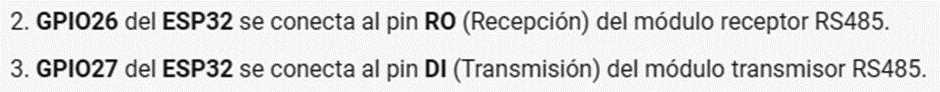
El variador de frecuencia se conectó con el RS485 mediante sus salidas que son 485, al RS485 se le conecto una fuente con 5V de continua para alimentarla.  
  
Las conexiones serian de la siguiente manera:

* Del variador de frecuencias salen 2 cables, uno definido como A (485+) y otro como B (485-) conectados al RS485.
* El RS485 tiene 2 entradas, una llamada VCC y otra llamada GND que ahi van conectados los cables de alimentación de la fuente, VCC a positivo y GND a negativo.



Luego de poner todas las conexiones de manera correcta en 2 protoboards se conectó a la computadora mediante un cable de datos y se programó por la aplicación Arduino IDE.

Antes de empezar a programar aclaramos los parámetros del variador en base al pdf para su correcto funcionamiento al momento de recibir y mandar datos, se ajustaron los siguientes parámetros

Seteo de la Comunicación:

Grupo H (pág 58):

H0-01: 0004

baudrate = 57600 | dataFormat = 1-8-2-N RTU

H0-02: 1

Address 1 para el variador de Roll; address 2 para el variador de pitch

H0-05: 0

-------------------------------

-------------------------------

Configuración para modificar los parámetros POR comunicación:

Grupo b (pág 40):

b0-01: 9 Master freq set en Communication

b0-03: 10 Auxiliary freq set en Communication

b1-00: 2 Run command en Communication Control

b1-01: 111 Binding of run command and freq setting en Communication input

:

Después averiguamos cuáles eran los comandos de prueba para poner en el arduino IDE que son los siguientes:

Comandos de Prueba:

Versa:

010603020000284E

Reversa:

010603020001E98E

35.00Hz:

010602020DAC2D5F

50.00Hz:

01060202138824E4

10.00Hz

0106020203E8290C

00.00Hz

01060202000029B2

Rampa a 20

010604010014D935

Despues usamos el siguiente codigo en el Arduino IDE:

#include <SoftwareSerial.h>

// Definición de pines para RE y DE

#define RE 32

#define DE 33

// Configuración del puerto SoftwareSerial

SoftwareSerial mod(26, 27); // RX, TX

void setup() {

Serial.begin(115200);

Serial.println("Iniciando...");

mod.begin(9600); // Configuración de Modbus

pinMode(RE, OUTPUT);

pinMode(DE, OUTPUT);

digitalWrite(RE, LOW); // Habilitar recepción

digitalWrite(DE, LOW); // Habilitar recepción

}

void loop() {

Serial.println("Enviando comando Modbus...");

// Ejemplo de comando Modbus para establecer la frecuencia a 30 Hz

byte ModWriteBuffer[8] = {0x01, 0x06, 0x00, 0x01, 0x00, 0x2135, 0x49, 0xC5}; // Cambia 0x001E por la frecuencia deseada

byte BufferValue[8];

// Enviar comando Modbus

digitalWrite(DE, HIGH); // Habilitar transmisión

digitalWrite(RE, HIGH); // Habilitar transmisión

delay(10);

mod.write(ModWriteBuffer, sizeof(ModWriteBuffer));

delay(10);

digitalWrite(DE, LOW); // Habilitar recepción

digitalWrite(RE, LOW); // Habilitar recepción

// Leer respuesta

int i = 0;

while (mod.available()) {

BufferValue[i] = mod.read();

i++;

}

// Imprimir respuesta en el Monitor Serie

Serial.print("Respuesta Modbus: ");

for (int j = 0; j < i; j++) {

Serial.print(BufferValue[j], HEX);

Serial.print(" ");

}

Serial.println();

delay(5000); // Esperar 5 segundos antes de enviar el siguiente comando

}

Obviamente descargamos las librerías necesarias para poder programar correctamente que son las siguientes:

* ArduinoModbus (de Arduino)
* ArduinoRS485 (de Arduino)
* EspSoftwareSerial (de Dirk Kaar, Peter Lerup)
* Modbus (de UL DARA)
* Modbus master (de Doc Walker)

También en el gestor de placas se descargo lo siguiente:

* Arduino AVR Boards (de Arduino)
* Arduino ESP32 Boards (de Arduino)
* ESP32 (de espressif Systems)

Se definió el ESP32 como ESP32 dev module en el Arduino IDE

No se llegó a ver la respuesta a dicho código todavía.