

PORTADA:

S k y l F



Integrantes

- Esteban Lautaro.
- Rubio Santiago Gabriel.
- Meabrio Lucas David.
- Flores Leandro.
- Brizuela Agustín.
- Leiva Santiago.
- Romocordoba Emiliano.
- Godoy Baldovino Jofiel

Contacto

Lautaro Esteban:

Mail: Lautyesteban14@gmail.com

Linkedin: Lautaro Sebastian Esteban

Instagram: @lauty.esteban

Santiago Rubio:

Mail: Santy201205@gmail.com

Linkedin: Santiago Rubio

Instagram: @santy_201205

Lucas Meabrio:

Mail: Meabriolucas@gmail.com

Linkedin: Lucas Meabrio

Instagram: @lucass_meab

Leandro Flores:

Mail: leandro200flores@gmail.com

Linkedin: Leandro Flores

Instagram: @lean.floresss

Agustin Brizuela:

Mail: brizuu750@gmail.com

Linkedin: Agustin Lionel Brizuela

Instagram: @agustiin.brizuela

Santiago Leiva:

Mail: Santiagoleiva745@gmail.com

Linkedin: Santiago Leiva

Instagram: @leivva.s

Emiliano Romocordoba:

Mail: Romoemiliano324@gmail.com

Linkedin:

Instagram:

Jofiel Godoy Baldovino:

Mail: Markbaldj@gmail.com

Linkedin: Marco Baldovino

Instagram: @jofiel_godoy

ÍNDICE

[Descripción del Proyecto:](#)

[Variadores de Frecuencia:](#)

[Resumen del objetivo.](#)

[Lista de materiales usados.](#)

[Paneles de Control:](#)

Descripción del Proyecto

Nuestro proyecto se enfocará en el armado y reacondicionamiento de un simulador de vuelo con el fin de recrear una experiencia inmersiva y entretenida para el que se meta dentro de la cabina.

Variadores de Frecuencia

Resumen del objetivo

El objetivo principal con el uso del variador de frecuencia es lograr una comunicación eficiente tanto con el programa de control como con los motores del simulador. Este tipo de comunicación es crucial para garantizar un funcionamiento sincronizado y preciso entre el software y el hardware. En nuestro caso, la comunicación se establece utilizando el protocolo **Modbus**, un estándar de comunicación ampliamente utilizado en sistemas industriales debido a su fiabilidad y flexibilidad.

El protocolo Modbus permite el intercambio de datos entre el variador de frecuencia y el microcontrolador **ESP32**, facilitando el control de parámetros clave como la velocidad y la dirección de los motores. Esta programación la estamos llevando a cabo mediante la plataforma **Arduino IDE**, donde se desarrollan los códigos que gestionan la interacción entre los distintos componentes del sistema. A través de Arduino IDE, se ha configurado el protocolo Modbus para que las señales enviadas desde el programa de simulación sean interpretadas correctamente por el variador de frecuencia, y a su vez, estas se traduzcan en comandos que controlan el movimiento de los motores.

La integración exitosa de este sistema es fundamental para garantizar que la simulación de vuelo sea lo más realista posible, permitiendo que los motores respondan con precisión a las condiciones simuladas, como cambios de altitud, aceleración y maniobras.

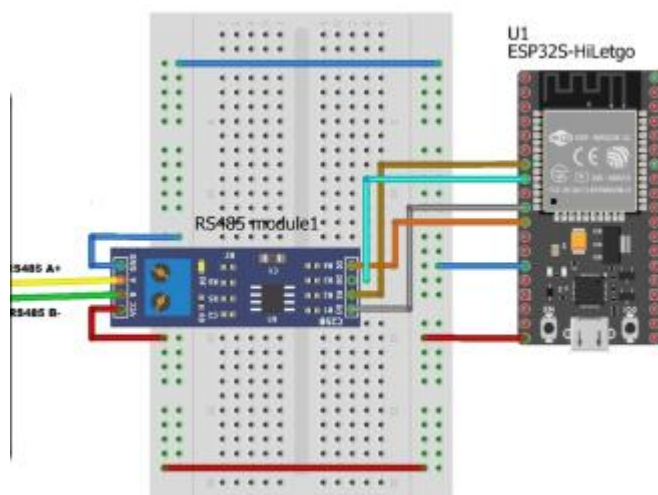
Lista de materiales usados

Usamos los siguientes materiales para este proceso actualmente:

- Módulo de RS485
- ESP32 de 30 pines
- MPU6050
- Fuente calibrada a 5V
- 8 cables macho-macho
- 2 protoboards
- Cable de datos

Comunicaciones

Comunicación entre el ESP32 y el RS485:



- MAX485 TTL a RS485 >> VCC >> +5V de ESP32
- MAX485 TTL A RS485 >> GND >> GND de ESP32
- MAX485 TTL a RS485 >> RO >> GPIO26 de ESP32 (SoftwareSerial RX)
- MAX485 TTL a RS485 >> DI >> GPIO27 de ESP32 (SoftwareSerial TX)

2. **GPIO26** del **ESP32** se conecta al pin **RO** (Recepción) del módulo receptor RS485.

3. **GPIO27** del **ESP32** se conecta al pin **DI** (Transmisión) del módulo transmisor RS485.

Comunicación entre el RS485 y el Variador de frecuencia (GK500):

- **Pin A (A+)** del módulo RS485: Conéctalo al terminal **U** (fase U) del variador de frecuencia.
 - **Pin B (B-)** del módulo RS485: Conéctalo al terminal **V** (fase V) del variador de frecuencia.
- Pin A va conectado al terminal “485+”
 - Pin B va conectado al terminal “485-”

- Conecta el pin **SCL** del **MPU6050** al pin **D1 (GPIO8)** del **ESP32**.
- Conecta el pin **SDA** del **MPU6050** al pin **D2 (GPIO9)** del **ESP32**.
- Conecta el pin **VCC** del **MPU6050** a **3.3V** del **ESP32**.
- Conecta el pin **GND** del **MPU6050** a **GND** del **ESP32**.

ESP32

De 38 pines:

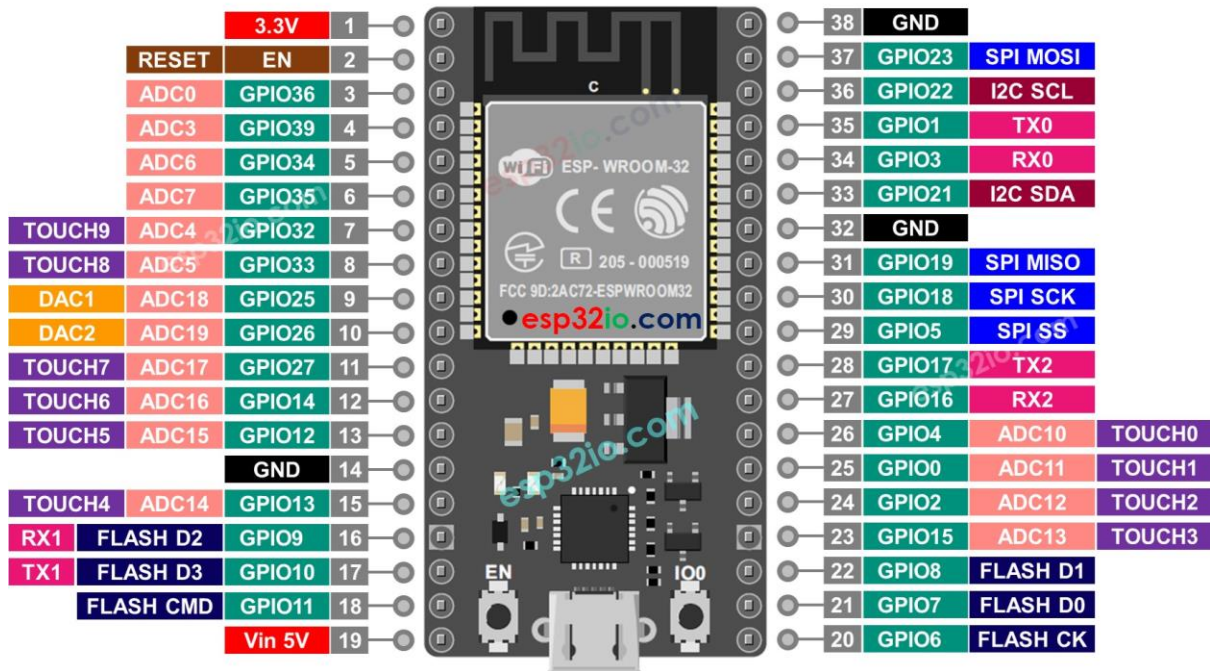


Table 2-1. Pin Overview

Name	No.	Type	Function
Analog			
VDDA	1	P	Analog power supply (2.3 V ~ 3.6 V)
LNA_IN	2	I/O	RF input and output
VDD3P3	3	P	Analog power supply (2.3 V ~ 3.6 V)
VDD3P3	4	P	Analog power supply (2.3 V ~ 3.6 V)
VDD3P3_RTC			
SENSOR_VP	5	I	GPIO36, ADC1_CH0, RTC_GPIO0
SENSOR_CAPP	6	I	GPIO37, ADC1_CH1, RTC_GPIO1
SENSOR_CAPN	7	I	GPIO38, ADC1_CH2, RTC_GPIO2
SENSOR_VN	8	I	GPIO39, ADC1_CH3, RTC_GPIO3
CHIP_PU	9	I	High: On; enables the chip Low: Off; the chip shuts down Note: Do not leave the CHIP_PU pin floating.
VDET_1	10	I	GPIO34, ADC1_CH6, RTC_GPIO4
VDET_2	11	I	GPIO35, ADC1_CH7, RTC_GPIO5
32K_XP	12	I/O	GPIO32, ADC1_CH4, RTC_GPIO9, TOUCH9, 32K_XP (32.768 kHz crystal oscillator input)
32K_XN	13	I/O	GPIO33, ADC1_CH5, RTC_GPIO8, TOUCH8, 32K_XN (32.768 kHz crystal oscillator output)
GPIO25	14	I/O	GPIO25, ADC2_CH8, RTC_GPIO6, DAC_1, EMAC_RXD0
GPIO26	15	I/O	GPIO26, ADC2_CH9, RTC_GPIO7, DAC_2, EMAC_RXD1
GPIO27	16	I/O	GPIO27, ADC2_CH7, RTC_GPIO17, TOUCH7, EMAC_RX_DV
MTMS	17	I/O	GPIO14, ADC2_CH6, RTC_GPIO16, TOUCH6, EMAC_TXD2, HSPICLK, HS2_CLK, SD_CLK, MTMS

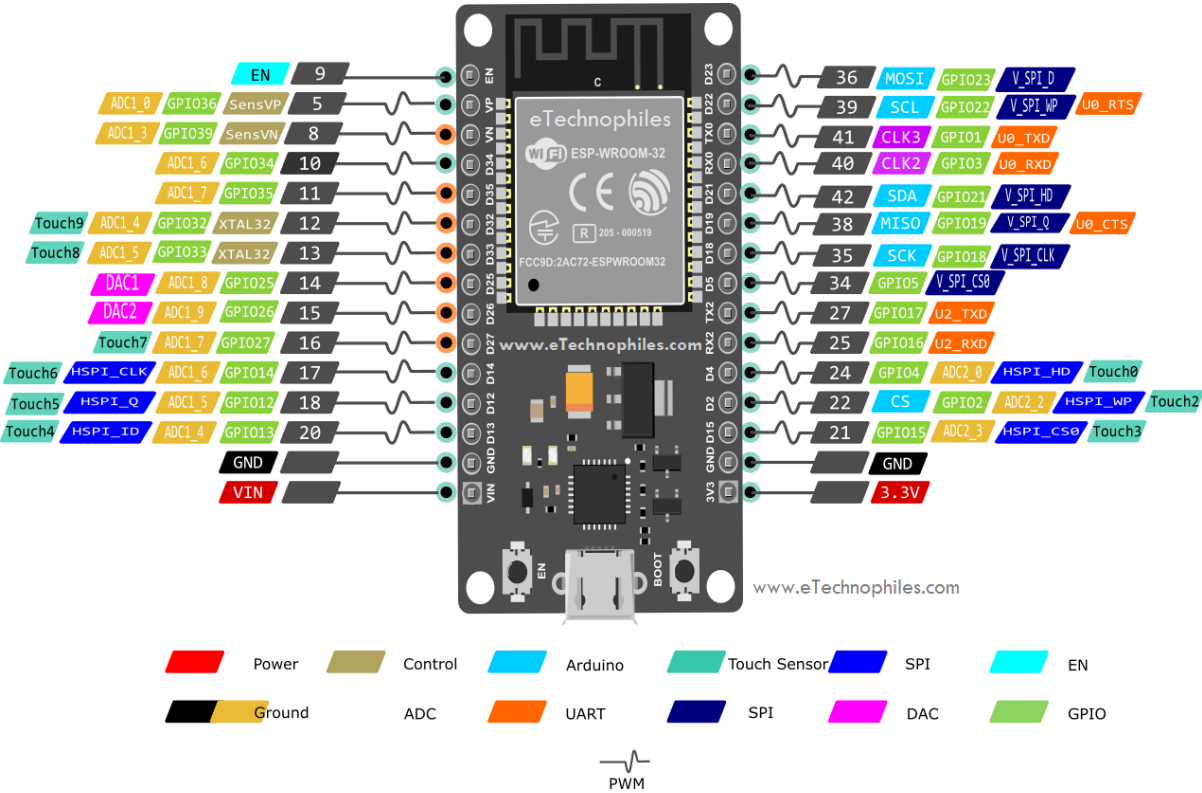
MTDI	18	I/O	GPIO12, ADC2_CH5, RTC_GPIO15, TOUCH5, EMAC_TXD3, HSPIQ, HS2_DATA2, SD_DATA2, MTDI
VDD3P3_RTC	19	P	Input power supply for RTC IO (2.3 V ~ 3.6 V)
MTCK	20	I/O	GPIO13, ADC2_CH4, RTC_GPIO14, TOUCH4, EMAC_RX_ER, HSPID, HS2_DATA3, SD_DATA3, MTCK
MTDO	21	I/O	GPIO15, ADC2_CH3, RTC_GPIO13, TOUCH3, EMAC_RXD3, HSPICSO, HS2_CMD, SD_CMD, MTDO

Name	No.	Type	Function
GPIO2	22	I/O	GPIO2, ADC2_CH2, RTC_GPIO12, TOUCH2, HSPiWP, HS2_DATA0, SD_DATA0
GPIO0	23	I/O	GPIO0, ADC2_CH1, RTC_GPIO11, TOUCH1, EMAC_TX_CLK, CLK_OUT1,
GPIO4	24	I/O	GPIO4, ADC2_CH0, RTC_GPIO10, TOUCH0, EMAC_TX_ER, HSPiHD, HS2_DATA1, SD_DATA1
VDD_SDIO			
GPIO16	25	I/O	GPIO16, HS1_DATA4, U2RXD, EMAC_CLK_OUT
VDD_SDIO	26	P	Output power supply: 1.8 V or the same voltage as VDD3P3_RTC
GPIO17	27	I/O	GPIO17, HS1_DATA5, U2TXD, EMAC_CLK_OUT_180
SD_DATA_2	28	I/O	GPIO9, HS1_DATA2, U1RXD, SD_DATA2, SPIHD
SD_DATA_3	29	I/O	GPIO10, HS1_DATA3, U1TXD, SD_DATA3, SPIWP
SD_CMD	30	I/O	GPIO11, HS1_CMD, U1RTS, SD_CMD, SPICSO
SD_CLK	31	I/O	GPIO6, HS1_CLK, U1CTS, SD_CLK, SPICLK
SD_DATA_0	32	I/O	GPIO7, HS1_DATA0, U2RTS, SD_DATA0, SPIQ
SD_DATA_1	33	I/O	GPIO8, HS1_DATA1, U2CTS, SD_DATA1, SPID

VDD3P3_CPU			
GPIO5	34	I/O	GPIO5, HS1_DATA6, VSPICSO, EMAC_RX_CLK
GPIO18	35	I/O	GPIO18, HS1_DATA7, VSPICLK
GPIO23	36	I/O	GPIO23, HS1_STROBE, VSPID
VDD3P3_CPU	37	P	Input power supply for CPU IO (1.8 V ~ 3.6 V)
GPIO19	38	I/O	GPIO19, UOCTS, VSPIQ, EMAC_TXD0
GPIO22	39	I/O	GPIO22, UORTS, VSPiWP, EMAC_TXD1
UORXD	40	I/O	GPIO3, UORXD, CLK_OUT2
UOTXD	41	I/O	GPIO1, UOTXD, CLK_OUT3, EMAC_RXD2
GPIO21	42	I/O	GPIO21, VSPIHD, EMAC_TX_EN
Analog			
VDDA	43	P	Analog power supply (2.3 V ~ 3.6 V)
XTAL_N	44	O	External crystal output
XTAL_P	45	I	External crystal input
VDDA	46	P	Analog power supply (2.3 V ~ 3.6 V)
CAP2	47	I	Connects to a 3.3 nF (10%) capacitor and 20 kΩ resistor in parallel to CAP1

Name	No.	Type	Function
CAP1	48	I	Connects to a 10 nF series capacitor to ground
GND	49	P	Ground

De 30 pines:



Precio:

Nuevo | 2 vendidos
Placa Wifi Iot Esp32 Devkit V1

1 / 4



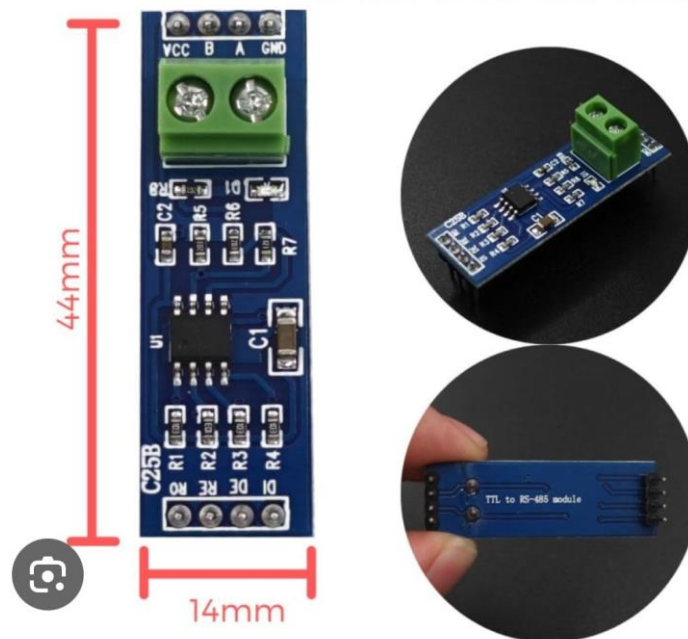
\$ 9.990

en 6 cuotas de \$ 2.370¹³

[Ver los medios de pago](#)

RS485

MODULO RS485



conexión en modo emisor es la siguiente



conexión en modo receptor es la siguiente



Si queremos que durante la conexión el convertor RS485 pueda cambiar su papel de emisor a receptor (conexión half duplex) simplemente tenemos que conectar los pines RE y DE a una salida digital para poder cambiar su tensión de Gnd a Vcc.



Nuevo | +500 vendidos

4.3 ★★★★★ (8)

Modulo Conversor Rs485 Ttl Max485 Transceiver Arduino

1 / 7



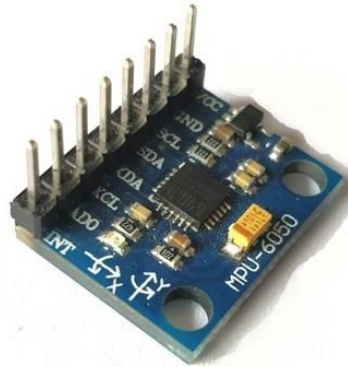
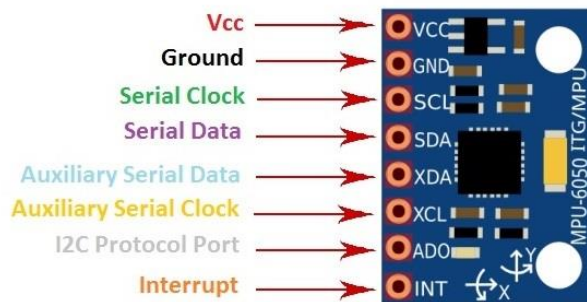
\$ 2.449

[Ver los medios de pago](#)

MPU6050

El MPU6050 es una unidad de medición inercial o IMU (Inertial Measurement Units) de 6 grados de libertad, pues combina un acelerómetro de 3 ejes y un giroscopio de 3 ejes. Este sensor es muy utilizado en navegación, geometría, estabilización, etc. Este componente nos sirve para inclinar la cabina en sus 3 ejes, basándose en las mediciones de los SimVars.

Introduction to MPU6050



www.TheEngineeringProjects.com



\$ 3.730

[Ver los medios de pago](#)

Listado de parámetros que pusieron en el variador

Parámetros	Designación	Rango	Rango puesto
b0-01	Fuente de Mando de Frecuencia Maestra	0: Configuración Digital (b0-02) + \wedge /v Ajuste en Panel de Control 1: Configuración Digital (b0-02) + Ajuste por Bornera UP/DOWN 2: Entrada Analógica AI 3: Potenciómetro 6: Salida de PID 8: Comando Multivelocidad 9: Comunicación	9 9: Comunicación
b0-02	Configuración Digital de Frecuencia Maestra	Límite Inferior ~ Límite Superior de Frecuencia	50,00 Hz
b1-00	Comando RUN	0: Control por Panel de Control 1: Control por Bornera 2: Control por Comunicación	2 2: Control por Comunicación
b1-01	Enlace de comando run y configuración frecuencia	Unidades: Fuente de configuración frecuencia agrupada bajo control panel de control: 0: Sin enlace 1: Configuración Digital (b0-02) + \wedge /v Ajuste en Panel de Control 2: Configuración Digital (b0-02) + Ajuste por Bornera UP/DOWN 3: AI 4: Potenciómetro 7: Salida de PID 9: Comando Multivelocidad A: Entrada de Comunicación Decenas: Fuente de	AAA A: Entrada de Comunicación

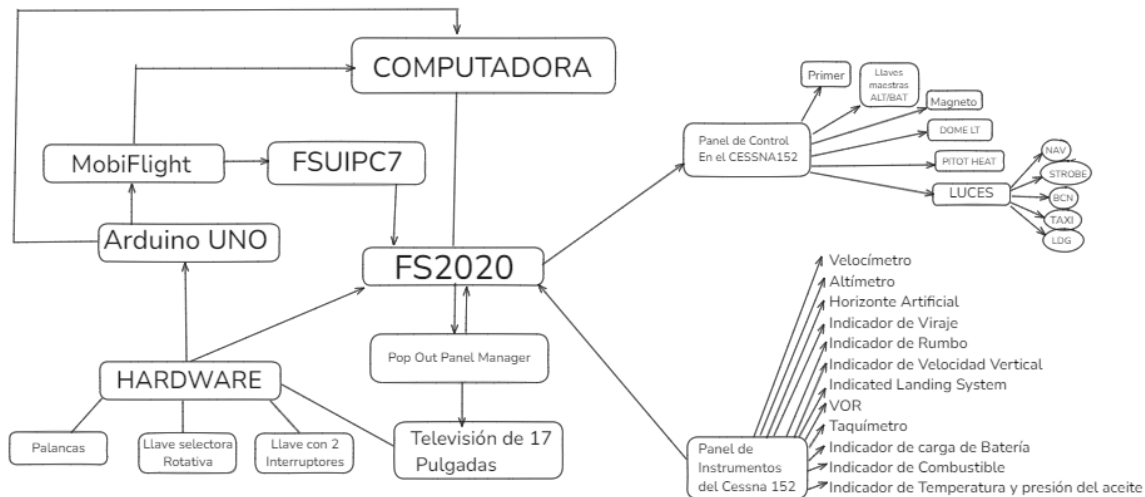
Parámetros	Designación	Rango	Rango puesto
		configuración frecuencia agrupada bajo control terminal (igual a unidades) Centenas: Fuente de configuración frecuencia agrupada bajo control de comunicación (igual a unidades)	
H0-01	Configuración del Puerto de Comunicación RS-485	Unidades: Velocidad en Baudios 0: 4800 bps 1: 9600 bps 2: 19200 bps 3: 38400 bps 4: 57600 bps Decenas: Formato de datos 0: Formato 1-8-2-N, RTU 1: Formato 1-8-1-E, RTU 2: Formato 1-8-1-O, RTU 3: Formato 1-7-2-N, ASCII 4: Formato 1-7-1-E, ASCII 5: Formato 1-7-1-O, ASCII Centenas: Tipo de conexión 0: Conexión cable directo (232/485) 1: MODEM (232) Unidades de Mil: Almacenamiento 0: Sin almacenar ante Pérdida de Energía 1: Almacenado ante Pérdida de Energía	0001 1: 9600bps
H0-05	Opción Maestro/Esclavo	0: Se usa independientemente 1: Como Maestro 2: Como Esclavo	2 2: Como Esclavo

Paneles de Control

El simulador posee dos paneles que cumplen su respectiva función; el panel de control (mediante palancas y botones interactuando con el entorno virtual) y el panel de instrumentos (estos se van a mostrar mediante una tele que transmite los datos reales de vuelo en los instrumentos)

El panel de control cuenta con todo el Hardware que se comunicará con el software a utilizar, (que en nuestro caso sería el Flight Simulator 2020).

Diagrama de Bloques del Panel de Control



Panel de Control

En el panel de control hay 7 interruptores de palanca, una llave selectora rotativa de 6 estados y una llave con dos interruptores.

Para que estos componentes se accionen en simultáneo al vuelo que se haga en el Flight Simulator 2020, a través de una herramienta llamada "MobiFlight" y dos Arduino UNO. Con el MobiFlight asignamos a cada componente a que realice determinada acción dentro del FS2020. Y con la Arduino UNO, en conjunto al MobiFlight, ejecuta esa acción a realizar

A continuación, explicaremos cuáles son estos componentes a utilizar y cuál es su función dentro del entorno virtual del FS2020:

Interruptores de palanca:



- 7 teclas de navegación:
 - Tecla 1 "Dome LT": Controla las luces interiores del avión. Se usa para iluminar la cabina durante el vuelo nocturno o en condiciones de poca luz.
 - Tecla 2 "Pitot Hit": Activa el sistema de calefacción del tubo Pitot.
 - Tecla 3 "Nav": Enciende las luces de navegación, que son las luces de posición en las alas y la cola.
 - Tecla 4 "Strobe": Activa las luces estroboscópicas, son luces blancas intermitentes en las alas.
 - Tecla 5 "BCN": Enciende la Luz de baliza (beacon), una luz roja intermitente ubicada en la parte superior o inferior del fuselaje. Si el motor este encendido, esta luz siempre se encenderá.

- Tecla 6 “Taxi”: Enciende la luz de taxi, es una luz delantera usada para iluminar la pista mientras el avión rueda por el suelo.
- Tecla 7 “LDG”: Controla la luz de aterrizaje, usada para iluminar la pista durante el aterrizaje o despegue.
- Primer: se encarga de inyectar combustible a los cilindros del motor antes de arrancarlo, facilitando el arranque en condiciones especialmente frías. (Generalmente se bombea 2 a 3 veces).



- Llaves:



- Llave maestra ALT: esta se encarga de encender el alternador del motor para poder cargar la batería de la aeronave.
- Llave maestra Bat On/Off: esta se encarga de prender la batería para que pueda alimentar los circuitos eléctricos de la aeronave.

Magneto (con llave selectora rotativa de 6 estados):



- Magneto: este cuenta con 5 estados:
 - Estado 1 “OFF”: Ambos magnetos están apagados, y el motor no recibirá chispa.
 - Estado 2 “Right”: Solo el magneto derecho estará encendido.
 - Estado 3 “Left”: Solo el magneto Izquierdo estará encendido.
 - Estado 4 “Both”: Ambos magnetos están activos (posición normal durante el vuelo)
 - Estado 5 “Start”: Esta posición activa el motor de arranque y ambos magnetos, usada solo para el encendido del motor

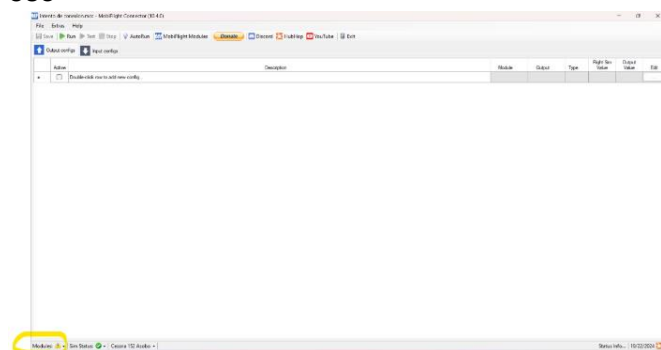
Instrucciones de Configuración

Vinculación del MobiFlight con el FS2020 y el Arduino UNO

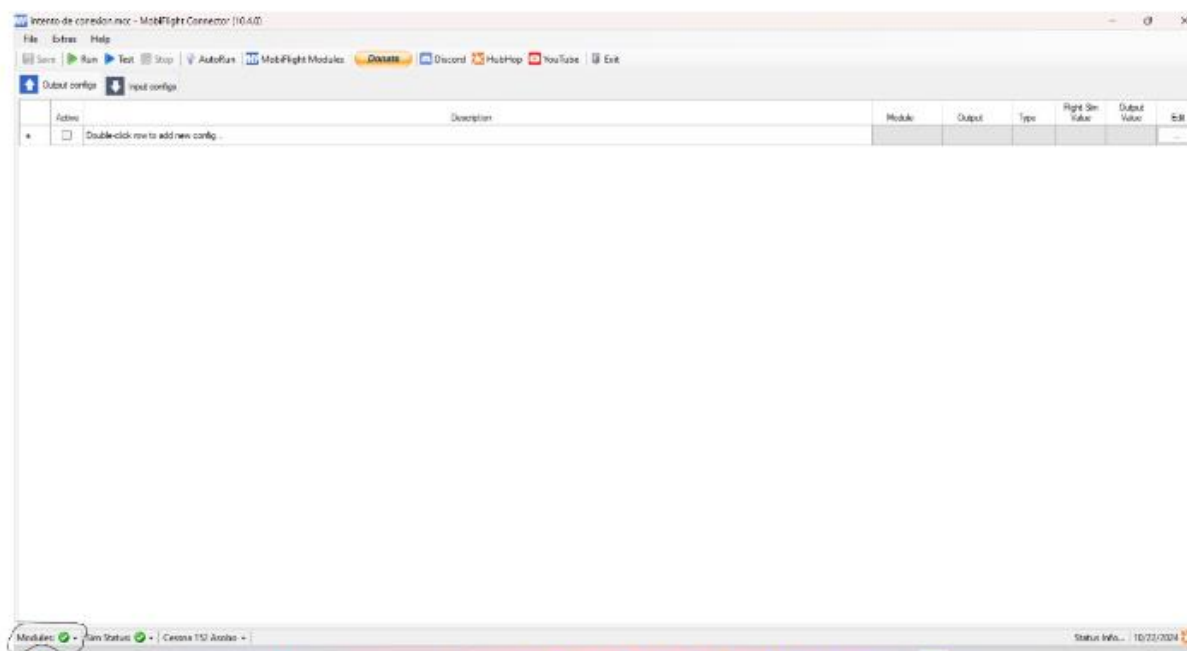
Para que el MobiFlight se vincule con el hardware del simulador de vuelo, nos tiene que aparecer en la barra inferior una tilde que confirme si se vincularon los siguientes parámetros:

Module:

- Si no hay ningún microcontrolador conectado al MobiFlight aparecerá un triángulo amarillo indicando eso



- Caso contrario, si está conectado al MobiFlight algún Microprocesador, (siendo el Arduino UNO en nuestro caso) se indicará que MobiFlight lo detectó correctamente.



Sim Status

- El Sim Status indica el estado de las conexiones al simulador de vuelo. En nuestro caso usamos el programa "FSUIPC7" para que el FS2020 detecte el hardware. Así nos tiene que aparecer:

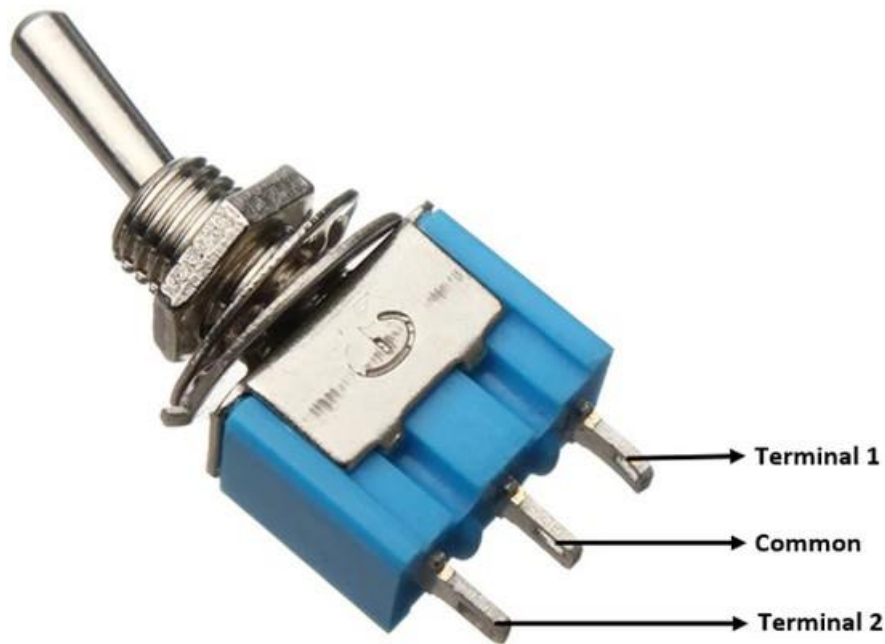


Configuración de las Palancas en el MobiFlight

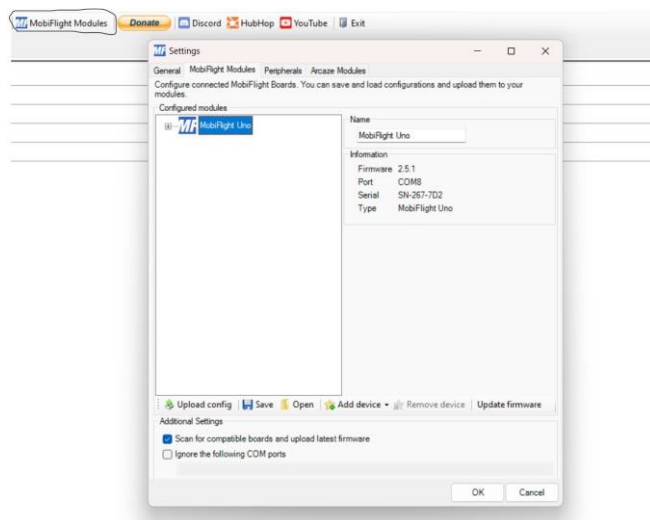
Paso a Paso de configuración de palancas para vincularlas con MobiFlight y el FS2020

1. Conectamos el Arduino UNO al MobiFlight y el FSUIPC7

2. Asignamos un Pin del 2 al 13 del Arduino UNO a uno de los terminales de la palanca (En nuestro caso lo pusimos al pin 8 pero puede ser cualquiera)



3. Seleccionar en MobiFlight la pestaña de "MobiFlight Modules"



4. Para no tener inconvenientes actualizar el firmware así el MobiFlight detecte correctamente el la Arduino UNO



5. En la barra inferior ir a la pestaña de "Add Device" para agregar un dispositivo que en nuestro caso es una Palanca.

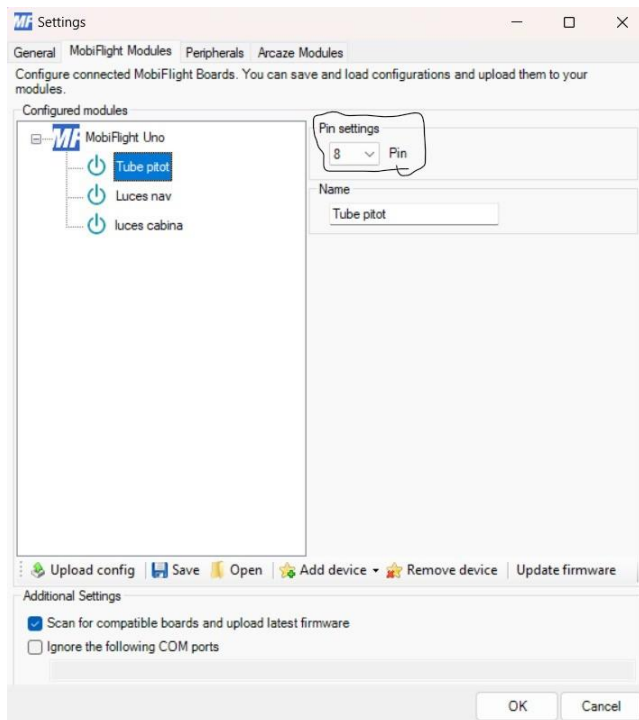


6. Luego le asignamos a la palanca el nombre con el que queramos que se identifique, siendo en este caso "Tubo Pitot" para la función de la palanca que es sobre el Calentador del Tubo Pitot

Name

Tube pitot

7. Poner en la "Configuración de Pin" el pin que le asignamos a la palanca (en nuestro caso el Pin 8 de la Arduino UNO)

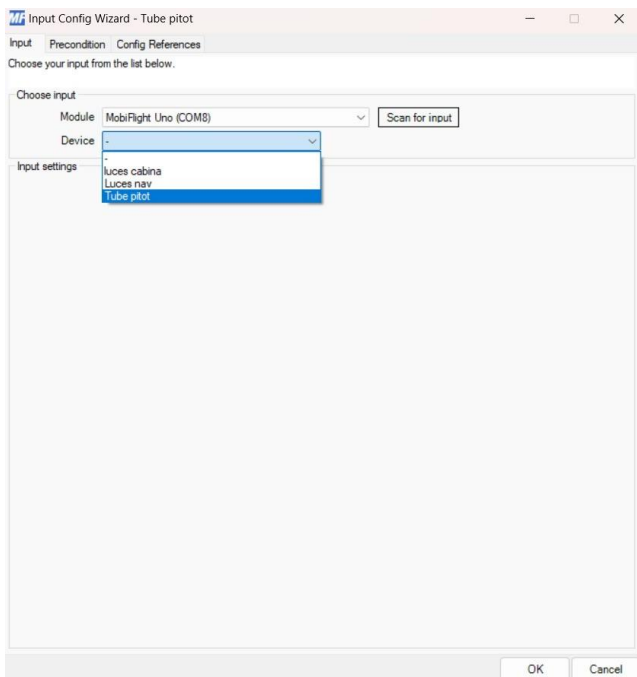


8. En el Menú principal crear una configuración de entrada en el "Input Configs"

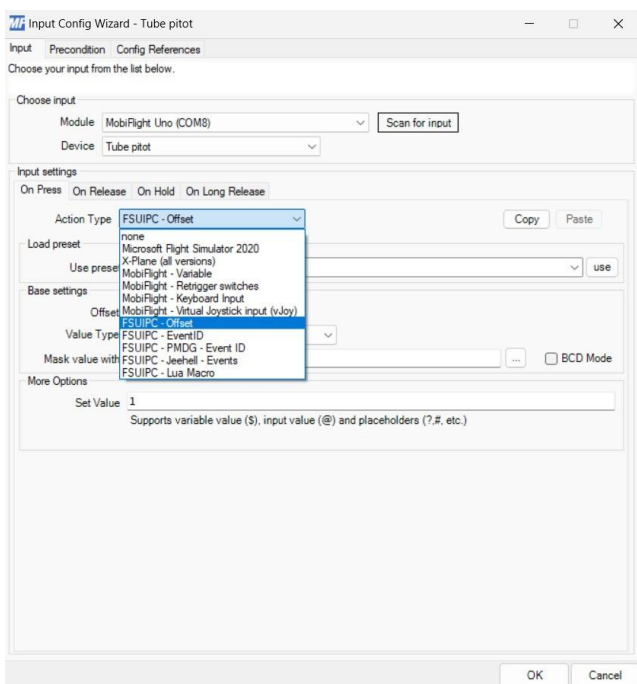
Output configs		Input configs			
Active	Description	Module	Input	Type	Edit
<input checked="" type="checkbox"/>	Tube pitot	MobiFlight Uno	Tube pitot	Button	...
<input checked="" type="checkbox"/>	Luces nav	MobiFlight Uno	Luces nav	Button	...
<input checked="" type="checkbox"/>	Luces cabina	MobiFlight Uno	Luces cabina	Button	...

9. Para editar la configuración selecciona a la derecha de la fila en "Edit"

10. Luego seleccionar los dispositivos conectados al MobiFlight

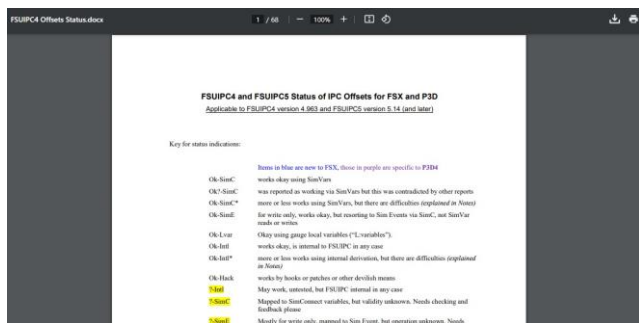


11. Seleccionar en el Action Type el "FSUIPC - Offset" para luego asignarle la acción al dispositivo que hayamos conectado (en nuestro caso la palanca)



12. Dentro de la configuración de entrada se encuentran las configuraciones para cuando se activa la palanca "On Press" y cuando está del lado contrario "On Release"

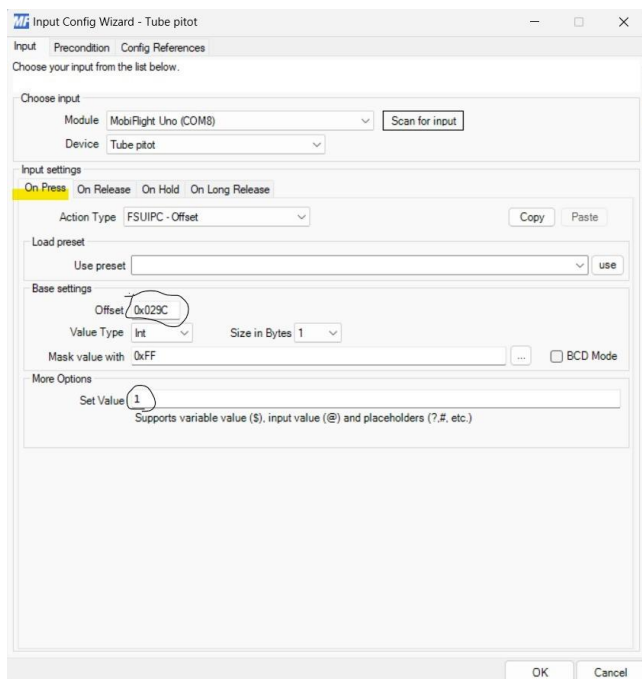
13. Para la configuración del Offset nos fijamos en el manual de Offset del FSUIPC que está en internet para buscar la acción de la palanca, y nos dice cuál código de Offset tiene que ir.



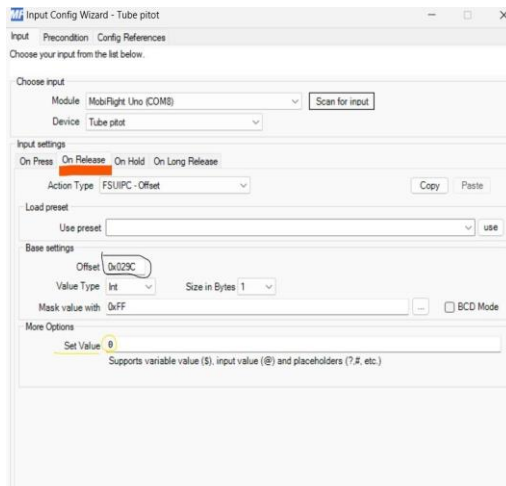
14. En nuestro caso buscamos el del calentador del tubo pitot. Por lo que ponemos el que nos dice "029C"

029B	1	Alternate static air source (V=off, I=on)	029C	029D
029C	1	Pitot Heat switch (0=off, 1=on)	Ok-SimC	Ok-SimE

15. Para que el Calentador del tubo pitot del Cessna 152 se active en simultáneo a la palanca que conectamos al FS2020, le asignamos "1" como valor de seteo.



16. En "On Release" de igual forma poner el mismo código de Offset pero que cuando se baje la palanca que el calentador se apague ponemos "0" como valor de seteo



17) Luego de esto al probar la palanca que vinculamos a dispositivo que sea (en nuestro caso el calentador del tubo pitot), se tendr

Estructura:

Estos componentes irán colocados en una madera rígida pero liviana, la cuál ha sido medido con el fin de mejorar la estética, y esta medida sería de 500mmx140/120mm. Ya cortadas las medidas se colocarán los interruptores y demás componentes. Luego en la parte final se pintará la madera con un aerosol negro y se pondrán letras en blanco que formen los nombres de cada interruptor, tal como en una cabina real del CESSNA 172 para dar más realismo.

Panel de Visualización de Instrumentos

Con el fin de generar una experiencia realista de vuelo, implementamos un Panel de Instrumentos virtual, el cuál por medio de una pantalla de 17 pulgadas transmitiremos los datos reales de cada instrumento de vuelo. Por lo que para transmitir esos datos o también llamados “Variables”, se encargan los programadores del equipo: Santiago Rubio y Lucas Meabrio.

Para darle una mejor estética al panel de instrumentos y que no sea solo una televisión mostrando los instrumentos, decidimos usar una placa de plástico con los agujeros de los instrumentos del avión usada por el grupo anterior del simulador (AVIS), la cuál para la fecha de la exposición será forrada para mejorar la estética del panel en conjunto a el otro panel.

Una parte importante del Panel de Instrumentos es toda la labor de *Personalización de la ubicación de los instrumentos en la pantalla*. Nosotros elegimos usar una aplicación creada por la comunidad de FS2020 llamada “Pop Out Panel Manager”, para así que se muestre de la forma que queramos en la pantalla de 17 pulgadas los instrumentos de vuelo.

MATERIALES:

- **1 Madera para el panel de control**
- **1 Placa de plástico con los agujeros para cada instrumento**
- **2 Arduino UNO**
- **40 cables macho-macho aprox.**
- **1 llave selectora rotativa de 6 estados**
- **7 interruptores de palanca**
- **8 pulsadores - PUL-ST030-N**
- **4 interruptores/switches**
- **1 Televisión de 18 pulgadas**
- **1 Aerosol de color negro/balde de pintura color negro**
- **1 Pack de letras transferibles color blanco**
- **2 SIMPLE FILA CONTACTO DOBLE PASO - PB20S**

Gastos:

- **40 cables macho-macho aprox: 5000\$**
- **1 llave selectora rotativa de 6 estados: 3800\$**
- **8 interruptores de palanca: 9714\$ (1214\$ c/u)**
- **4 interruptores/switches: 4000\$**
- **1 Pack de letras transferibles color blanco: 11400\$**
- **2 filas de contacto doble paso: 1435\$ (717\$ c/u)**

TOTAL: 35.340\$

Anexos