xx

aAL

Versión 1.0, ENERO 2021

# índice

# objetivo

El objeto de este documento es describir un robot autónomo explorador que se denominará a lo largo de todo el documento **iROB-EA**.

## glosario de términos

* **iROB-EA. i**nternet **ROB**obot **E**xplorador **A**utónomo. Robot descrito en el presente documento.
* **UF\_xxx**. **U**nidad **F**uncional.

El término concreto será del tipo **UF\_xxx** dónde “xxx” serán tres caracteres alfanuméricos que identifican la Unidad Funcional.

* **SS\_xxx**. **S**ub **S**istema.

El término concreto será del tipo **SS\_xxx** dónde “xxx” serán tres caracteres alfanuméricos que identifican el Subsistema.

# introducción

Se propone el diseño de un robot autónomo cuya misión será explorar y recabar información del entorno que le rodea.

A lo largo del presente documento se aportará toda la información necesaria para comprender el funcionamiento del iROB-EA así como para su construcción a nivel hardware, en tanto que la información relacionada con el software necesario para su correcto funcionamiento se encuentra fuera de este documento, formando parte de la documentación general del proyecto.

Así mismo, si bien este documento recoge la documentación de detalle del hardware del iROB-EA la construcción “final” del mismo queda a la elección del lector, por lo que no se darán indicaciones de cómo se debe construir una parte concreta u otra, si el lector decide por ejemplo utilizar un chasis de otro tipo quedará pues a su elección.

# especificaciones

Se definen las siguientes especificaciones para el iROB-EA.

# arquitectura

En este capítulo se recoge la arquitectura del iROB-EA.

La Arquitectura del iROB-EA se estructura en base a xx Unidades Funcionales (UFs).

Cada una de estas unidades está a su vez formada por varios Subsistemas que tienen relación directa entre ellos, estos subsistemas en conjunto proveen la funcionalidad que la UF soporta.

Un subsistema debe ser entendido como un circuito o conjunto de ellos que operan bien de forma autónoma o controlados por una aplicación o software.

Algunos subsistemas son simples circuitos mientras que otros pueden tener más entidad.

La siguiente ilustración muestra las distintas UFs que conforman la arquitectura y su relación entre ellas.

Ilustración . Arquitectura iROB-EA.

Como se puede apreciar existen xx UFs claramente diferenciadas cuya misión se describe a continuación.

## UF\_SYS

**U**nidad **F**uncional de **SYS**tema.

Esta UF es la encargada de proveer los siguientes servicios:

* Gestión de sensores.
* Gestión de actuadores.
* Interfaz de depuración (debug).
* Gestión de alimentación subsistemas.
* Implementación del mecanismo de reacciones básicas.

Esta UF implementa el “BIOS” del iROB-EA soportando todas las funciones de bajo nivel tales como gestión de sensores y actuadores, obteniendo de los mismos la información que facilitan y poniéndola a disposición de la UF\_APL que es la encargada de ejecutar los programas y aplicaciones de alto nivel del iROB-EA.

Una tarea importante llevada a cabo en esta UF es la de soportar un mecanismo denominado de *reacciones básicas*, por el cual la UF\_SYS responde de forma automática ante determinadas situaciones reconocidas en base a la información obtenida de los diferentes sensores. Este mecanismo funciona de forma similar al que provoca los *actos reflejos* en los seres vivos.

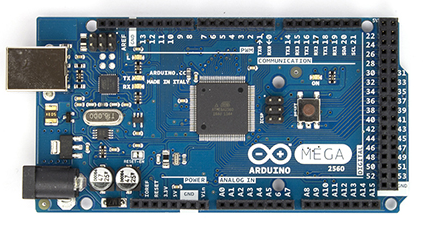
Existen situaciones de alta prioridad ante las que es preciso actuar de forma inmediata dado que son potencialmente peligrosas para la integridad del iROB-EA, ante cualquiera de estas situaciones la UF\_SYS procede a asegurar la integridad del iROB-EA para posteriormente comunicar la aparición de esta situación a la UF\_APL.

La UF\_SYS está formada por un único subsistema:

* SS\_SYS. Subsistema principal de la UF\_SYS.

### SS\_SYS

Este subsistema es el único componente de la UF\_SYS y está implementado entorno a un Arduino MEGA2560,



El modelo MEGA2560 posee entre otras, las siguientes características,

|  |  |
| --- | --- |
| Microcontrolador | ATmega2560 |
| Tensión de operación | 5V |
| Digital I/O Pins | 54 (15 de ellos soportan salida PWM) |
| Analógicos I/O Pins | 16 |
| Corriente DC por I/O | 40 mA |
| Memoria Flash | 256 KB (8 KB usados por el bootloader) |
| SRAM | 8 KB |
| EEPROM | 4 KB |
| Clock | 16 MHz |

Para más información consultar la web del fabricante,

<http://arduino.cc/en/Main/ArduinoBoardMega2560>

Así mismo se encuentra disponible en numerosos sitios online, entre otros,

<http://tienda.bricogeek.com/arduino/306-arduino-mega-2560.html>

<https://www.sparkfun.com/products/11061>

<http://www.cooking-hacks.com/arduino-mega-2560>

Como se ha comentado, este subsistema es el encargado de gestionar todos los sensores y actuadores del iROB-EA.

Este subsistema se comunica utilizando los puertos serie que incorpora el ArduinoMEGA2560 manteniendo conexión con los siguientes elementos:

* **Exterior.**

Puerto de comunicación serie RS-232 con el exterior para tareas de depuración y test, este puerto está presente en el conector exterior de DEBUG, soportado a través del SS\_DBG de la unidad funcional UF\_ENE, que convierte los niveles de Transmisión/Recepción a RS-232.

* **UF\_APL.**

Puerto de comunicación serie con la UF\_APL que es la unidad funcional encargada de ejecutar las aplicaciones que proveen de funcionalidad al iROB-EA.

## UF\_APL

**U**nidad **F**uncional de **APL**icaciones.

Esta UF es la encargada de proveer los siguientes servicios:

* xx
* xx
* xx

La UF\_APL está formada por los siguientes subsistemas:

* xx
* xx
* xx

## UF\_ENE

**U**nidad **F**uncional de **ENE**rgía.

Esta UF es la encargada de proveer los siguientes servicios:

* Encendido y Apagado general del iROB-EA.
* Recarga de baterías y obtención de información acerca del estado general de las mismas.
* Encendido y apagado de elementos y subsistemas.
* Alimentación y comunicaciones en modo pruebas (DEBUG).
* Watchdog.

La UF\_ENE está formada por los siguientes subsistemas:

* SS\_POW. Este subsistema es el encargado del encendido y apagado (power On/Off) del iROB-EA.
* SS\_BAT. Este subsistema permite generar las tensiones de alimentación, recargar las baterías y operaciones asociadas con las mismas.
* SS\_CNX. Subsistema que permite activar/desactivar la alimentación suministrada a distintos elementos del iROB-EA.
* SS\_DBG. Subsistema encargado de soportar la depuración exterior
* SS\_DOG. Subsistema WatchDog.

### SS\_POW

Este subsistema es el encargado del encendido y apagado del iROB-EA.

#### Procedimiento de encendido (Power-ON).

El iROB-EA solo puede ser encendido utilizando el pulsador “power ON/OFF”, al hacerlo se iniciará el proceso de encendido (siempre que las baterías tengan energía suficiente).

Una vez presionado el pulsador de encendido y tras inicializarse correctamente el Arduino MEGA (algunos segundos) el display multifunción mostrará un mensaje indicando “espera de código de inicio” en ese momento se deberá generar el código de 4 dígitos necesario para la activación efectiva del iROB-EA.

Este código está formado por una secuencia de 4 dígitos que se envía al iROB-EA en forma de tonos multifrecuencia (DTMF) que pueden ser generados desde cualquier teclado de un teléfono convencional o dispositivo generador de tonos multifrecuencia DTMF compatible, para ello bastará acercar a cualquiera de los micrófonos del iROB-EA el elemento emisor y generar los tonos correspondientes a la clave de encendido.

Se dispone de xx segundos para enviar los 4 tonos, pasado ese tiempo el iROB-EA se apagará automáticamente.

Si la secuencia es correcta, el iROB-EA pasará al modo activo, en caso contrario se dispone de dos intentos más, agotados los cuales el iROB-EA pasará al modo de protección contra inicios incorrectos bloqueándose de forma permanente.

**Importante:**

*Llegados a este punto, si se produce el bloqueo del iROB-EA, será necesario modificar en el código del programa fuente que ejecuta el Arduino MEGA, la constante que define la clave de arranque, definiendo otra distinta a la que exista, realizar una nueva compilación y cargar la nueva versión en el Arduino MEGA, esto conllevará entre otras cosas estar en disposición de los ficheros fuente con el código que ejecuta el Arduino MEGA para poder efectuar los cambios y generar una nueva versión.*

Si la secuencia introducida es correcta el display multifunción mostrará un mensaje indicativo y el Arduino MEGA completará la secuencia de encendido, pasando al modo operativo.

#### Procedimiento de apagado (Power-OFF).

Para el apagado del iROB-EA existen dos posibilidades:

* Apagado “físico”.

Esto es, utilizando de nuevo el pulsador “power ON/OFF”, se deberá mantener pulsado el mismo durante al menos xx segundos para confirmar el apagado.

* Apagado “remoto”.

Existe la posibilidad de apagar el iROB-EA en modo remoto, desde la interfaz web de control.

Hay que tener en cuenta que si se utiliza esta modalidad de apagado, el iROB-EA quedará totalmente fuera de servicio, siendo preciso el encendido del mismo utilizando de nuevo el pulsador “power ON/OFF”.

En ambos casos el display multifunción mostrará un mensajeindicando que se está realizando el proceso de apagado. Las operaciones de apagado son realizadas por el Arduino MEGA y conllevan el apagado “ordenado” de los diferentes módulos, así por ejemplo la UF\_APL no puede ser desconectada “sin más” puesto que en caso de estar activa es preciso que se cierren todas las aplicaciones que estuvieran ejecutándose antes de proceder a su desconexión física. Finalmente se desconectará la alimentación general de todo el sistema.

### SS\_BAT

Este subsistema permite generar las tensiones de alimentación de los diferentes elementos, recargar las baterías y obtener información relacionada con las baterías.

En el diseño del iROB-EA se ha considerado prioritario unificar las distintas tensiones a utilizar en los circuitos de cara a reducir la complejidad de los circuitos necesarios para generarlas, así como optimizar el espacio ocupado por elementos como baterías, reguladores etc.

El resultado es la utilización de tres únicas tensiones:

### SS\_CNX

Este subsistema permite activar/desactivar la alimentación suministrada a distintos elementos del iROB-EA.

Algunos elementos del iROB-EA pueden ser alimentados o no, dependiendo si son necesarios, esto permite gestionar y el consumo de los mismos, optimizando así el consumo general del sistema.

Un ejemplo típico de esta circunstancia es la cámara IP, efectivamente, se trata de un elemento que presenta un consumo significativo y que no es necesario tener alimentado permanentemente.

Los elementos que pueden activarse/desactivarse desde este subsistema son:

* Cámara de vídeo.
* Módulo de GPS.
* Luces.
* Sensores de consumo elevado.
* Raspberry (Máquina Linux).

Estos elementos se conectan /desconectan siempre desde el Arduino que es el sistema que determina qué elementos deben estar funcionando en todo momento.

La conexión/desconexión de los diferentes elementos se realiza utilizando relés de enclavamiento, este tipo de relés tienen como principal ventaja la de no precisar alimentación permanente mientras están activados, por el contrario precisan de dos señales para su maniobra, (una para abrir los contactos y otra para cerrarlos) dado que en el Robot uno de los requisitos más importantes es el del consumo, se ha optado por implementar este subsistema con este tipo de relés.

### SS\_DBG

Subsistema de depuración del iROB-EA (Modo DEBUG).

El iROB-EA posee un conector exterior que permite realizar tareas de comprobación y depuración, este conector permite además alimentar el iROB-EA exteriormente lo cual es especialmente útil cuando se realizan tareas de mantenimiento, supervisión, pruebas etc dado que evita la utilización de las baterías en estos momentos.

Este subsistema “desconecta” automáticamente las baterías y alimenta todos los circuitos directamente desde las alimentaciones que se suministran a través del conector.

Para efectuar la comunicación con el exterior, el SS\_DBG comunica el Arduino MEGA con el exterior mediante uno de sus puertos serie, los niveles lógicos de este puerto (TX, RX) son enviados y recibidos del exterior a través de un driver RS-232 que hace que la comunicación se más robusta y fiable, pudiendo alcanzar mayores distancias que con los niveles lógicos TTL.

### SS\_DOG.

Este subsistema implementa dos servicios:

* Master Reset.

Señal de reset maestro, se utiliza para inicializar el Arduino y aquellos elementos que precisen ser inicializados en el momento del arranque del iROB-EA.

* WatchDog.

El término *WatchDog* hace referencia a un circuito que de forma independiente es capaz de generar un pulso cada cierto tiempo si antes no recibe una señal de control.

El uso típico de este tipo de circuitos es el de “resetear” un sistema cuando este ha perdido el control y no responde (sistemas basados en software, microcontroladores, ordenadores etc).

Es importante destacar que el Arduino MEGA tiene implementado un sistema de watchdog pero de la documentación existente se desprende que es preciso modificar el *bootloader* que incorpora de fábrica para habilitar este watchdog, además de las fuentes consultadas se desprende que en determinados casos podría producirse algún tipo de funcionamiento no deseado por lo que se ha optado por implementar un sistema de watchdog externo al Arduino MEGA.

El SS\_DOG generará un pulso de Master Reset automáticamente si no recibe una señal de control periódica generada en este caso desde el Arduino MEGA.

La generación del pulso de Master Reset es configurable mediante unos puentes, pudiendo elegirse entre 1 y 64 segundos.

## UF\_MOT

**U**nidad **F**uncional de **MOT**ores.

El iROB-EA cuenta con dos motores de CC que son los encargados de relazar el movimiento, cada uno de ellos está asociado a cada una las orugas, derecha e izquierda.

Mediante el accionamiento de estos motores es posible realizar los movimientos necesarios para desplazar / girar el iROB-EA.

Esta UF es la encargada de proveer los siguientes servicios:

* Movimiento del iROB-EA.
* Monitorización de los motores de propulsión.

La UF\_MOT está formada por los siguientes subsistemas:

* SS\_MOT. Este subsistema es el encargado del control de los motores y realiza las tareas relacionadas con el movimiento del iROB-EA:
  + Avance
  + Retroceso
  + Giro derecha
  + Giro Izquierda
* SS\_MSE. Este subsistema permite monitorizar el estado de ambos motores.

### SS\_MOT

Este subsistema gira entorno a dos motores de CC que permiten realizar todos los movimientos del iROB-EA.

Cada uno de los motores está operado por un controlador independiente que recibe las señales de control del Arduino MEGA.

### SS\_MSE

Como se ha indicado al describir el SS\_MOT, cada uno de los motores está operado por un controlador independiente, estos controladores además proporcionan al sistema de control una serie de señales que permite monitorizar el estado del motor controlado.

Adicionalmente se ha añadido a cada uno de los controladores un sensor de corriente que permite monitorizar el consumo de cada motor. La monitorización de la corriente consumida por cada motor es especialmente importante dado que permite detectar averías y bloqueos del motor durante su funcionamiento.

# diseño detallado

Este capítulo recoge la información precisa para la construcción del iROB-EA.

## consideraciones previas

Antes de abordar en detalle la descripción de los diferentes subsistemas es necesario hacer las siguientes consideraciones:

**Para la construcción de los circuitos** se ha optado por la técnica de *wire-wrap*, por lo que no se ha recurrido a la fabricación de circuitos impresos.

Esta técnica es ideal para prototipado y dado que el Robot descrito en este documento lo es, se ha considerado una solución ideal para la construcción del mismo.

Par más información sobre esta técnica de construcción se puede consultar,

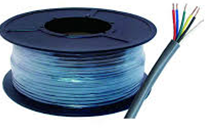
<http://es.wikipedia.org/wiki/Wire-wrap>

La técnica de wire-wrap aporta entre otras las siguientes ventajas frente a la utilización de circuitos impresos “clásicos”:

* Facilidad y mayor flexibilidad para introducir cambios en el diseño, no siendo necesariamente precisa la fabricación de nuevos circuitos impresos, ante modificaciones en el diseño.
* Robustez y fiabilidad, permitiendo aprovechar al máximo el espacio en circuitos complejos (uso de buses y conexiones numerosas).

Por el contrario la técnica de wrapping requiere:

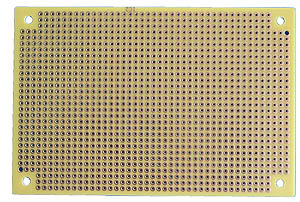
* Herramientas especiales (a menudo caras), necesarias para realizar las conexiones y la utilización de hilo específico (AWG20) para efectuarlas.
* Componentes adicionales como hileras de pines de diversos tipos para efectuar el wrapping de los hilos.
* Utilización de tarjetas de prototipado multi taladradas.











Estas desventajas se traducen finalmente en un coste final más elevado, si bien la ventaja de poder efectuar cambios en el diseño suple con creces la mayor inversión en la implementación.

## uf\_ene

**U**nidad **F**uncional de **ENE**rgía.

Seguidamente se describen en detalle los subsistemas que forman esta UF.

### ss\_pow

### ss\_cnx

### ss\_bat

### ss\_dbg

### ss\_dog

## UF\_MOT

**U**nidad **F**uncional de **MOT**ores.

Seguidamente se describen en detalle los subsistemas que forman esta UF.

### SS\_MOT

### SS\_MSE

# descripción de conectores

Este capítulo detalla los diferentes conectores que conexionan los distintos circuitos del iROB-EA.

* **Conector CN01**

Monitorización de carga de baterías y Power ON/OFF.

|  |  |  |
| --- | --- | --- |
| **CN\_01** | **TIPO:** |  |
| **PIN** | **NOMBRE** | **DESCRIPCIÓN** |
| 1 | GND | GND |
| 2 | BAT\_N100 | Test 100% Carga |
| 3 | BAT\_N75 | Test 75% Carga |
| 4 | BAT\_N50 | Test 50% Carga |
| 5 | BAT\_25 | Test 25% Carga |
| 6 | CTRL\_ON\_OFF | Pulsador de encendido de la Batería |

|  |  |  |
| --- | --- | --- |
| **CN\_02** | **TIPO:** |  |
| **PIN** | **NOMBRE** | **DESCRIPCIÓN** |
| 1 | GND | GND |
| 2 | +3.3VA­ |  |
| 3 | +5VA |  |
| 4 | POWER\_CNX\_A0 |  |
| 5 | POWER\_CNX\_A1 |  |
| 6 | POWER\_CNX\_A2 |  |
| 7 | -- |  |
| 8 | POWER\_OFF\_PETICION |  |
| 9 | ICC\_SENSE\_5VP |  |
| 10 | ICC\_SENSE\_12P |  |

|  |  |  |
| --- | --- | --- |
| **CN\_03** | **TIPO:** |  |
| **PIN** | **NOMBRE** | **DESCRIPCIÓN** |
| 1 | +12V\_DEBUG | +12V, Alimentación para pruebas |
| 2 | GND\_DEBUG | GND, Alimentación para pruebas |
| 3 | +5V\_DEBUG | +5V, Alimentación para pruebas |

|  |  |  |
| --- | --- | --- |
| **CN\_04** | **TIPO:** |  |
| **PIN** | **NOMBRE** | **DESCRIPCIÓN** |
| 1 | OUT12V\_VCC | +12V, salida de la Bateria |
| 2 | OUT5V\_GND | GND, salida de la Bateria |
| 3 | OUT5V\_VCC | +5V, salida de la Bateria |

|  |  |  |
| --- | --- | --- |
| **CN\_05** | **TIPO:** |  |
| **PIN** | **NOMBRE** | **DESCRIPCIÓN** |
| 1 | GND | GND |
| 2 | +12P | +12V, Alimentación motores |

|  |  |  |
| --- | --- | --- |
| **CN\_06** | **TIPO:** |  |
| **PIN** | **NOMBRE** | **DESCRIPCIÓN** |
| 1 | GND | GND |
| 2 | +5VP | +5V, Alimentación PC |

|  |  |  |
| --- | --- | --- |
| **CN\_07** | **TIPO:** |  |
| **PIN** | **NOMBRE** | **DESCRIPCIÓN** |
| 1 | GND | GND |
| 2 | POWER ON/OFF | Pulsador ON/OFF |
| 3 | CTRL\_ON\_OFF | Pulsador de encendido de la Batería |

|  |  |  |
| --- | --- | --- |
| **CN\_08** | **TIPO:** |  |
| **PIN** | **NOMBRE** | **DESCRIPCIÓN** |
| 1 | +5V | +5V, Alimentación General, Robot |
| 2 | GND | GND |
| 3 | +12V | +12V, Alimentación General, Robot |

# anexo

## tabla de asignación de pines en arduino mega2560.

Resumen de asignación de pines en Arduino MEGA.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Asignación de Pines I/O** | | | | | |
| **FUNCIONALIDAD** | **PINES I/O** | **TIPO** | **ASIGNACIÓN** | | |
| **PIN** | **NOMBRE** | **MODO** |
| **Leds rojos traseros** | 1 | Digital: 1 | D37 | LED RO | O |
| **Leds blancos traseros** | 1 | Digital: 1 | D38 | LED BA DETRAS | O |
| **Leds blancos delanteros** | 1 | Digital: 1 | D39 | LED BA DELANTE | O |
| **Watchdog, Flag** | 1 | Digital: 1 | D12 | DOG\_SFLAG | I |
| **Watchdog, Done** | 1 | Digital: 1 | D53 | DOG\_DONE | O |
| **Señal de power OFF** | 1 | Digital: 1 | D02 | OFF\_PETICION | I  (**Interrupción 0 del Arduino MEGA2560**) |
| **Display multifunción** | 5 | Digital: 5 | D44  D41  D43  D42  D40 | DATA IN  RS  CLK  CE  RESET | O  O  O  O  O |
| **Motor derecho** | 6 | Digital: 5  Analógico: 1 | D03  D35  D34  D33  D36  A3 | PWM\_DERECHO  RESET\_DERECHO  FF2\_DERECHO  FF1\_DERECHO  DIR\_DERECHO  SENSE\_DERECHO | O  O  I  I  O  I |
| **Motor izquierdo** | 6 | Digital: 5  Analógico: 1 | D04  D31  D30  D28  D32  A4 | PWM\_IZQUIERDO  RESET\_IZQUIERDO  FF2\_IZQUIERDO  FF1\_IZQUIERDO  DIR\_IZQUIERDO  SENSE\_IZQUIERDO | O  O  I  I  O  I |
|  | | | | | |
| Estación Metereológica | | | | | |
| **Sensor temperatura Melexis MLX90614** | I2C | I2C | I2C | I2C | I/O |
| **GPS MTK 3339, Pin Apagado/Encendido** | 1 | Digital: 1 | D26 | GPS\_ON\_OFF | O |
| **GPS MTK 3339** | TRX2 | Puerto serie | TX2  RX2 | GPS\_TX  GPS\_RX | O  I |
| **Estación Meteorológica** | I2C | I2C | I2C | I2C | I/O |
| **Estación Meteorológica ( Sensor luz )** | 1 | Analógico: 1 | A9 | SEN\_MET\_LUZ | I |
|  | | | | | |
| **Módulo 9DOF-Razor-v21 ( acelerómetro, giroscopio, magnetómetro)**  http://blog.bricogeek.com/noticias/tutoriales/imu-razor-9-dof-con-ahrs-y-conexion-usb/ | TRX3 | Puerto Serie | TX3  RX3 | RAZOR\_TX  RAZOR\_RX | O  I |
| **Módulo 9DOF-Razor-v21 ( acelerómetro, giroscopio, magnetómetro)**,  señal DTR | 1 | Digital: 1 | D24 | DTR\_RAZOR\_1 | O |
| **Control de relés, alimentación de subsistemas** | 3 | Digital: 3 | D47  D46  D45 | POW\_CNX\_A0  POW\_CNX\_A1  POW\_CNX\_A2 | O  O  O |
| **Decodificador tonos DTMF** | 5 | Digital: 5 | D52  D51  D50  D49  D48 | D3  D2  D1  D0  DV | I  I  I  I  I |
| **Detector infrarrojos** | 1 | Digital: 1 | D22 | IR01 | I |
| **RTC ( Reloj de Tiempo Real)** | I2C | I2C | I2C | I2C | I/O |
| **Indicador de conector DEBUG conectado** | 1 | Digital: 1 | D13 | DEBUG\_ACTIVO | I |
| **Sistema de control de consumo y recarga de baterías** | 7 | Digital: 4  Analógico: 3 | D08  D05  D06  D07  A8  A1  A2 | INBP  INB0  INB1  INB2  INBS  POW\_RECARGA  LiPo\_RECARGA | O  O  O  O  I  I  I |
| **Ultrasonidos derecho** | 2 | Digital:1  Analógico: 1 | D11  A5 | USR\_DERECHO\_S  USR\_DERECHO\_C | O  I |
| **Ultrasonidos central** | 2 | Digital:1  Analógico: 1 | D10  A6 | USR\_CENTRAL\_S  USR\_CENTRAL\_C | O  I |
| **Ultrasonidos izquierdo** | 2 | Digital: 1  Analógico: 1 | D09  A7 | USR\_IZQUIERDO\_S  USR\_IZQUIERDO\_C | O  I |
| **Micrófono DTMF** | 1 | Analógico: 1 | A0 | MIC | I |
| **Sistema de encendido/apagado Raspberry** |  |  |  |  |  |

## asignación de puertos serie en arduino mega2560.

|  |  |
| --- | --- |
| **FUNCIONALIDAD** | **PUERTO SERIE** |
| Comunicación Arduino MEGA2560 – UF\_APL  Programación Arduino MEGA2560 | **TX0** |
| Monitorización Arduino MEGA2560, accesible en el conector de DEBUG | **TX1** |
| GPS MTK 3339 | **TX2** |
| Módulo 9DOF-Razor-v21 ( acelerómetro, giroscopio, magnetómetro) | **TX3** |

## identificación de relés

**Verde**

RE1, Relé de encendido.

**Amarillo**

RE2, Relé +5Vcc\_Raspberry.

**Azul**

RE3, Relé +5Vcc\_Camara.

**Rojo**

RE4, Relé +7V4\_MOTORES

**Verde**

Re1, Relé DEBUG, +7V4\_VCC

**Amarillo**

Re2, Relé DEBUG +5\_1AVcc

**Rojo**

Re2, Relé DEBUG +5\_2AVcc

## identificación de FUSIBLES