

# Arquitectura

REMOTE-LAB-LIB

Interfaz FrontEnd-Arduino para laboratorio remoto

UNIVERSIDAD DE LA MARINA MERCANTE, CABA, BUENOS AIRES ARGENTINA

[leandro.cintioli@alumnos.udemm.edu.ar](mailto:leandro.cintioli@alumnos.udemm.edu.ar)  
[pablo.tavolaro@alumnos.udemm.edu.ar](mailto:pablo.tavolaro@alumnos.udemm.edu.ar)

## **Alumnos:**

Leandro Martin Cintioli

Pablo Oscar Tavolaro Ortiz

Abril 2022

## **Práctica profesional supervisada.**

Tutor interno: **Ing. Daniel Alejandro Ramondegui.**

Tutor externo: **Ing. Marcelo Bellotti.**

**Universidad de la Marina Mercante**

**Facultad de Ingeniería**

<b>Objetivo</b>	<b>3</b>
<b>Antecedentes</b>	<b>3</b>
<b>Arquitectura</b>	<b>3</b>
<b>Flujo de ejecución</b>	<b>4</b>
<b>Clase log.cpp</b>	<b>4</b>
<b>Clase memory</b>	<b>5</b>
Eeprom	5
Estructura	5
Mapeo	6
Consistencia	6
Procesamiento de Json	6
<b>Ejemplos de uso</b>	<b>8</b>
<b>Ejecución del experimento</b>	<b>8</b>
<b>Clase log</b>	<b>8</b>
Impresión de string	9
Impresión para excel / arduino plotter	9
Sensado de señales	9
<b>Clase memory</b>	<b>10</b>
Métodos getter y setter.	10
<b>Anexo A. Usos de RemoteLabLib</b>	<b>12</b>
1 Remote Lab Lib uso de soporte para impresión por puerto serie	12
2 Demostración del soporte para Arduino Plotter	12
3 Demostración del soporte para modos de ejecución.	12
4 Demostración del uso del acceso a memoria no volátil del Arduino	12
<b>Anexo B. Intención del sistema de RemoteLab</b>	<b>13</b>
5 Demostración del sistema de desarrollo de experimentos con Arduino controlados por una interfaz-web	13

# Objetivo

Describir la arquitectura de la librería Remote-lab-lib y el uso de las funciones para el usuario.

## Antecedentes

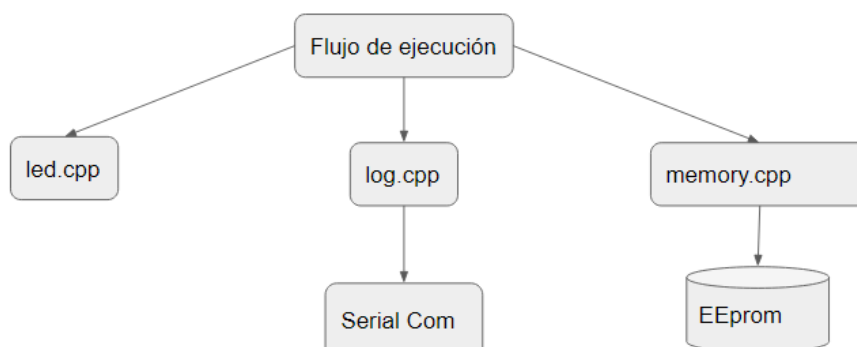
La intención del proyecto es suministrar una interfaz entre una página web y una plataforma Arduino, que acepta los comandos para empezar y terminar el experimento, y almacena los valores de configuración en la memoria no volátil de la placa.

Los módulos principales de son tres: front-end, back-end y librerías Arduino.

Ver Anexo.B

## Arquitectura

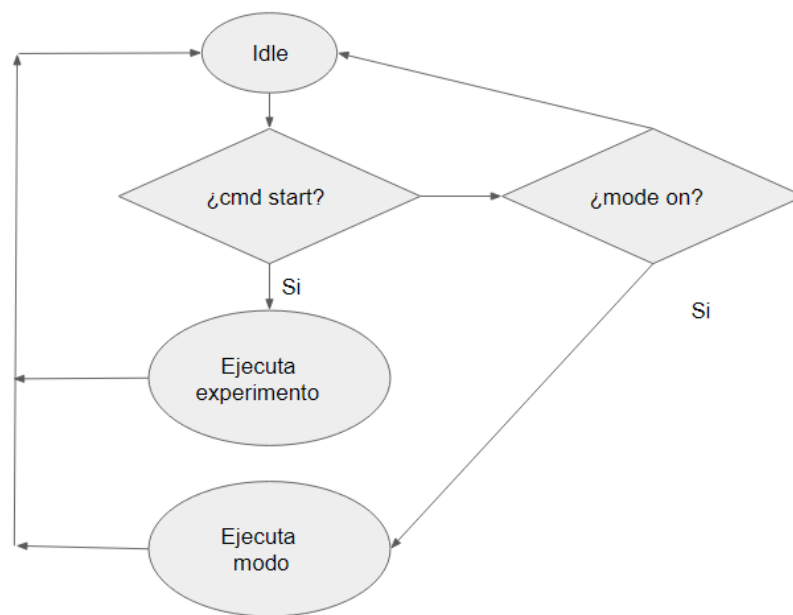
La arquitectura se basa en un loop que controla el flujo de ejecución del programa basado en los estados del sistema y los comandos recibidos. En general, por cada tarea existe una clase que aísla el comportamiento y controla los detalles del módulo.



## Flujo de ejecución

En reposo (Idle) el loop permanece esperando el ingreso de un comando por puerto serie en formato JSON.

Cuando recibe el comando START, da comienzo a la ejecución del experimento, y si recibe el comando de cambio de modo configurado, lo ejecuta.



## Clase Log

Esta clase contiene:

Soporte para impresión de texto por el puerto serie.

Soporte para impresión de Json por puerto serie.

Soporte para impresión para Arduino Plotter.

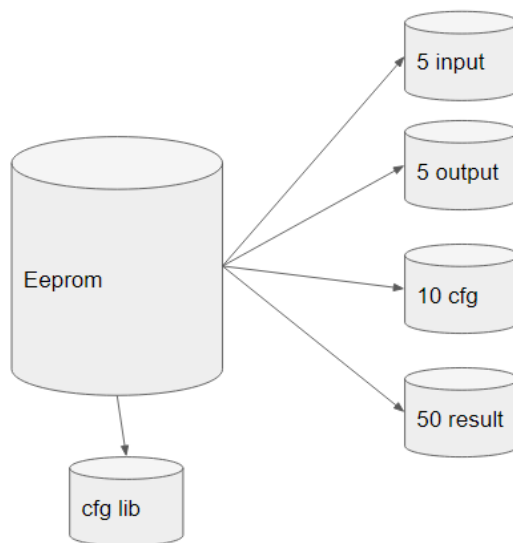
## Clase Memory

Esta clase se encarga de procesar el formato Json y de almacenar la configuración en la memoria Eeprom de Arduino.

### Eeprom

#### Estructura

La memoria tiene la siguiente estructura.



Todos los datos son del tipo float, excluyendo los cfg lib que son del tipo unsigned int 8

## Mapeo

La memoria está mapeada de la siguiente manera:

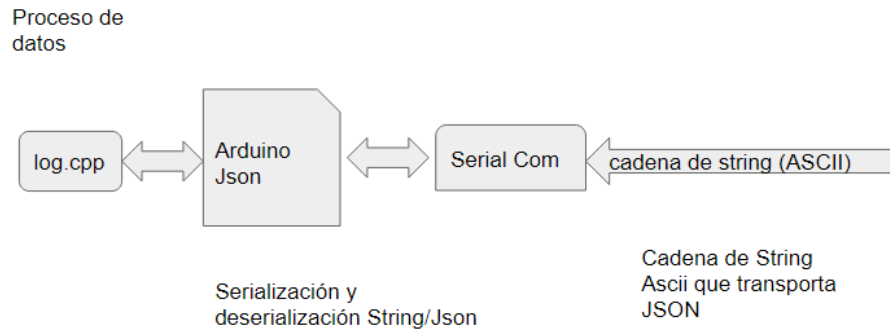
pos 0	magic_number
	st_test
	st_mode
	serial level
	input 0 - 4
	output 0 - 4
	cfg 0 - 9
	result 0 - 49

## Consistencia

Para evitar datos corruptos en la EEPROM y poder reordenar el mapeo en el caso de agregar datos, se utiliza el método MAGIC\_NUMBER, para que cuando en el inicio del programa el valor almacenado de MAGIC\_NUMBER no coincide, proceda a inicializar la memoria con los parámetros por defecto.

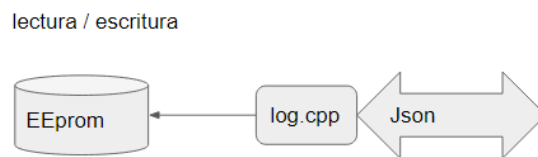
## Procesamiento de Json

La estructura de la comunicación es la siguiente



La librería Arduino Json es la encargada de manejar el formato Json entre el puerto serie y la clase log

La clase log se encarga de leer/escribir en la memoria según lo que se solicite por el puerto serie.



# Ejemplos de uso

## Ejecución del experimento

Al enviar el JSon {cmd:'start'} por el puerto serie se ejecuta , “ experiment(); “  
Acá el usuario escribirá su experimento.

```

/

/*
 * Aca el usuario escribe su experimento.
 */

static void experiment( void ){
    // Escribir experimento

}
}
```

## Clase log

Existe un parámetro serial\_level, éste define cuatro modos de uso de la impresión serie

LOG_DISABLED:	Desactivado
LOG_MSG:	Impresión de string activado
LOG_CTRL_JSON:	Impresión de excel activada.
LOG_CTRL_ARDUINO_PLOTTER:	Impresión para arduino plotter activada.

para configurar este parámetro se envía por el puerto serie el valor requerido

```
// {serial_level:'0'}    serial_level:0= Desactivado.
// {serial_level:'1'}    1= Impresión de string activado.
// {serial_level:'2'}    2= Impresión de excel activada..
// {serial_level:'3'}    3= Impresión para arduino plotter activada.
```



## Impresión de string

Con el siguiente método se puede imprimir texto y variables cuando la opción LOG\_MSG está activada

```
void msg( const __FlashStringHelper *fmt, ... );
```

uso:

Log.msg( F("Ejemplo de prueba texto")); **ver ANEXO A.1**

## Impresión para excel / arduino plotter

### Sensado de señales

En general cuando uno adquiere señales en crudo (raw), necesita filtrarlas y comparar el valor filtrado (filtered) contra un valor seteado (danger\_point) para definir un estado de salida (state).

En este escenario se provee soporte para la impresión de estos valores.

```
void ctrl( uint16_t raw, uint16_t filtered, uint8_t state, uint16_t danger_point );
```

Uso

```
/*
 * Experimento de usuario: Demo Arduino Plotter
 * este codigo se usa en el video: https://youtu.be/7ykRqe4GuWU
 */

static void experiment( void ){
    // Escribir experimento
    uint16_t raw, filtered, danger_point=2500;
    uint8_t state=0;

    Led.n_blink(5, 500);
    for (raw=0;raw<5000;raw++){
        filtered = raw+1 ;
        if (filtered >danger_point ){
            state=1 ;
        }
        Log.ctrl (raw, filtered,state,danger_point);
    }
}
```

**Ver ANEXO A.2**

## Clase memory

Métodos getter y setter.

Para que el usuario acceda a la memoria se provee una función por cada tipo de parámetro.

```
// Getter and setter's.  
  
float get_input(uint8_t param);  
void set_input(uint8_t param, float value);  
  
float get_output(uint8_t param);  
void set_output(uint8_t param, float value);  
  
float get_cfg(uint8_t param);  
void set_cfg(uint8_t param, float value);  
  
float get_result(uint8_t param);  
void set_result(uint8_t param, float value);|
```

Los nombres de los parámetros se puede definir a elección a través de #define en param.h

```
*/  
// Inputs.  
  
#define FUERZA    1  
#define PESO      2  
#define ENERGIA   3  
#define TENSION   4  
#define POTENCIA  5  
  
// Outputs.  
  
#define ACELERACION    10  
#define MILIMETROS     20  
#define AMPER          30  
#define NEWTON          40  
#define ANGULO         50  
  
// Configuracion  
  
#define INPUT0_ADD     11  
#define INPUT1_ADD     22  
#define INPUT2_ADD     33  
#define INPUT3_ADD     44  
#define INPUT4_ADD     55  
#define INPUT0_MUL     66  
#define INPUT1_MUL     77  
#define INPUT2_MUL     88  
#define INPUT3_MUL     99  
#define INPUT4_MUL    100
```

Uso: en el siguiente ejemplo se escribe la variable de entrada FUERZA en 15, la variable de salida ACELERACIÓN con el valor de FUERZA más 4, o sea 19.

```

/*
 * Aca el usuario escribe su experimento.
 */

static void experiment( void ){
    // Escribir experimento

    Led.n_blink(5, 500);
    Memory.set_input(FUERZA, 15);
    Memory.set_output(ACELERACION, Memory.get_input(FUERZA) + 4 );
}

```


**Ver ANEXO A.4**

## Anexo A. Usos de RemoteLabLib


### 1 Remote Lab Lib uso de soporte para impresión por puerto serie

 Remote Lab Lib uso de soporte para impresion por puerto serie


### 2 Demostración del soporte para Arduino Plotter

 RemoteLab-lib Arduino Plotter

### 3 Demostración del soporte para modos de ejecución.

 RemoteLabLib uso del estado modo.

### 4 Demostración del uso del acceso a memoria no volátil del Arduino

 Demostración del uso del acceso a memoria no volátil del Arduino

## Anexo B. Intención del sistema de RemoteLab

5 Demostración del sistema de desarrollo de experimentos con Arduino controlados por una interfaz-web

 Intencion del proyecto RemoteLab-LIb Arduino