



UNIVERSIDAD NACIONAL DE CÓRDOBA
FACULTAD DE MATEMÁTICA ASTRONOMÍA
FÍSICA Y COMPUTACIÓN

TRABAJO ESPECIAL DE LA LICENCIATURA EN
CIENCIAS DE LA COMPUTACIÓN

Sistema de gestión de experimentos y
control de Módulo Digital para
Resonancia Magnética Cuadrupolar y
Nuclear

Victor Pablo Bovina Astorga

dirigida por
Ing. Walter ZANINETTI

Diciembre 2018



"Sistema de gestión de experimentos y control de Módulo Digital para Resonancia Magnética Cuadrupolar y Nuclear" por Victor Pablo Bovina Astorga, se distribuye bajo la Licencia Creative Commons Atribución-NoComercial-SinDerivadas 2.5 Argentina.

Contents

1	Marco de trabajo	6
1.1	Rol del Software	6
2	Partes de un experimento	7
2.0.1	Radio frecuencia	7
2.0.2	Pulso TTL	7
2.0.3	Muestras	7
2.1	Tipo de Experiencia	7
2.1.1	Experiencia Promedio	7
2.1.2	Experiencia Promedio con Pulso variable	7
3	Diagrama de conexiones	8
3.1	Mixer	8
3.2	Amplificador	8
3.3	Resonador	8
4	Descripción del módulo digital	9
5	Submódulo PP2	10
5.1	Instrucciones	10
5.1.1	Duración de una instrucción	10
5.1.2	Continue	10
5.1.3	Loop	10
5.1.4	Retl	11
5.1.5	End	11
5.2	Registros del PP2	11
5.2.1	Carga de un programa en el PP2.	12
5.2.2	Ejecución de un programa.	12
6	Submódulo DDS2	13
6.1	Configuración inicial, activación y desactivación.	13
6.2	Registros del DDS2	14
6.3	Fases	14
6.3.1	Algoritmo base de almacenamiento de fases	15
6.3.2	direccionamiento de fases	15
6.4	Frecuencias	15
6.4.1	Algoritmo base de almacenamiento de frecuencias	16
6.4.2	Direccionamiento de frecuencias	16
7	Submódulo AD	17
7.1	Registro de comando	17
7.2	Registro de datos	17
7.3	Configuración de un ciclo de adquisición	18
7.3.1	Configuración bloque de 1KB y muestreo 1 microsegundo	18
7.4	Extracción de los datos de la memoria interna	19

8 Submodulo USB	21
9 Visión general del sistema	22
9.1 Navegador web	22
9.2 Servidor	22
9.3 Módulo Digital	22
9.4 Resonador	22
10 Definición de un experimento	23
11 Algoritmo de traducción de experimentos	24
12 Ejecución de un experimento	27
13 Hilo de ejecución de un experimento	28
14 Run y DryRun	29
15 Requisitos del sistema	30
16 Planificacion del desarrollo	32
17 Planificacion de Testing	34
18 Arquitectura del Modulo Digital	36
19 Arquitectura del Servidor	37
20 Arquitectura de Interfaz Grafica	38
21 Integracion de los submodulos	39
21.1 Repositorio	39
21.2 Codigo	39
22 Analisis de Performance	40
23 Casos de Prueba	41
23.1 Modulo Digital	41
23.2 Servidor	41
23.3 Interfaz Grafica	42
24 Errores conocidos	43
25 Limitaciones de la plataforma	44
26 Mejoras a futuro	45
27 Analisis de esfuerzo	46

Agradecimientos

Quiero agradecer al estado en democracia y a los ciudadanos que lo componen por haberme brindado la educación pública y gratuita la cual me brindó por medio de los docentes una educación de calidad y competitiva.

Recuerdo mis primeros días en FaMAF, hice mi ingreso en Agosto por adelantado para tener mis vacaciones extendidas en verano y no cursar en Febrero el ingreso.

Los 3 primeros años fueron durísimos. Luego comencé a trabajar para hacer algo distinto. Volví con muchas ganas después de ponerme de novio buscando más conocimiento y experiencia. Termine de cursar y volví a trabajar, en cosas mucho mejores.

El espíritu de los docentes me mantuvo vivo, esto es fruto de eso.

Muchas cosas decantan con el tiempo, sobre todo las ideas más profundas.

Ciencias de la computación es la carrera que volvería a elegir si tuviese que volver a empezar, por el temple de mis docentes y lo que significa la computación en estos tiempos.

Este camino no termina acá, "te sobran años Bovina" una frase de un profesor que me enseñó que el tiempo aun lo tengo, que vale y lo tengo que usar muy bien.

Gracias FaMAFC, familia, amigos y director de tesis!

Paciencia Paciencia Más Paciencia!

era verdad ...

1 Marco de trabajo

En el proceso de investigación del fenómeno físico de *Resonancia Magnética Cuadrupolar y Nuclear* el investigador manipula módulos electrónicos digitales de medición precisos y estables, en algunos casos si intervienen mas de uno a la vez, entre ellos sincronizados por medio de interfaces digitales. Estos módulos electrónicos digitales colaboran con la creación del contexto necesario para investigar lo planeado según la necesidad del investigador. En general junto con los módulos electrónicos digitales intervienen otros módulos electrónicos de naturaleza analógica tales como amplificadores operacionales, mezcladores de señales, filtros pasa bajos, entre otros.

La correcta conexión entre los diferentes módulos electrónicos digitales, analógicos y su configuración durante el proceso de experimentación son responsabilidad del investigador y una tarea de suma importancia para el éxito de la experiencia a realizar.

En este marco de responsabilidades del investigador y el módulo digital a ser utilizado es deseable y necesario el desarrollo de mecanismos sencillos de manipulación del mismo y su monitoreo.

1.1 Rol del Software

El rol primario del software en el contexto descripto previamente es el de manipular el módulo digital a través de la PC utilizada por el investigador de forma local o remota.

El rol secundario la administración de usuarios del módulo digital, monitoreo de los experimentos en curso y resultados.

2 Partes de un experimento

Estos son algunos conceptos clave para comprender el contexto y definición de un experimento, los cuales se tuvieron en cuenta al momento de la implementación del sistema.

2.0.1 Radio frecuencia

Es una señal senoidal que tiene como objetivo estimular la muestra presente en el resonador y que puede ser manipulada previamente por otros módulos externos. Tiene los siguientes atributos:

- Frecuencia: 0 a 80 megahercios.
- Fase: 0 a 360 grados.
- Duración: 0 a 16 segundos.

2.0.2 Pulso TTL

Es una señal digital de amplitud 5 voltios y duración predefinida con la finalidad de transmitirla como entrada a otros módulos digitales o analógicos para sincronizar momentos de relajación y estimulación de las muestras durante la experiencia.

2.0.3 Muestras

Es un grupo de datos obtenidos a cierta frecuencia de muestreo y agrupados de manera contigua en un bloque de longitud 2^n con $n \in \mathbb{N}$.

2.1 Tipo de Experiencia

El tipo de experiencia a realizar esta determinada por el investigador, existen 2 bien diferenciadas:

2.1.1 Experiencia Promedio

Una experiencia promedio es una que se repite y donde los bloques de muestras ordenadas se suman uno a uno obteniendo una mejor representación los datos para su análisis.

2.1.2 Experiencia Promedio con Pulso variable

Es una experiencia promedio donde la configuración de los pulsos cambia en las sucesivas iteraciones de la misma. Los pulsos cambian su configuración para suprimir interferencias de estimulaciones previas y obtener mediciones mas precisas. Los atributos del pulso que pueden cambiar en las sucesivas iteraciones son la fase y duración.

3 Diagrama de conexiones

Este diagrama representa la interconexión física entre los diferentes módulos digitales y analógicos que forman parte de una experiencia planificada.

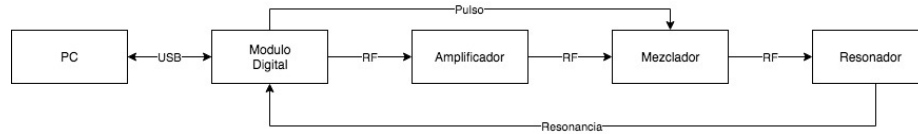


Figure 1: Diagrama de conexiones

3.1 Mixer

Los pulsos de salida y la señal de radiofrecuencia son las entradas del mezclador donde se convierten en una sola cuando el pulso TTL esta activo en 5 voltios. Esto permite crear pulsos de estimulación y relajación con presición.

3.2 Amplificador

3.3 Resonador

4 Descripción del módulo digital

El siguiente diagrama de bloques muestra la configuración interna del aparato junto a sus características técnicas.

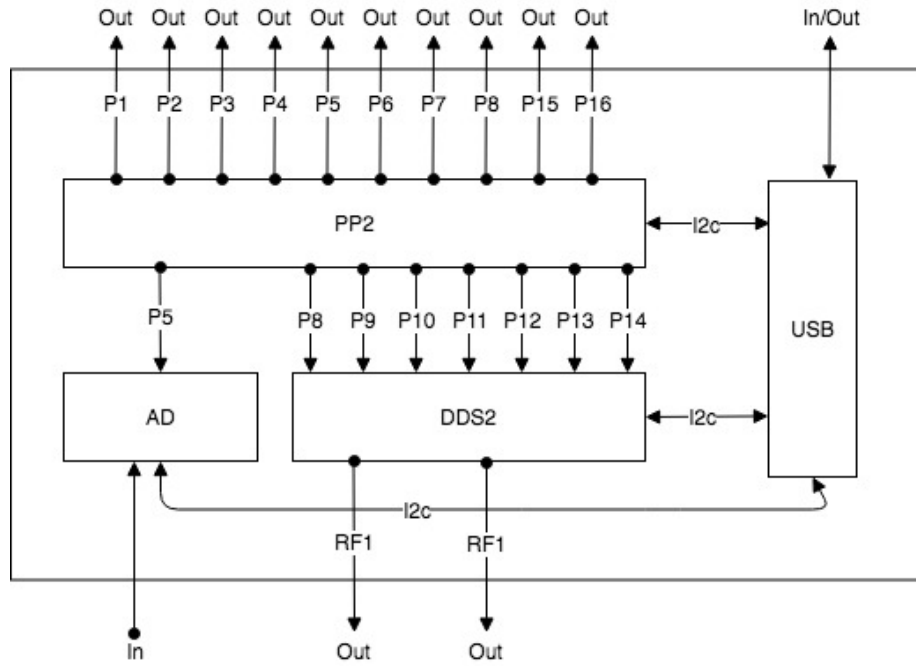


Figure 2: Diagrama de bloques del módulo digital

5 Submódulo PP2

El programador de pulsos digitales PP2 es un microprocesador diseñado a medida con instrucciones que generan un *patrón de salida*. Un *patrón de salida* es una combinación de 16 pulsos TTL en paralelo por un periodo de tiempo determinado. Un programa ejecutable por el PP2 será una secuencia de no más de 512 instrucciones y la ejecución derivará en una *secuencia de patrones de salida* de duración determinada.

5.1 Instrucciones

El PP2 tiene 4 instrucciones básicas: *Continue*, *Retl*, *Loop* y *End* cada una es una secuencia de 64 bits con la siguiente estructura:

Patrón de salida	Dato	Nivel de lazo	Código de instrucción	Duración
16 bits	11 bits	2 bits	3 bits	32 bits

Table 1: Estructura de las instrucciones.

5.1.1 Duración de una instrucción

Cada instrucción requiere de 4 pulsos de reloj ($4 * 40ns = 160ns$) y la duración mínima es 2 pulsos ($2 * 40ns = 80ns$). Por lo tanto el pulso de valle mínimo es de $160ns + 80ns = 240ns$.

5.1.2 Continue

Mantiene una combinación de 16 pulsos de salida por un tiempo determinado.

- patrón de salida: es la combinación de 16 pulsos de salida.
- dato: no utilizado.
- nivel de lazo: no utilizado.
- código de instrucción: 0x01.
- duración: duración de la instrucción.

5.1.3 Loop

Marca el apertura de un bloque de repetición.

- patrón de salida: es la combinación de 16 pulsos de salida.
- dato: contador de repeticiones.
- nivel de lazo: nivel de profundidad de anidamiento entre lazos.

- código de instrucción: 0x02.
- duración: duración de la instrucción.

5.1.4 Retl

Marca de cierre de un bloque de repetición.

- patron de salida: es la combinación de 16 pulsos de salida.
- dato: dirección de la instrucción Loop de inicio.
- nivel de lazo: no utilizado.
- código de instrucción: 0x03.
- duración: duración de la instrucción.

5.1.5 End

Finaliza la secuencia de pulsos.

- patrón de salida: es la combinación de 16 pulsos de salida.
- dato: no utilizado.
- nivel de lazo: no utilizado.
- código de instrucción: 0x07.
- duración: duración de la instrucción.

5.2 Registros del PP2

El PP2 cuenta con los siguientes registros de 8 bits:

Dirección	Descripción	modo
0x50	comando	escritura
0x51	carga	escritura
0x52	señal	escritura

Table 2: Estructura de las instrucciones.

5.2.1 Carga de un programa en el PP2.

Algorithm 1 Carga de una programa en el PP2

```
1: procedure UPLOADPROGRAM()
2:   // Reset
3:   write(0x50, 0x02)
4:   // Modo carga
5:   write(0x50, 0x03)
6:   // Continue 0x55AA 4
7:   write(0x51, 0x04)
8:   write(0x51, 0x00)
9:   write(0x51, 0x00)
10:  write(0x51, 0x00)
11:  write(0x51, 0x01)
12:  write(0x51, 0x00)
13:  write(0x51, 0xAA)
14:  write(0x51, 0x55)
15:  write(0x52, 0x00)
16:  // End
17:  write(0x51, 0x00)
18:  write(0x51, 0x00)
19:  write(0x51, 0x00)
20:  write(0x51, 0x00)
21:  write(0x51, 0x00)
22:  write(0x51, 0x00)
23:  write(0x51, 0x00)
24:  write(0x51, 0x00)
25:  write(0x52, 0x00)
```

5.2.2 Ejecución de un programa.

Algorithm 2 Ejecución de un programa.

```
1: procedure EXCUTEPROGRAM()
2:   // Modo microprocesador
3:   write(0x50, 0x00)
4:   // Señal de ejecución
5:   write(0x08)
```

6 Submódulo DDS2

El submódulo DDS2 es un generador de señales digitales con un rango de frecuencia de 0 a 80 Mhz. El usuario puede almacenar hasta 2 frecuencias y un total de 16 fases para combinarlos en un experimento. Existe una relación entre el PP2 y el DDS2, puesto que este tiene como entrada los pulsos 8,9,10,11,12,13,14 para su manipulación externa por aquel durante la ejecución de un programa.

6.1 Configuración inicial, activación y desactivación.

Dirección	B7	B6	B5	B4	B3	B2	B1	B0	Hex
1D	0	0	0	1	1	1	1	1	0x17
1E	0	1	0	0	0	1	0	0	0x44
1F	0	0	0	0	0	0	1	0	0x02
20	0	0	0	0	0	0	0	0	0x00

Table 3: Reset

Direccion	B7	B6	B5	B4	B3	B2	B1	B0	Hex
1D	0	0	0	1	0	0	0	0	0x10
1E	0	1	0	0	0	1	0	0	0x44
1F	0	0	0	0	0	0	1	0	0x02
20	0	0	0	0	0	0	0	0	0x00

Table 4: Activación

Dirección	B7	B6	B5	B4	B3	B2	B1	B0	Hex
1D	0	0	0	1	1	1	1	1	0x17
1E	0	1	0	0	0	1	0	0	0x44
1F	0	0	0	0	0	0	1	0	0x02
20	0	0	0	0	0	0	0	0	0x00

Table 5: Desactivación

6.2 Registros del DDS2

Dirección	Descripción	Modo
0x70	direccionamiento	Escritura
0x71	modo	Escritura
0x72	reset	Escritura
0x73	test	Lectura
0x74	señal de escritura	Escritura
0x75	direccionamiento	Escritura
0x76	señal de transferencia	Escritura
0x77	test	Lectura
0x78	señal de escritura	Escritura

Table 6: Registros internos del DDS2.

6.3 Fases

El DDS2 tiene disponible 32 posiciones de memoria 8 bits para el almacenamiento de fases con un bus de datos de 8 bits. La fase es un valor de 14 bits, los 2 bits restantes del MSB son descartados. El MSB de una fase debe almacenarse siempre en una dirección par de la memoria, luego el LSB de la misma en el valor impar contiguo siguiente. Antes de almacenar el valor entero de la fase F debemos convertirlo a su equivalente en unidades de 45 de la siguiente manera:

$$F_h \in \mathbb{N} \quad (1)$$

$$F_l \in \mathbb{N} \quad (2)$$

$$F \in \mathbb{N} \quad (3)$$

$$F_h = (45 * F) / 256 \quad (4)$$

$$F_l = (45 * F) - (F_h * 256) \quad (5)$$

direcciones de ram disponibles	Modo
0x00 0x02 0x04 0x06 0x08 0x0A 0x0C 0x0E	0x02
0x10 0x12 0x14 0x16 0x18 0x1A 0x1C 0x1E	0x02

Table 7: Direcciones de Fase.

6.3.1 Algoritmo base de almacenamiento de fases

Algorithm 3 Almacenamiento de una fase en direccion 0x00

```
1: procedure SAVEPHASE(F)
2:   // MSB y LSB de valor de fase
3:    $F_h = (45 * F) / 256$ 
4:    $F_l = (45 * F) - (F_h * 256)$ 
5:   // direcciones contiguas
6:    $dir_h \leftarrow 0x00$ 
7:    $dir_l \leftarrow dir_h + 1$ 
8:   // Modo carga de fases
9:    $write(0x71, 0x02)$ 
10:  // MSB
11:   $write(0x70, dir_h)$ 
12:   $write(0x74, f_h)$ 
13:  // LSB
14:   $write(0x70, dir_l)$ 
15:   $write(0x74, f_l)$ 
16:  // Modo PC
17:   $write(0x71, 0x00)$ 
```

6.3.2 direccionamiento de fases

Los 16 valores de fase son direccionados con 4 pulsos correspondientes a los pulsos 11,12,13,14. Con el pulso 9 se activa la carga de fases y con el pulso 10 se transfiere la fase direccionada al registro de trabajo. El tiempo entre el pulso 9 y 10 debe ser menor a 100 nanosegundos.

6.4 Frecuencias

El DDS2 tiene disponible 12 registros internos de frecuencia de 8 bits, cada frecuencia ocupa 6 registros, por lo tanto se pueden almacenar 2 frecuencias. Las frecuencias disponibles van de 0 a 80 Mhz y el valor que se almacena en los registros en representación se obtiene con la siguiente cálculo:

$$clock = 2 \times 10^6 \quad (6)$$

$$freq \in \mathbb{N} \wedge 0 < freq < clock \quad (7)$$

$$valor = freq \times (2^{48} - 1) / clock \quad (8)$$

Nro. Frecuencia	Registros	Modo
1	0x04 0x05 0x06 0x07 0x08 0x09	0x00
2	0x0A 0x0B 0x0C 0x0D 0x0E 0x0F	0x00

Table 8: Registros de Frecuencia.

6.4.1 Algoritmo base de almacenamiento de frecuencias

Algorithm 4 Almacenamiento de una frecuencia de trabajo 1.

```

1: procedure SAVEFREQUENCY(F)
2:   // Conversión de la frecuencia deseada al valor
3:    $clock \leftarrow 2 \times 10^6$ 
4:    $value = freq \times (2^{48} - 1) / clock$ 
5:   // Modo PC
6:    $write(0x71, 0x00)$ 
7:   // primero MSB hasta LSB
8:   // Byte 5
9:    $write(0x75, 0x04)$ 
10:   $write(0x78, value_5)$ 
11:  // Byte 4
12:   $write(0x75, 0x05)$ 
13:   $write(0x78, value_4)$ 
14:  // Byte 3
15:   $write(0x75, 0x06)$ 
16:   $write(0x78, value_3)$ 
17:  // Byte 2
18:   $write(0x75, 0x07)$ 
19:   $write(0x78, value_2)$ 
20:  // Byte 1
21:   $write(0x75, 0x08)$ 
22:   $write(0x78, value_1)$ 
23:  // Byte 0
24:   $write(0x75, 0x09)$ 
25:   $write(0x78, value_0)$ 
26:  // Actualización registro de trabajo
27:   $write(0x76, 0x00)$ 

```

6.4.2 Direccionamiento de frecuencias

Se admiten hasta 2 frecuencias de trabajo, se seleccionan con el pulso 8.

7 Submódulo AD

El submódulo conversor analógico digital AD tiene 2 canales de adquisición en paralelo con una resolución de 12 bits por canal, una frecuencia de muestreo máxima de 10 Mhz y capacidad de almacenamiento en bloques de 1KB, 2KB, 4KB, 8KB, 16KB, 32KB, 64KB, 128KB.

Dirección	Descripción	modo
0x0B	comando y control	lectura/escritura
0x0C	muestreo	escritura
0x08	canal B	lectura
0x09	canal AB	lectura
0x0A	canal A	lectura

Table 9: Registros del AD

7.1 Registro de comando

Bit	Descripción
0	modo
1	modo
2	-
3	-
4	bloque
5	bloque
6	bloque
7	reset

Table 10: Registro de comando.

7.2 Registro de datos

Dirección	Descripción
0x08	canal B
0x09	canal AB
0x0A	canal A

Table 11: Registros de datos.

7.3 Configuración de un ciclo de adquisición

Un ciclo de adquisición requiere la configuración del intervalo de muestreo de acuerdo a la siguiente fórmula:

$$period = 100ns \quad (9)$$

$$interval \in \mathbb{N} \wedge 0 < interval < 25500 \quad (10)$$

$$n = interval/period \quad (11)$$

$$delta = 255 - n \quad (12)$$

$$(13)$$

7.3.1 Configuración bloque de 1KB y muestreo 1 microsegundo

Algorithm 5 Configuración 1KB 1 microsegundo

```
1: procedure SETUP()
2:
3:   config  $\leftarrow$  0x00
4:   config  $\leftarrow$  config  $\vee$  0x02
5:   config  $\leftarrow$  config  $\vee$  0x80
6:   config  $\leftarrow$  config  $\wedge$  0xFE
7:   write(0x0B, config)
8:
9:   config  $\leftarrow$  0x00
10:  config  $\leftarrow$  config  $\vee$  0x02
11:  config  $\leftarrow$  config  $\wedge$  0x7F
12:  config  $\leftarrow$  config  $\vee$  0x01
13:  write(0x0B, config)
14:
15:  period  $\leftarrow$  100
16:  samples = 1000/period
17:  delta = 255 - samples
18:  write(0x0C, delta)
```

7.4 Extracción de los datos de la memoria interna

El bus de datos de la memoria del AD es de 8 bits y las muestras de 12 bits, siendo necesarias 3 lecturas de 8 bits y una operación de partición para obtener la muestra final de los canales A y B.



Figure 3: Buffers del submódulo AD

Para la extracción de los datos es necesaria esta serie de pasos:

- modo PC, deshabilitada la adquisición y reset contador de direcciones
- modo PC, deshabilitada la adquisición y contador de direcciones en modo normal
- lectura del registro 0x08 del canal B
- modo PC, deshabilitada la adquisición y reset contador de direcciones
- modo PC, deshabilitada la adquisición y contador de direcciones en modo normal
- lectura del registro 0x0A del canal A
- modo PC, deshabilitada la adquisición y reset contador de direcciones
- modo PC, deshabilitada la adquisición y contador de direcciones en modo normal
- lectura del registro 0x09 con nibbles del canal A y B.
- armado de los datos de los respectivos canales.

Algorithm 6 Extracción de datos del AD

```
1: procedure GETDATAFROMAD()
2:   config  $\leftarrow$  0x00
3:   config  $\leftarrow$  config  $\vee$  0x02
4:   config  $\leftarrow$  config  $\vee$  0x80
5:   config  $\leftarrow$  config  $\wedge$  0xFE
6:   write(0x0B, config)
7:   config  $\leftarrow$  0x00
8:   config  $\leftarrow$  config  $\vee$  0x02
9:   config  $\leftarrow$  config  $\wedge$  0x7F
10:  config  $\leftarrow$  config  $\wedge$  0xFE
11:  write(0x0B, config)
12:  A  $\leftarrow$  read(0x08)
13:
14:  config  $\leftarrow$  0x00
15:  config  $\leftarrow$  config  $\vee$  0x02
16:  config  $\leftarrow$  config  $\vee$  0x80
17:  config  $\leftarrow$  config  $\wedge$  0xFE
18:  write(0x0B, config)
19:  config  $\leftarrow$  0x00
20:  config  $\leftarrow$  config  $\vee$  0x02
21:  config  $\leftarrow$  config  $\wedge$  0x7F
22:  config  $\leftarrow$  config  $\wedge$  0xFE
23:  write(0x0B, config)
24:  B  $\leftarrow$  read(0x0A)
25:
26:  config  $\leftarrow$  0x00
27:  config  $\leftarrow$  config  $\vee$  0x02
28:  config  $\leftarrow$  config  $\vee$  0x80
29:  config  $\leftarrow$  config  $\wedge$  0xFE
30:  write(0x0B, config)
31:  config  $\leftarrow$  0x00
32:  config  $\leftarrow$  config  $\vee$  0x02
33:  config  $\leftarrow$  config  $\wedge$  0x7F
34:  config  $\leftarrow$  config  $\wedge$  0xFE
35:  write(0x0B, config)
36:  AB  $\leftarrow$  read(0x09)
37:
38:  sample_A  $\leftarrow$  join_buffers(A, AB)
39:  sample_B  $\leftarrow$  join_buffers(B, AB)
```

8 Submodulo USB

9 Visión general del sistema

Este diagrama respresenta la interacción entre los diferentes elementos de hardware y software en una experiencia planificada.

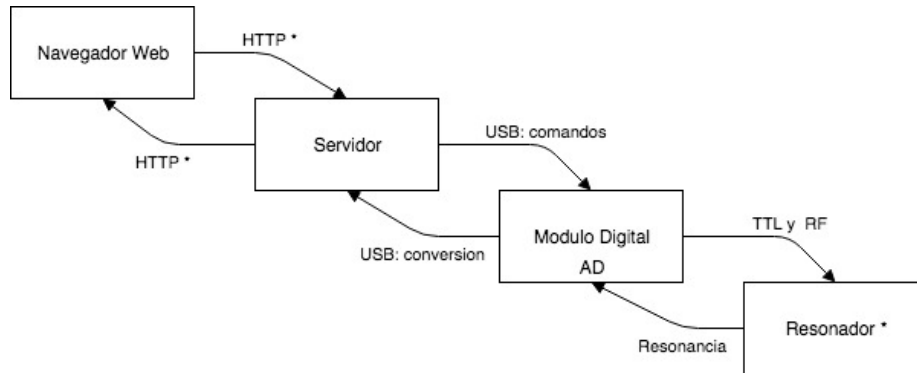


Figure 4: Visión General del sistema

9.1 Navegador web

El navegador web renderiza la *Interfaz Gráfica* solicitada al *Servidor*, esta provee al usuario final el control remoto del sistema.

9.2 Servidor

El *Servidor* provee servicios REST solicitados por la *Interfaz Gráfica* durante la vida de la sesión del usuario. Estos servicios hacen llamadas al controlador del *Módulo Digital* via USB.

9.3 Módulo Digital

El *Módulo Digital* procesa los mensajes del controlador via USB y ejecuta el microcódigo del mismo para la configuración y ejecución de las secuencias de pulsos.

9.4 Resonador

El *Resonador* recibe los pulsos provenientes del *Módulo Digital* generando una señal de resonancia enviada al *Módulo Digital* para su conversión digital.

10 Definición de un experimento

Para representar la definición de un experimento se eligió notación JSON, por el soporte built-in por parte de los lenguajes elegidos para desarrollar el sistema, Python y Javascript y la implementación de servicios REST necesarios en el *Servidor*

Haciendo uso de aritmética podemos deducir que la representación de un experimento en JSON es liviana puesto que el número de instrucciones de un programa P sera de $N < 512$ y para un archivo de texto de 512 líneas de 80 columnas de caracteres de 1 Byte tiene un peso aproximado de 327.68 Kilobytes.

La validación del esquema JSON de un experimento, es llevada a cabo por un módulo programado dentro del sistema para tal fin.

Cabe destacar que los tipos de datos presentes en la especificación de un experimento son soportados sin pérdida de precisión por parte de la notación JSON.

11 Algoritmo de traducción de experimentos

Un programa almacenable y ejecutable en el PP2 es una secuencia de instrucciones $I_1 \dots I_N$ con $1 < N < 512$.

Donde las instrucciones disponibles son:

- *Continue*
- *Loop*
- *Retl*
- *End*

No todos los programas escritos con estas instrucciones son admitidos como válidos para el PP2, por esto tenemos las siguientes reglas:

- Regla 1: la instrucción *End* siempre es la última.
- Regla 2: la instrucción *End* aparece solo una vez.
- Regla 3: la instrucción *End* siempre esta presente.
- Regla 4: un bucle siempre inicia con instrucción *Loop* y finaliza con *Retl*.
- Regla 5: puede haber hasta 3 instrucciones *Loop* anidados en un *Loop*.
- Regla 6: la longitud del programa no puede ser mayor a 512 instrucciones.

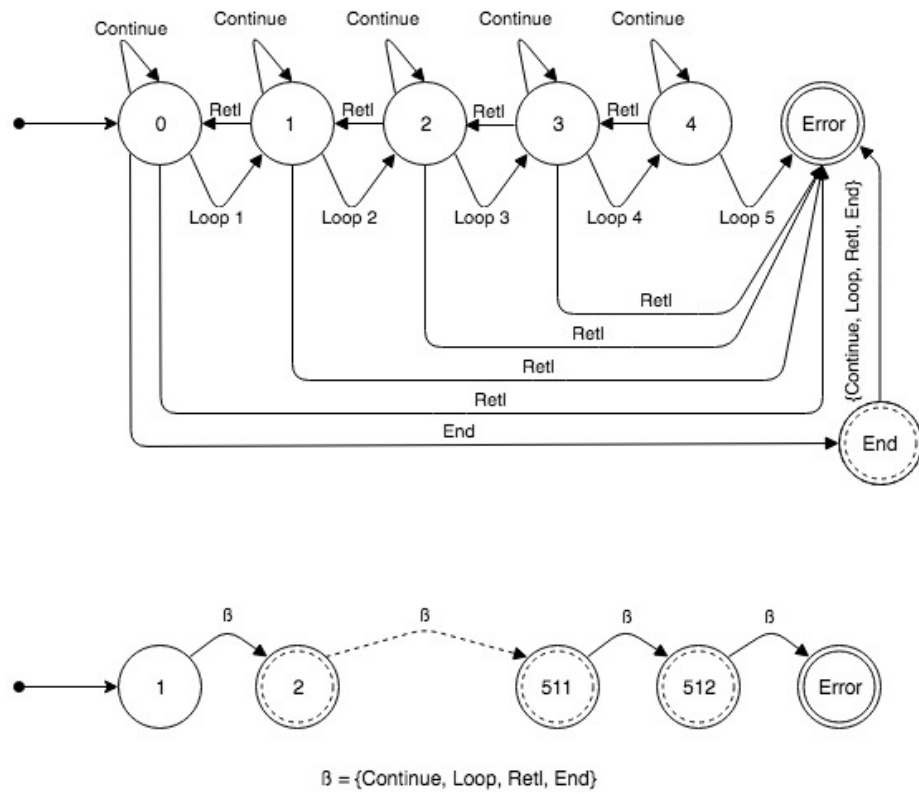


Figure 5: Los programas aceptados por el PP2.

Algorithm 7 Algoritmo de traducción base

```
1: procedure TRANSLATE( $P, L=0, S=[]$ )
2:    $ins \leftarrow pop(P)$ 
3:   if  $ins = Continue$  then
4:      $S.append(ins)$ 
5:   if  $ins = Retl$  then
6:     if  $L > 0$  then
7:        $S.append(ins)$ 
8:        $L \leftarrow L - 1$ 
9:        $Translate(P, L, S)$ 
10:    else
11:      return error
12:   if  $ins = Loop$  then
13:     if  $L < 4$  then
14:        $S.append(ins)$ 
15:        $L \leftarrow L + 1$ 
16:        $Translate(P, L, S)$ 
17:     else
18:       return error
19:   if  $ins = End$  then
20:     if  $len(P) = 0$  then
21:        $S.append(ins)$ 
22:     else
23:       return error
24:
25:   assert( $0 < len(S) < 512$ )
26:   return  $S$ 
```

12 Ejecución de un experimento

Un modo de capturar fallos es simular la ejecución de un experimento, evitando la interacción via usb, con el módulo digital. En modo simulado el experimento se ejecuta en un entorno limitado con el objetivo de detectar tres situaciones no deseadas:

- Una excepción de software no capturada.
- Parámetros fuera de rango para alguno de los submódulos.
- Un error en la traducción del experimento a un programa ejecutable en el PP2.

Un experimento simulado se ejecuta el hilo principal, puesto que la demora es baja. De manera alternativa el modo simulado puede verse también como un test de integración entre las unidades de software desarrolladas. Si el experimento simulado tiene éxito entonces se procede con la ejecución efectiva del experimento que se desprende del hilo principal y en el cual debemos tener algunos detalles en cuenta:

- Puede durar horas.
- Puede ser interrumpido por el usuario en cualquier momento.
- Debe estar sincronizado con la base de datos.

El siguiente diagrama muestra el flujo de ejecución de un experimento simulado y no simulado.

13 Hilo de ejecución de un experimento

La duración de un experimento puede ser de horas, esta característica involucra tres problemáticas a resolver:

- Evitar time-out del lado del cliente web.
- Desacoplar la atención de una petición HTTP de usuario de la ejecución de un experimento.
- Lograr una experiencia de usuario agradable en lo que respecta a performance.

Modelando el proceso de ejecución con un hilo separado del principal es una solución válida para resolver el problema y teniendo en cuenta los siguientes:

- Seguimiento del estado del hilo.
- Control sobre el numero de hilos creados.
- Gestión de los datos durante la ejecución del hilo.

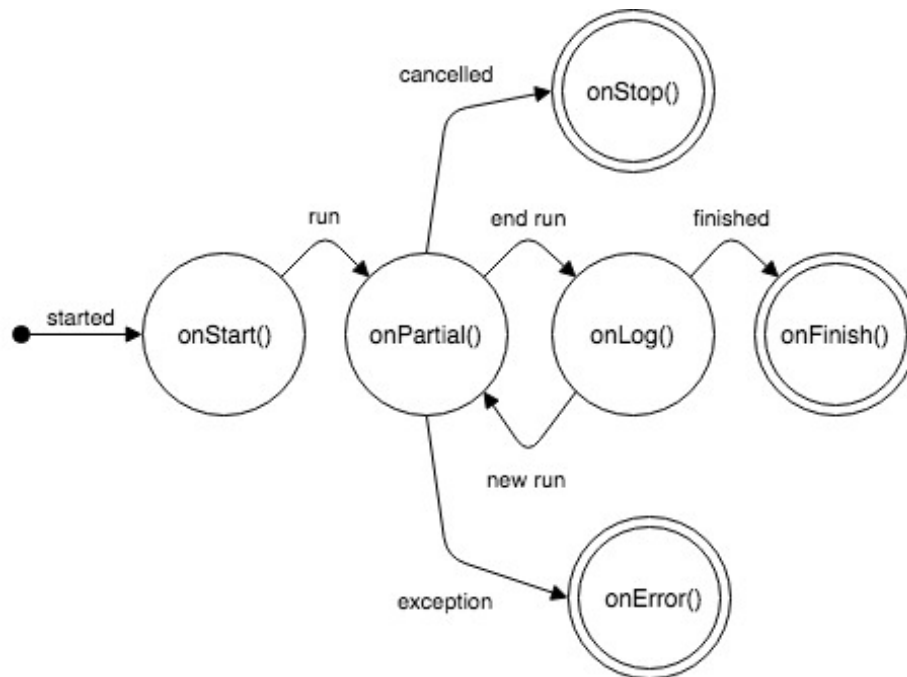


Figure 6: Estados de un hilo de experimento.

14 Run y DryRun

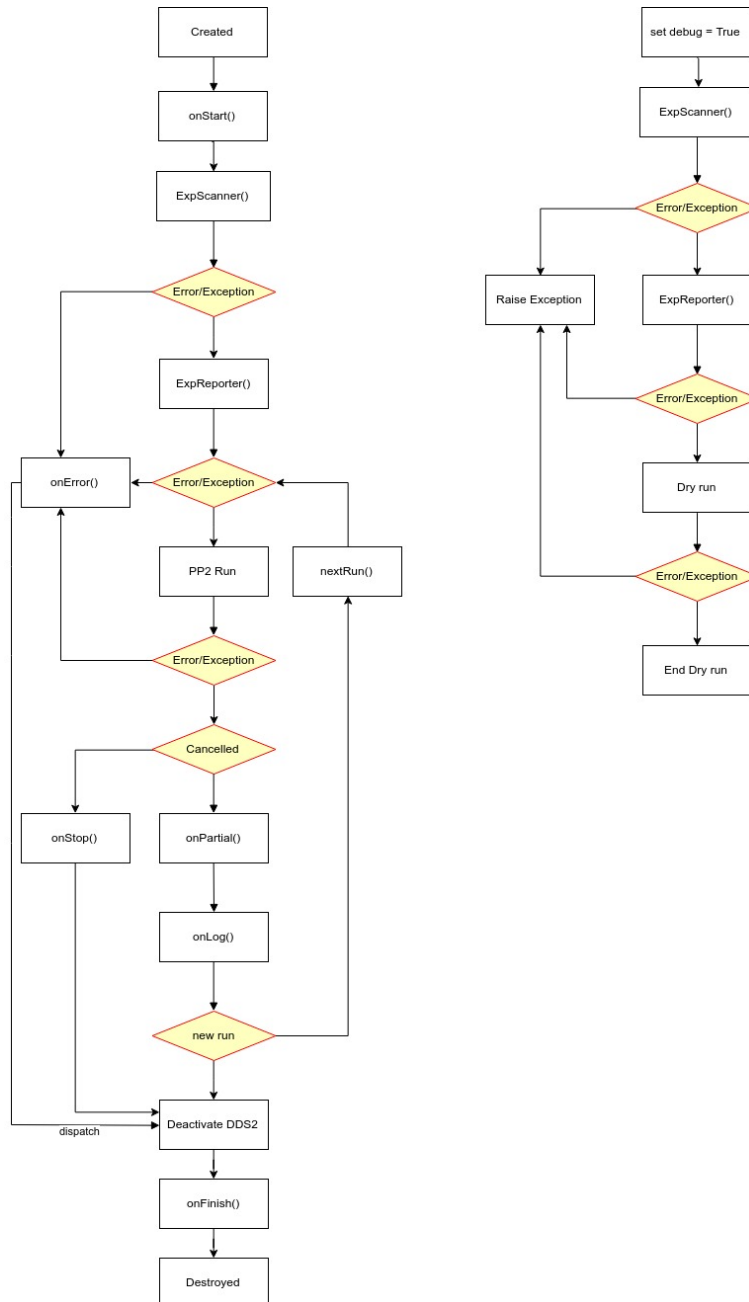


Figure 7: Run y DryRun

15 Requisitos del sistema

Podemos ver los requerimientos en generales

- permitir a mas de un usuario utilizar el sistema
- permitir a usuarios la gestion de experimentos
- modelar un experimento y su ejecucion
- proveer acceso remoto al sistema

De estos requerimientos se desprenden los siguientes especificos:

- el usuario tiene una sesion asignada al ingreso del sistema y revocada a la salida del mismo.
- varios usuarios pueden acceder al sistema al mismo tiempo
- el usuario puede ver si hay un experimento en ejecucion y el detalle del mismo
- el usuario tiene un espacio de trabajo donde puede crear/ver/actualizar/eliminar sus experimentos
- el usuario puede cancelar el experimento en ejecucion
- el usuario puede solicitar un reporte parcial de un experimento en ejecucion
- el usuario puede solicitar el reporte final de un experimento finalizada la ejecucion
- el usuario puede verificar el estado de un experimento:
- el usuario es notificado antes de la ejecucion de un experimento cuando:
 - salida voluntaria por error:
 - * no es una secuencia ejecutable por el PP2
 - * ya existe algun experimento ejecutandose
 - * algun parametro de configuracion es invalido en algun submodulo PP2, DDS2, AD, USB
 - * fallo en conexion via USB
 - salida involuntaria por error:
 - * hubo alguna excepcion de sistema no controlada
 - * no se pudo establecer conexion con el servidor
 - * un experimento finalizado no actualizo su estado impidiendo la ejecucion de solcitido

- un experimento tiene una secuencia definida con sus parametros
- un experimento tiene una marca de tiempo de creacion/actualizacion
- un experimento eliminado no es recuperable
- un experimento tiene un autor que es el usuario que lo creo
- un experimento tiene estado created cuando esta almacenado en base de datos
- un experimento es solo visible para el usuario que lo creo
- un experimento tiene un titulo
- un experimento tiene una descripcion
- un experimento no tiene un historial de edicion asociado
- un resultado es el producto de la ejecucion de un experimento
- un resultado es parcial cuando la ejecucion del experimento asociado no finalizo
- un resultado es final cuando la ejecucion del experimento asociado finalizo
- un resultado tiene un unico experimento asociado
- un reporte parcial es el grupo de datos de un resultado parcial
- un reporte final es el grupo de un resultado final
- un reporte contiene:
 - log de la ejecucion
 - datos del AD en formato CSV
 - experimento ejecutado
- el sistema tiene servicios REST para:
 - crear/ver/editar/eliminar experimentos
 - iniciar/cancelar ejecucion de un experimento
 - cancelar la ejecucion de todos los experimentos
 - descargar reportes parciales y finales
 - proveer autenticacion a todos los servicios
- el sistema tiene una interfaz de usuario web

16 Planificacion del desarrollo

De acuerdo con los requerimientos descritos en el capitulo XX se propone implementar el sistema con:

- Django framework
- Python 2.7
- React js
- Javascript ECM6
- Git

Las tecnologias elegidas son ampliamente soportadas por la comunidad de desarrolladores, de codigo libre y soporte multiplataforma en PC.

Con el objetivo de asegurar que lo primero que se desarrolle cumpla son los requerimientos asociados con el control del Modulo Digital, luego proveer una manipulacion remota y finalmente una interfaz de usuario para el manejo a traves del navegador web, se organizo el desarrollo de la siguiente manera:

- Modulo Digital
- Servidor
- Interfaz Grafica Web

El mapeo entre los requerimientos y el orden propuesto genera el siguiente esquema de tareas a realizar.

- Modulo Digital
 - Configuracion del proyecto y entornos
 - Implementacion control AD
 - Implementacion control DDS2
 - Implementacion control PP2
 - Implementacion abstraccion Experimento
 - Implementacion de interfaz USB
 - Implementacion de un programa principal
 - Gestion de logs
 - Integracion con Servidor
- Servidor
 - Configuracion del proyecto y entornos
 - Servicios REST para la edicion de experimentos
 - Servicios REST para control de ejecucion de experimentos

- Servicios REST para la generacion de reportes
- Modelado de Base de datos
- Configuracion de sesiones
- Configuracion de acceso a los servicios REST
- Gestion de logs
- Integracion con Interfaz WEB
- Integracion con Modulo Digital
- Interfaz WEB
 - Configuracion de proyecto y entornos
 - Implementacion de transiciones
 - Componente de edicion de experimentos
 - Componente de control de ejecucion de experimentos
 - Componente de login
 - Componente de visualizacion de errores
 - Componente de definicion de experimento
 - Implementacion de control de estados
 - Integracion con Servidor

17 Planificacion de Testing

La planificacion de testing del sistema tiene varios enfoques en respuesta a diferentes objetivos:

- brindar una experiencia predecible y agradable al usuario final
- lograr una integracion coherente entre modulos
- aumentar cobertura de requerimientos de usuario y sistema
- facilitar la comprension del comportamiento esperado del sistema

Por unidad existen las siguientes validaciones:

- Modulo Digital
 - entradas de configuracion para el PP2, DDS2 y AD
 - entradas de programa ejecutable por el PP2
 - existencia de conexion via interfaz USB durante la ejecucion de experimentos
 - salidas de señales RF
 - salidas de pulsos TTL
 - input de conversion de datos del AD
 - salida de reportes de los canales A y B.
- Servidor
 - autenticacion obligatoria en los servicios
 - ejecucion secuencial de los experimentos
 - aislamiento de espacio de trabajo de usuarios
 - coherencia en los estados de los experimentos
 - generacion de reportes
 - gestion de procesos de ejecucion de experimentos
- Interfaz de usuario web
 - visualizacion coherente de elementos y estilos
 - tiempos cortos de carga de las vistas
 - visualizacion de alertas de usuario
 - transiciones entre vistas

Dados los diferentes enfoques son necesarias se eligieron las siguientes para probar:

- Postman para simular comunicacion http con servicios REST

- Arduino para visualizar los pulsos TTL via Serial Plotter
- Osciloscopio para medir señales RF generadas en los experimentos.
- Generador de señales para simular adquisicion de datos del AD.
- Google Chrome y herramientas para validar estados de la interfaz grafica.
- Mock del servidor escrito en Flask-Python para simular servicios REST.

18 Arquitectura del Modulo Digital

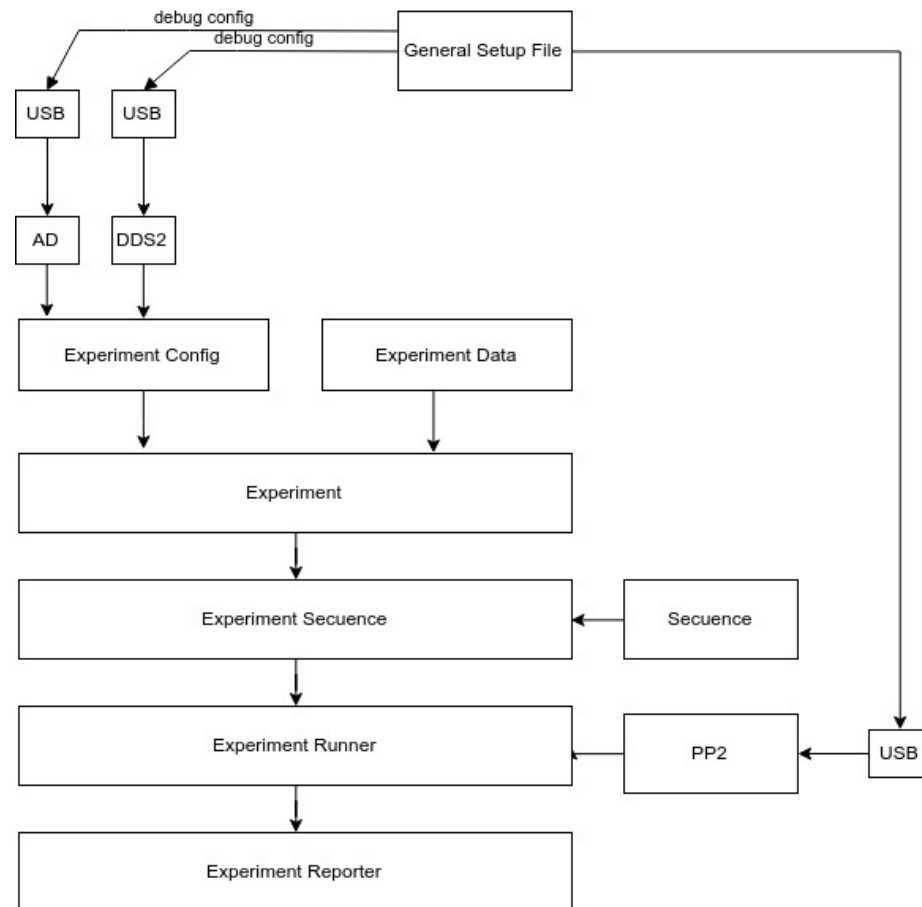


Figure 8: Arquitetura Modulo Digital

19 Arquitectura del Servidor

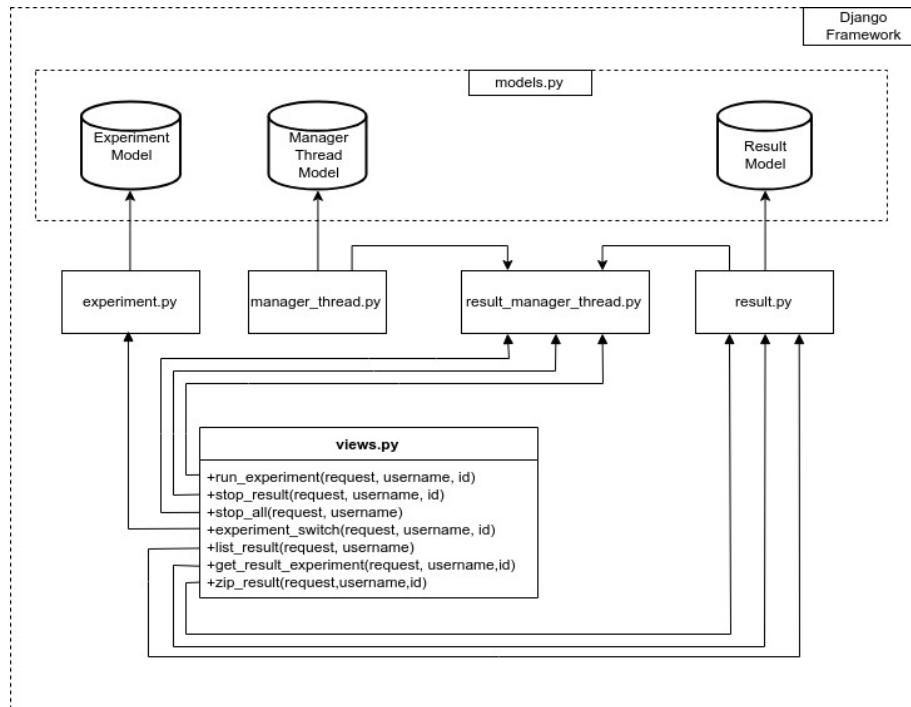


Figure 9: Arquitectura del Servidor

20 Arquitectura de Interfaz Grafica

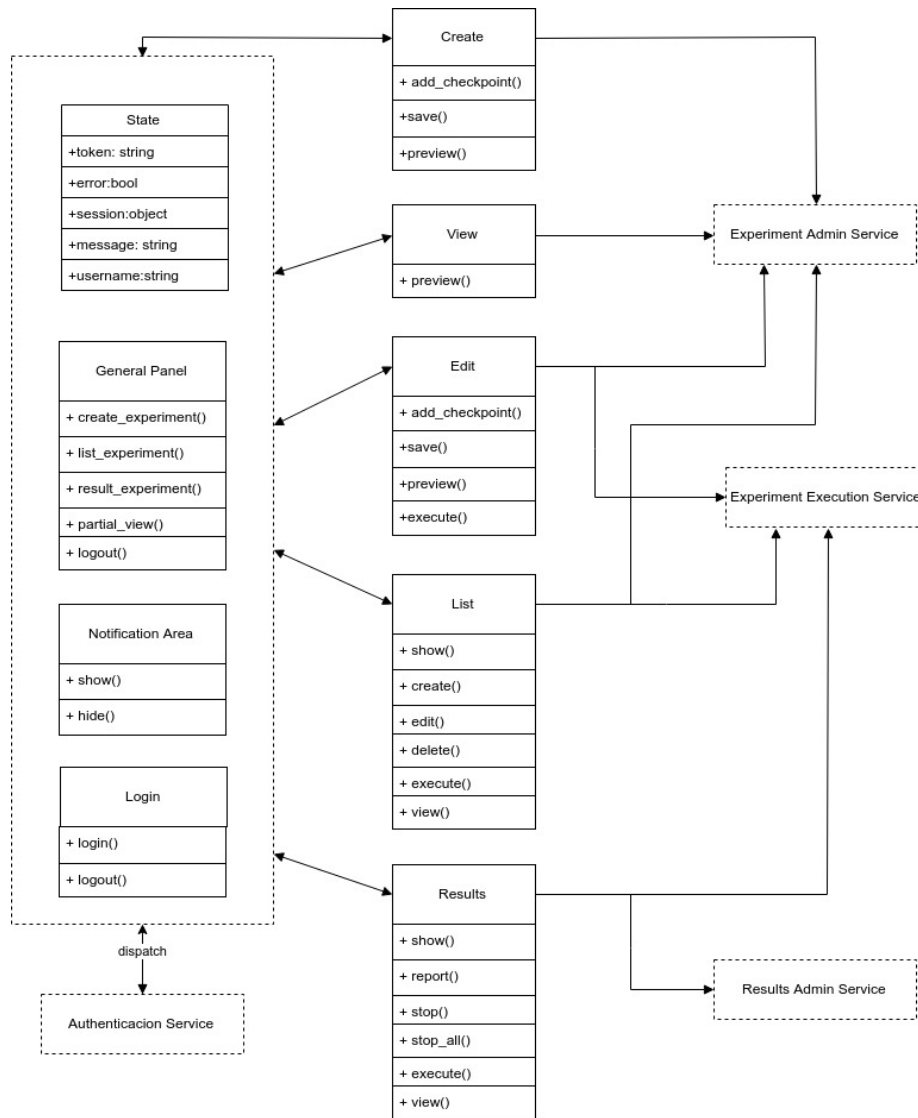


Figure 10: Arquitectura del UI

21 Integracion de los submodulos

La integracion de los modulos del sistema se implementa a diferentes niveles

21.1 Repositorio

Un submodulo git es un repositorio dentro de otro que lo incluye, con su propio historial de cambios. Asi mismo cuando hay un cambio en un submodulo el contenedor lo contemplara siempre y cuando lo agregue al historial de este. El repositorio afectado por esta metodologia es el del Servidor incluyendo al Modulo Digital y a la Interfaz Grafica.

El arbol de directorios del repositorio queda asi:

```
Server
├── build
│   └── static
├── error
├── experiments
│   └── Modulo Digital
├── log
├── login
├── out
├── Interfaz Grafica
└── settings
```

21.2Codigo

El servidor provee el ManagerThread para poder interactuar con el Modulo Digital y con el modelo de la base de datos segun el estado de la ejecucion del experimento en curso.

Para integrar con la Interfaz grafica lo hace de manera estandar via configuracion del servidor que provee seguridad y autenticacion de los servicios via CSRF-Token presente en el archivo principal index.html de la Interfax Grafica.

22 Analisis de Performance

Se propone como analisis de performace un experimento con las siguientes características:

- 500 promedios
- 2 loops con un pulso cada uno
- bloque de 8K
- 4 fases

Los consumos de CPU - RAM - DISCO

23 Casos de Prueba

De acuerdo con los enfoques generales y por unidad se diseñaron los siguientes casos de prueba

23.1 Modulo Digital

- experimento con configuracion invalida del AD
- experimento con configuracion invalida del DDS2
- experimento con secuencia invalida para el PP2
- experimento un pulso largo
- experimento un pulso corto
- experimento con pulsos largos y cortos intercalados
- experimento promedio con un pulso
- experimento promedio con un pulso variable en duracion
- experimento promedio con un pulso variable en fase
- experimento un pulso dentro de un loop
- experimento un pulso dentro de un loop de nivel 4
- experimento con un RETL sin LOOP asociado
- experimento con un LOOP sin RETL asociado
- experimento con un LOOP de nivel 5

23.2 Servidor

- solicitud de servicio sin token de autenticacion
- solicitud de autenticacion con usuario no registrado
- solicitud de autenticacion con usuario registrado pero contraseña invalida
- solicitud de fin de sesion
- solicitud de creacion de un experimento
- solicitud de edicion de un experimento
- solicitud de edicion de un experimento inexistente
- solicitud de edicion de un experimento en ejecucion
- solicitud de eliminacion de un experimento

- solicitud de eliminacion de un experimento inexistente
- solicitud de eliminacion de un experimento en ejecucion
- solicitud de ejecucion de un experimento habiendo uno en ejecucion
- solicitud de ejecucion de un experimento inexistente
- solicitud de ejecucion de un experimento
- solicitud de cancelacion de una ejecucion en curso
- solicitud de reporte durante la ejecucion
- solicitud de reporte finalizada la ejecucion
- solicitud de reporte de un experimento que nunca se ejecuto
- solicitud de reporte de una ejecucion cancelada
- solicitud de reporte de una ejecucion fallida

23.3 Interfaz Grafica

- ingreso al sitio con usuario y contraseña validos
- ingreso al sitio con usuario ó contraseña invalidos
- salida del sitio con boton login
- crear un experimento
- visualizacion de lista de experimentos creados
- filtrado de lista de experimentos creados
- ver un experimento seleccionado de la lista
- edicion de un experimento seleccionado de la lista
- eliminacion de un experimento seleccionado de la lista
- ejecucion de un experimento seleccionado de la lista
- visualizacion de lista de resultados
- ver detalles de un resultado
- visualizacion de menu de gestion de experimentos
- visualizacion de menu de gestion de resultados
- visualizacion de menu de navegacion
- visualizacion de alertas

24 Errores conocidos

- si el sistema es interrumpido por un apagado imprevisto durante una ejecución es posible que un estado sea inconsistente
- en vista edición hacer click en boton "guardar y ejecutar" demora 5 segundos la transición a vista de resultados
- las alertas deben ser cerradas para obtener nuevas si las hubiese
- soporte parcial a diferentes tamaños de pantalla

25 Limitaciones de la plataforma

Existen limitaciones de diversos tipos:

- Plataforma
 - driver y dll para windows posiblemente reemplazable usando libusb
 - administracion de archivos limitada
 - soporte librerias de matematica y graficos limitada
- Modulo Digital
 - requiere la implementacion de un manager para su utilizacion
 - requiere permisos para creacion de archivos de logs y reportes
- Server
 - gestion del proceso del servidor es manual
 - la gestion de usuarios es manual via interfaz Django
 - seguimientos de recursos de sistema es manual
- Interfaz Grafica
 - programacion orientada a componentes
 - JSX una variacion de JavaScript
 - JQuery no es soportado

26 Mejoras a futuro

- Modulo digital
 - interfaces de servicio
 - independencia como programa de consola
 - test automaticos
 - integracion continua
- Server
 - cola de experimentos
 - soporte a otros Modulos Digitales u Aparatos
 - post-procesado de datos en tiempo real
 - test automaticos
 - integracion continua
- Interfaz Grafica
 - soporte a telefonos moviles
 - visualizacion de graficos
 - tests automaticos
 - vista de manipulacion de resultados
 - vista de gestion de perfil
 - tests automaticos
 - integracion continua

27 Analisis de esfuerzo

- Niveles:
 - Muy Alto
 - Alto
 - Medio
 - Bajo
 - Trivial
- Modulo Digital
 - Usb: Alto
 - PP2: Alto
 - AD: Alto
 - DDS2: Muy Alto
 - Reporte: Bajo
 - Hilo: Alto
- Server:
 - Servicios: Alto
 - Login: Alto
 - Integracion: Muy Alto
 - Modelado: Medio
 - Manager Hilos: Alto
- Interfaz Grafica:
 - Vistas: Alto
 - Integracion: Alto
 - Alertas: Alto

References

- [1] JSON notation

