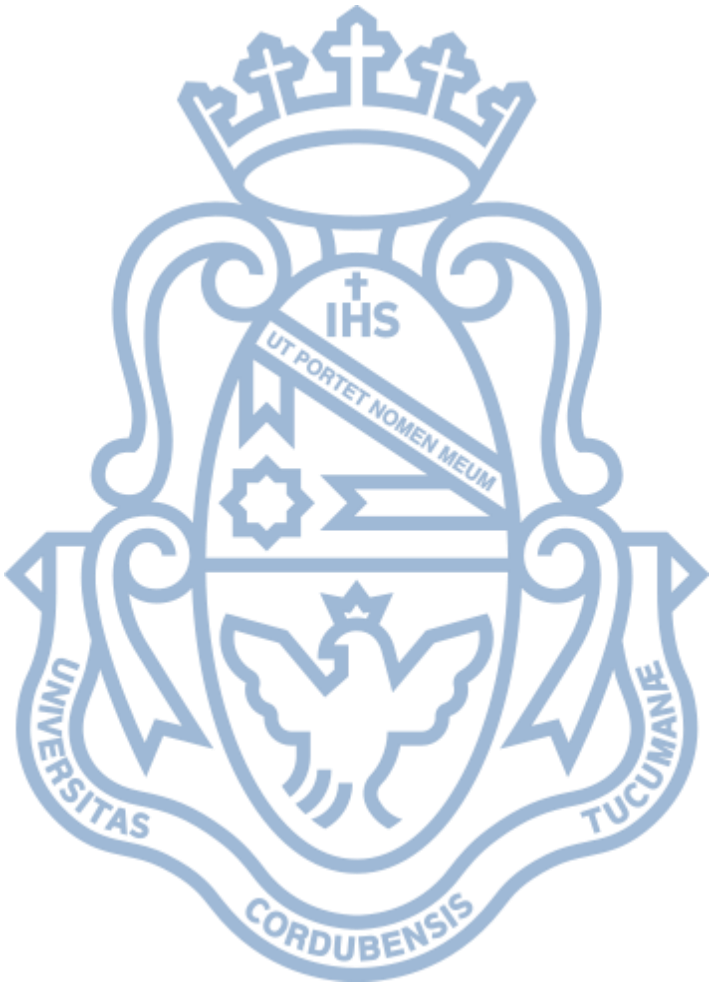
Sistema Disipador de Temperatura



Bustos Maximiliano
mxbustos@gmail.com
Colombo Agustín
agedcolombo@hotmail.com

Facultad de
Ciencias Exactas
Físicas y Naturales

Contenido

Contenido 2
1 Introducción. 3
1.1 Objetivos. 3
1.2 Sobre la aplicación. 3
1.2.1 Definición. 3
1.2.2 Marco Teórico. 3
1.2.3 Disipación de calor por renovación 3
1.2.4 Entorno del Sistema 4
1.3 Sobre el modelo de desarrollo. 4
1.4 Descripción del resto del documento. 5
2 Nota de entrega. 6
2.1 Estado 6
2.2 Bugs Conocidos 6
3 Requerimientos. 7
3.1 Diagrama preliminar de Arquitectura: 7
3.2 Casos de Uso: Diagrama general de casos de uso 8
3.3 Requerimientos Funcionales 9
3.3.1 Matriz de trazabilidad 9
3.4 Requerimientos No Funcionales 10
3.4.1 Escalabilidad 10
3.4.2 Facilidad de uso y entendimiento 10
3.4.3 Respuesta 10
4 Implementación del Sistema 11
4.1 Elementos utilizados 11
4.2 Diagrama arquitectura del Sistema 11
4.3 Cálculo de Resistencias 12
4.4 Periféricos de la Placa LPCxpresso 12
4.4.1 ADC 13
4.4.2 UART 13
4.4.3 TIMER1 13
4.4.4 SYSTICK 13
4.4.5 EINT 2 y 3 14
4.4.6 Interfaz gráfica 14
5 Testing 15
5.1 Test Unitario 15
5.1.1 Matriz Requerimiento – Test Unitario 15
5.2 Test de Sistema 16
5.2.1 Test de sensor 16
5.2.2 Test de disipación 17
6 Conclusión 18
7 Bibliografía 19

1 Introducción.

1.1 Objetivos.

Con el presente informe se pretende documentar todas las fases del proceso de desarrollo de un sistema de software y hardware. El objetivo general es el de realizar, desde el comienzo, todas las actividades referidas a la ingeniería necesarias para la creación eficiente y organizada de un producto de sistema embebido. Para ello se elaborará, a partir de una idea existente, un nuevo sistema que agregue otras funcionalidades que requerirán el uso de distintos patrones de diseño, el patrón de arquitectura y distintas prácticas, tales como la implementación de casos de test, control de configuraciones y documentación en general.

1.2 Sobre la aplicación.

1.2.1 Definición.

El sistema que se quiere construir llevará el nombre de "Sistema Disipador de Temperatura". Es un sistema destinado al ámbito académico, cuyos usuarios potenciales podrían ser:

- Profesores
- Alumnos
- Desarrolladores

Todos ligados de alguna manera a la enseñanza o aprendizaje de distintos patrones de diseño y formulación de documentación para un producto completo.

1.2.2 Marco Teórico.

Unos de los problemas más comunes que presentan los circuitos electrónicos que están encapsulados en algún contendor de algún material en particular, como es el aluminio, chapa, plásticos entre otros, es el de la alta temperatura que queda encerrada en dicho contenedor. Esto se debe al efecto Joule, el cual establece que si en un conductor circula corriente eléctrica la parte de la energía cinética se transforma en calor, en muchos de los casos ese fenómeno de aumento de calor puede ocasionar problemas en cuanto al funcionamiento y vida útil del circuito.

Frente a esta problemática, surge la necesidad de realizar un circuito el cual sea capaz de poder aumentar más la disipación de calor en el entorno que se encuentre el circuito en cuestión. El mismo deberá poder prender un ventilador que produzca un flujo de e intercambio de aire para disminuir la temperatura dentro del contenedor. A su vez también está la necesidad del usuario poder establecer una temperatura máxima, para que no afecte en nada al circuito electrónico.

1.2.3 Disipación de calor por renovación

El flujo de calor (Q) se refiere a la cantidad de calor que atraviesa una superficie por unidad de tiempo, que en el SI se mide en Vatios por metro cuadrado (W/m^2).

$P = Q \times S$ (W o Kcal/h)

Cuando un caudal C de aire se introduce a una temperatura Te (Temperatura externa) diferente a la del aire interior Ti (Temperatura interna) expulsado se produce una disipación de calor P, que se puede determinar por la siguiente expresión:

$P = C \times D \times \gamma_a \times (Te - Ti)$ (Kcal/h)

Siendo: P = Potencia o flujo de calor disipado (Kcal/h)

C = Caudal (m 3/h)

D = Densidad del aire, 1.2 Kg/m3 aproximadamente

γ_a = Calor específico del aire = 0.24 Kcal/Kg ºC

T = Temperatura seca (ºC)

1.2.4 Entorno del Sistema

El sistema que se implementa se utilizara en la ciudad de Córdoba, en donde la temperatura ambiente es de aproximadamente entre 24 ºC a 28 ºC en verano y baja a 15 ºC en invierno. El sensor se coloca en un recipiente de forma rectangular cuyas medidas son 30 cm de largo por 8 cm de ancho y su altura es de 8 cm, obteniendo un volumen de 1920 cm^3. Las temperaturas que soporta variaran en un rango de 0 ºC a 99 ºC, y se podrá setear temperaturas limite también entre los 0 ºC y los 99 ºC.

1.3 Sobre el modelo de desarrollo.

Para la creación del sistema se decidió utilizar el modelo de desarrollo conocido como "modelo en V". El mismo representa, en forma de V, las relaciones temporales entre las distintas fases del ciclo de desarrollo de un proyecto (lo cual está ilustrado en la Figura 1). El modelo elegido sugiere que para cada fase de desarrollo existe una fase paralela de verificación o validación, obedeciendo al principio de que para cada fase debe existir un resultado objetivamente verificable. La relación temporal que hay entre cada etapa y su correspondiente verificación se ve reflejada en la distancia que las separa.

Se optó por dicho modelo primeramente porque se trata de un proyecto pequeño, donde el grupo de trabajo es reducido (facilitando la comunicación), no habría muchos requerimientos ni riesgos, tampoco existe el rol activo de un "cliente", ni se vería en la necesidad de realizar prototipos ni versiones previas a la entrega formal del producto. Lo que se intenta, entonces, es de cumplir con cada etapa y no pasar a la siguiente sólo hasta que se concluya la presente (incluyendo los Plan Test generados en cada fase).

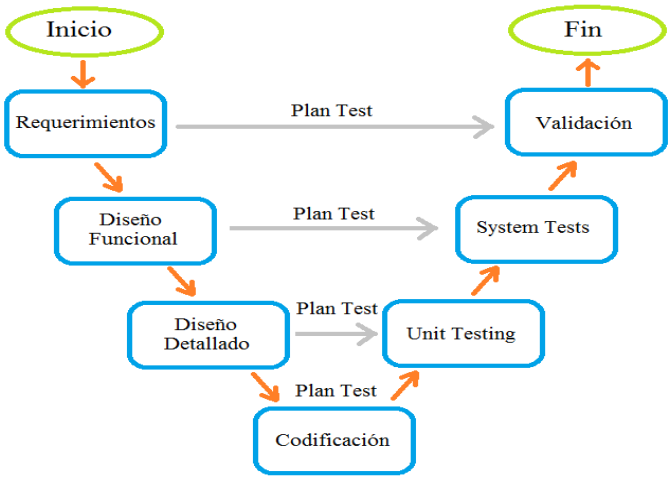


Figura 1.

Para implementar el modelo se decidió, al comienzo de cada etapa, donde se planearían las actividades y discutirían los temas concernientes, con el fin de dividir tareas y definir el modo de proceder. Durante cada fase, se usarían otras vías de comunicación para ponerse de acuerdo en demás detalles y evacuar las posibles dudas que puedan surgir. Para dar por finalizada una etapa (idealmente) se debería tener como resultado la documentación, los casos de test o validación, código fuente y demás archivos, según corresponda.

1.4 Descripción del resto del documento.

La información expuesta en el documento será organizada en las siguientes secciones:

- Requerimientos
- Arquitectura
- Diseño
- Test
- Información adicional
- Conclusión
- Bibliografía

Donde cada apartado es responsable de tareas específicas y son componentes de las distintas fases de desarrollo del producto.

2 Nota de entrega.

2.1 Estado

La presente nota pretende explicar el estado del entregable al momento del realease. Se proporciona un archivo comprimido en formato ".zip" conteniendo la documentación del trabajo, el código fuente con su respectivo esquema de directorio y el archivo de la interfaz gráfica.

El código fuente está compuesto por un main.c, configuración.c, defines.h y demás archivos que se incorporan para el uso de la arquitectura ARM.

2.2 Bugs Conocidos

Luego de presionar un botón del teclado, hay un pequeño delay entre esa acción y que aparece el numero representado en el display. Esto está hecho para asegurar que el número mostrado es el correcto. Solucionarlo llevaría más tiempo.

3 Requerimientos.

Esta sección describe los requerimientos del Sistema. El mismo está destinado a toda persona que esté avocada al desarrollo del sistema en cuestión como así también a sus pruebas de funcionamiento, correcciones de errores y evolución del mismo. Se espera que sirva como material de consulta y apoyo para mantener una línea de desarrollo acorde a las funcionalidades requeridas por el cliente, brindando información básica referente a los patrones de diseño y arquitectura utilizados y de esta forma lograr que la herramienta cumpla con el propósito para el que fue desarrollada.

Se describirá primeramente la arquitectura del sistema, luego se observan los casos de uso, a continuación los requerimientos funcionales y por último los requerimientos no funcionales.

Un requisito funcional define una función del sistema de software o sus componentes. Una función es descrita como un conjunto de entradas, comportamientos y salidas. Los requisitos funcionales pueden ser: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que se supone, un sistema debe cumplir. Los requerimientos de comportamiento para cada requerimiento funcional se muestran en los casos de uso. Son complementados por los requisitos no funcionales, que se enfocan en cambio en el diseño o la implementación.

3.1 Diagrama preliminar de Arquitectura:

En la Figura 4 se puede observar claramente los elementos que componen al sistema. Debido a que el propósito final del sistema es servir al aprendizaje de dicho patrón y demás patrones de diseño, es que se elige implementar esta forma de arquitectura. De esta manera, el usuario interactuaría con la interfaz y con el sistema en sí mediante el teclado. La salida del sistema se verá en los displays y en la interfaz gráfica en la computadora.

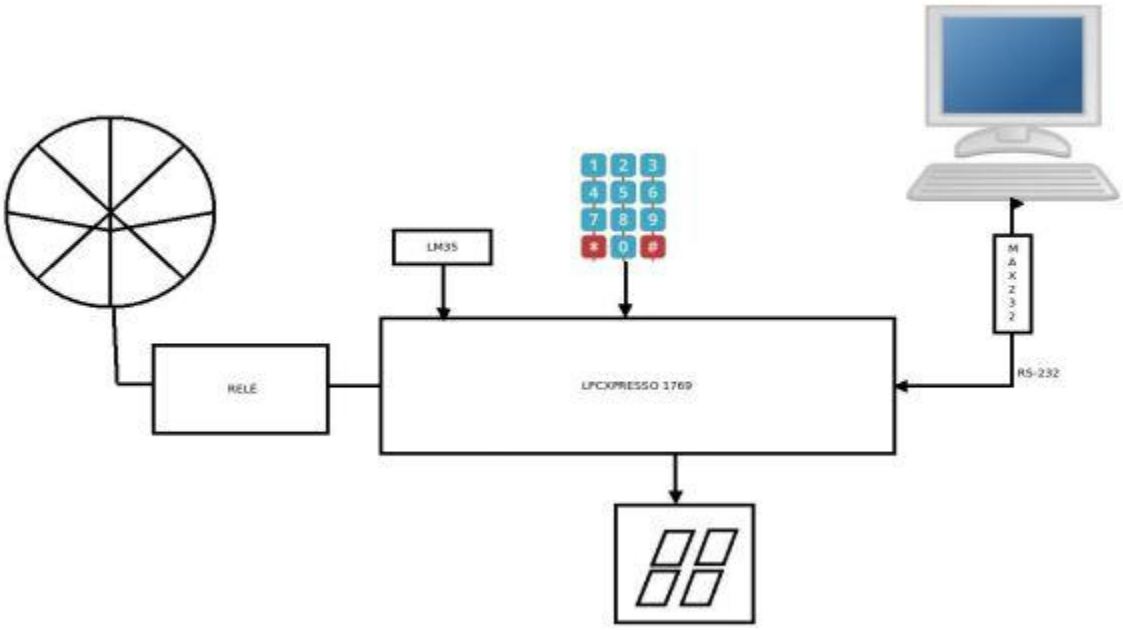
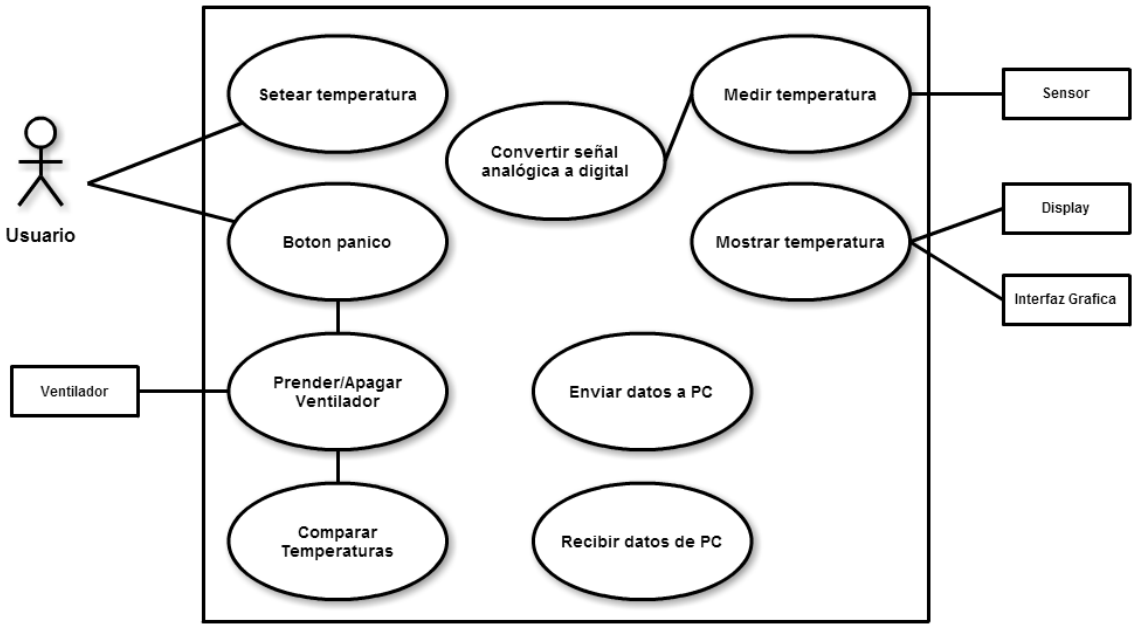


Figura 4.

3.2 Casos de Uso: Diagrama general de casos de uso

Un caso de uso es una descripción de los pasos o las actividades que deberán realizarse para llevar a cabo algún proceso. Los personajes o entidades que participarán en un caso de uso se denominan actores. En el contexto de ingeniería del software, un caso de uso es una secuencia de interacciones que se desarrollarán entre un sistema y sus actores en respuesta a un evento que inicia un actor principal sobre el propio sistema. Los diagramas de casos de uso sirven para especificar la comunicación y el comportamiento de un sistema mediante su interacción con los usuarios y/u otros sistemas.

Se observa que el sistema es una arquitectura y corresponde a un solo dispositivo. Y a su vez se consideran un actor, el usuario propiamente dicho y varias entidades que interaccionan con el sistema, como son los displays, la interfaz gráfica y el ventilador.



Lista de casos de Usos:

- CU01: Setear temperatura
- CU02: Medir Temperatura
- CU03: Convertir señal analógica a digital
- CU04: Botón Pánico
- CU05: Mostrar temperatura
- CU06: Prender/Apagar Ventilador
- CU07: Comparara Temperatura
- CU08: Enviar datos a PC
- CU09: Recibir datos de PC

3.3 Requerimientos Funcionales

A continuación se listan los requerimientos funcionales de alto nivel. A cada requerimiento, como podrá observarse, le fue asignado un identificador único, el que será parte de la matriz de trazabilidad definida en la sección 2.3, dándonos una referencia al mismo para mapear en ella cada requerimiento con sus correspondientes casos de test.

LPC=Etiqueta que hace referencia a la placa usada.

1. [LPC1] El sistema debe medir la temperatura mediante el sensor, que estará en el rango máximo de 0 a 99 grados centígrados.
2. [LPC2] Los números de las temperaturas (medida y seteada) se muestran en dos display de 7 segmentos: en el de la izquierda se verá el valor de la decena y en el de la derecha el de la unidad.
3. [LPC3] El sistema convierte la señal analógica del sensor en una señal digital mediante el módulo ADC.
4. [LPC4] El sistema puede recibir por parte del usuario dos números que se ingresa por teclado y toman el nombre de la temperatura seteada, que puede estar entre 0 y 99 grados centígrados.
5. [LPC5] El sistema compara las temperaturas sensada y seteada. Si la sensada es mayor a la seteada se debe prender un ventilador.
6. [LPC6] El tiempo necesario para disminuir 4 grados (por encima de la temperatura ambiente) es de 10 segundos.
7. [LPC7] El sistema cuenta con un pulsador que permite alternar entre modos (Setear temperatura limite / Mostrar Sensor)
8. [LPC8] Si el sistema se encuentra en el modo Setear Temperatura muestra por los display dicha temperatura. En el modo Mostrar Sensor, muestra la temperatura sensada.
9. [LPC9] En el encendido del sistema arrancara midiendo y mostrando la temperatura sensada (modo Mostrar Sensor).
10. [LPC10] El sistema se comunica con la computadora mediante el protocolo RS-232. Utilizando el módulo UART.
11. [LPC11] Se cuenta con una interfaz gráfica que muestra la temperatura sensada, seteada y el botón de pánico.
12. [LPC12] La interfaz cuenta con un botón Bi-estado que al activa/desactiva el Modo Pánico.
13. [LPC13] El sistema apaga el ventilador si el Modo Pánico esta desactivado y la temperatura disminuye tres grados por debajo de la temperatura seteada.

3.3.1 Matriz de trazabilidad

La matriz de trazabilidad muestra la relación entre los casos de uso definidos anteriormente con los diferentes requerimientos funcionales:

	CU 01	CU 02	CU 03	CU 04	CU 05	CU 06	CU 07	CU08	CU09
[LPC1]		x							
[LPC2]					x				
[LPC3]			x						
[LPC4]	x								
[LPC5]						x	x		
[LPC6]		x							
[LPC7]	x								
[LPC8]					x				
[LPC9]			x		x				
[LPC10]								x	x
[LPC11]					x				
[LPC12]				x					
[LPC13]						x	x		

3.4 Requerimientos No Funcionales

Esta sección no va a incluir una lista extensa de requerimientos No funcionales, puesto que entendemos que los RNF apuntan a características que terminan siendo restricciones del sistema, y queremos que nuestro sistema sea extensible, razón por la cual mientras menos restricciones impongan su desarrollo, uso y mantenimiento, mayor será el valor de este. Sin embargo, creemos que los siguientes requerimientos son esenciales para el proyecto:

- Escalabilidad
- Facilidad de uso y entendimiento
- Respuesta

Los alcances de lo anterior y sus métodos de prueba se detallan a continuación:

3.4.1 Escalabilidad

Se entiende que la escalabilidad es la relativa facilidad que tiene el sistema para incorporar características nuevas y reutilizar el código existente. Esto aplica a la arquitectura definida hasta el momento y se define como de la capacidad de incorporar funcionalidades o múltiples dispositivos de entrada/salida.

3.4.2 Facilidad de uso y entendimiento

Este requerimiento está apuntado principalmente al código que deberá ser entendible y comentado para quienes deberán mantener el sistema. Como así debe ser entendible el uso de los pulsadores.

3.4.3 Respuesta

Esto se aplica principalmente al tiempo en el que el sistema interprete la interrupción por medio de los pulsadores y accione en efecto de acuerdo a sus funciones.

4 Implementación del Sistema

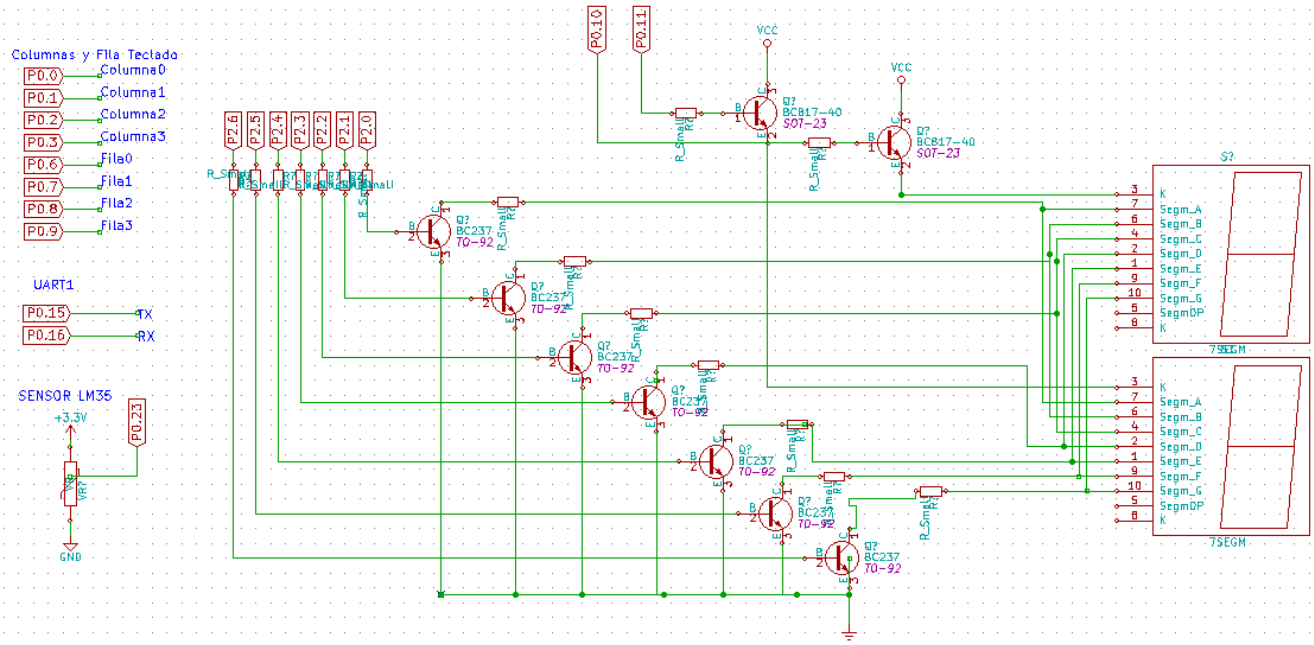
Esta sección del documento estará orientada más bien a los desarrolladores y técnicos de testing debido a que se desarrollara de manera técnica los instrumentos utilizados, el código realizado (no en su totalidad pero sus funciones más importantes).

4.1 Elementos utilizados

A continuación se listan todos los componentes que el sistema necesita:

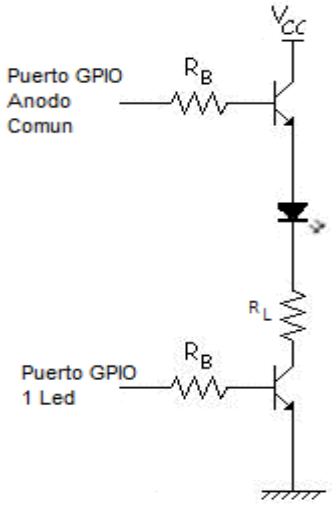
1. Placa de desarrollo LPCxpresso
2. Dos display 7 segmentos
3. Teclado matricial 4x4
4. Pulsador
5. Sensor LM35
6. Transistores
7. Resistencias
8. Cables
9. Protoboard
10. Dispositivo USB a RS-232
11. Relé
12. Ventilador

4.2 Diagrama arquitectura del Sistema



4.3 Cálculo de Resistencias

A continuación se desarrollará la parte dura del sistema, el cual para poder encender los display el puerto necesitará manejar cuidadosamente la corriente. La siguiente figura muestra la conexión básica de los display con los puertos de la placa LPCxpresso:



Lo primero que se deberá calcular es la resistencia que se coloca para controlar la corriente que pasará por el led del display cuando este se encuentre prendido. El cálculo es el siguiente:

$$Vcc - Vsat - Vled - Vsat - IL*RL = 0$$

$$RL = \frac{Vcc - Vsat - Vled - Vsat}{IL}$$

$$RL = \frac{3.3 - 0.2 - 1.5 - 0.2}{0.01} = 140\ \Omega$$

Luego se calcula la Resistencia que limitará a la corriente que ingresa al Puerto GPIO de la placa:

$$Vcc - Vsat - Vdiodo - Ib*Rb = 0$$

$$Rb = \frac{Vcc - Vdiodo - Vsat}{(Ic/\beta)}$$

$$Rb = \frac{3.3 - 0.7 - 0.4}{0.01} = 5000\ \Omega$$

Luego de los valores calculados se deberá aproximar a valores comerciales, quedará para este sistema Rb=5100Ω y RL= 220Ω. Se recomienda poner resistencias de mayor valor al estándar.

4.4 Periféricos de la Placa LPCxpresso

Se explicara a continuación como se usaron los diferentes periféricos de la placa para conformar al sistema como un conjunto.

4.4.1 ADC

El conversor analógico a digital se lo utiliza para obtener los valores que el sensor LM35 en el pin llamado AOUT, para ello el adc lleva la siguiente configuración:

```
LPC_SC->PCONP |=  ADC_POWERON ;
LPC_PINCON->PINSEL1 |= SELECT_ADC0;
LPC_PINCON->PINMODE1 |= PUERTO_ANALOGICO;
LPC_ADC->ADCR|= OPERATIONAL_ADC | DIV_VEL | MODE_BURTS |START_ADC;
NVIC_EnableIRQ(ADC_IRQn);
```

Donde la primera línea se enciende el módulo adc, luego se selecciona el canal 0, y el puerto que se utiliza se lo debe configurar analógicamente. A su vez la velocidad de trabajo del adc es la menor que se puede configurar ya que la temperatura varía muy lentamente. Este módulo una vez obtenido un dato interrumpe al microprocesador para entregar el valor.

4.4.2 UART

La unidad de transmisión que servirá para la comunicación entre la placa LPCxpresso y la computadora es la UART1, que se configura con una velocidad de transmisión de 9600 baudios, y se utiliza para enviar las temperaturas sensadas, saeteadas y a su vez si el usuario presiona el botón de pánico encenderá o apagara el ventilador según el caso.

```
LPC_SC->PCONP|=1<<25;     //Encendemos el periférico
LPC_UART1->LCR|=11;       // 8 bits
LPC_UART1->LCR|= (1<<7); //Activamos BaudRate
LPC_UART1->DLL=0b10100001; //DLL= 161 -> 9585 baudios -> error de 14 ->0.1458%
LPC_UART1->DLM=0;
LPC_UART1->LCR &= 0b01111111;
LPC_UART1->IER=1; // Corresponde a la interrupción por Receive Data Available
NVIC_EnableIRQ(UART1_IRQn);
LPC_PINCON->PINSEL0 |=1<<30; //TXD3
LPC_PINCON->PINSEL1 |=1<<0; //RXD3
```

4.4.3 TIMER1

El Timer 1 se configura para actualizar el valor de la temperatura sensada cada medio segundo aproximadamente. Esto se debe a que muchas veces el sensor queda fluctuando en un valor y se observa una ráfaga de números en los display. Cuando el timer interrumpe actualiza la variable tempSen con el valor que almacena la variable tempAux, la cual es obtenida mediante la función moda, que devuelve el valor mas esperado.

4.4.4 SYSTICK

Este módulo es utilizado para refrescar los valores de los displays, ya sean correspondiente a la temperatura sensada o a la temperatura seteada. Este módulo interrumpe cada 10 milisegundos

```
LPC_GPIO2->FIOCLR0 |= (0xFF);
LPC_GPIO0->FIOCLR = 1<<10;
LPC_GPIO0->FIOCLR = 1<<11;
if (selector==0){   //Variable selector=0 muestra Unidad, sino muestra decena
             LPC_GPIO0->FIOSET = 1<<11;//Prende puerto P0[11]
             if(habTec) LPC_GPIO2->FIOPIN0 = segmentos[tempSet/10];
```

```
            else       LPC_GPIO2->FIOPIN0 = segmentos[tempSen/10];
            selector++;
}else{                                          //Decena
            LPC_GPIO0->FIOSET = 1<<10;//Prende puerto P0[10]
            if(habTec) LPC_GPIO2->FIOPIN0 = segmentos[tempSet%10];
            else         LPC_GPIO2->FIOPIN0 = segmentos[tempSen%10];
            selector=0;
```

4.4.5 EINT 2 y 3

La interrupción externa 2 se utiliza para configurar y habilitar al teclado matricial. Una vez que se ingresa a dicha interrupción se configuran los puertos GPIO como interrupciones y se habilita la EINT3, que donde las interrupciones de los puertos gpio están unidas. Se muestra a continuación el código del handler de la EINT2:

```
LPC_GPIOINT->IO0IntClr |= ((1 << 3)|(1 << 2)|(1 << 1)|(1 << 0));
LPC_GPIO0->FIOCLR |= ((1 << 3)|(1 << 2)|(1 << 1)|(1 << 0));
if(habTec==0){
      LPC_GPIOINT->IO0IntEnF |= ((1 << 3)|(1 << 2)|(1 << 1)|(1 << 0));
      NVIC_EnableIRQ(EINT3_IRQn);
      habTec=1;
}else{
      LPC_GPIOINT->IO0IntEnF &= ~((1 << 3)|(1 << 2)|(1 << 1)|(1 << 0));
      NVIC_DisableIRQ(EINT3_IRQn);
      habTec=0;
      tempTemp= tempSet;
}
Delay(250);
LPC_GPIOINT->IO0IntClr |= ((1 << 3)|(1 << 2)|(1 << 1)|(1 << 0));
LPC_SC->EXTINT|=(1<<2);
```

Luego la interrupción externa 3 se utiliza para colocar las filas y columnas del teclado matricial y poder obtener los números de la temperatura seteada.

4.4.6 Interfaz gráfica

La interfaz gráfica que corre en un computador, se la programó en el leguaje Python, la cual se encarga de mostrar de manera legible los datos de las temperaturas sensada y seteada. A su vez tiene un botón tooogle que prende o apaga el router, denominado botón de pánico.

5 Testing

5.1 Test Unitario

En esta sección se identifican los casos de testing para cada uno de los requerimientos funcionales definidos en la sección anterior. Como se observa, en algunos casos no se tiene un unit test para validar un requerimiento funcional, ya que se hará en la próxima sección con el Test de Sistema. Al igual que a los requerimientos funcionales, a los casos de testing se les ha asignado un identificador para luego realizar el mapeo en la matriz de trazabilidad.

La mayoría de los tests propuestos son "pruebas de campo". La única forma que conocemos de verificar la medición de una magnitud física, es midiéndola varias veces con aparatos distintos, estando uno de estos homologados. Basándonos en principio como este, están pensados los siguientes casos de test.

TLPC=Etiqueta que hace referencia al test de la placa usada.

1. [TLPC1] Se ejecutará un código que mostrará solamente valores de números de 0 a 99 para comprobar visualmente la luminosidad de los leds de los displays.
2. [TLPC2] Se ejecuta un código que contiene una función de configuración de ADC y devuelve verdadero un valor.
3. [TLPC3] Se ejecuta un código que mostrara por pantalla los números ingresados por el usuario desde el teclado matricial.
4. [TLPC4] Se colocara en el handler de la interrupción externa 2 una impresión por pantalla que dirá: "Modo Seteando Temp", así se podrá comprobar que se encuentra en dicho modo.
5. [TLPC5] Se ejecutara un código que configura la uart1, y mediante una consola como gtkterm, en una computadora con un sistema operativo como Ubuntu, se pueda recibir números que se incrementen en un segundo.
6. [TLPC6] Se ejecutara un código que logra enviar un UNO al puerto GPIO0 30 que encenderá un led, para simular el botón de pánico que posteriormente será el propio ventilador.
7. [TLPC7] Se ejecutara un código que logra enviar un CERO al puerto GPIO0 30 que apagará un led, para simular el botón de pánico que posteriormente será el propio ventilador.

5.1.1 Matriz Requerimiento – Test Unitario

A continuación se especifica la matriz que mapea los requerimientos funcionales con los casos de testing de acuerdo a los identificadores asignados anteriormente.

	[TLPC1]	[TLPC2]	[TLPC3]	[TLPC4]	[TLPC5]	[TLPC6]	[TLPC7]
[LPC1]							
[LPC2]	X		X				
[LPC3]		X					
[LPC4]			X				
[LPC5]							
[LPC6]							
[LPC7]				X			
[LPC8]							
[LPC9]							
[LPC10]					X		
[LPC11]							
[LPC12]						X	
[LPC13]							X

5.2 Test de Sistema

Los test de sistemas pueden realizarse de dos formas distintas, Una llamada “Big Bang” y la otra se denomina “Incremental”. Se utiliza la primera de ellas para testear el sistema ya que desde un principio se logra medir a todo el sistema en sí, en cambio con el método incremental se parte desde las unidades del sistema y al final se puede testear el sistema por completo.

A continuación se desarrolla el test de sistema:

5.2.1 Test de sensor

Se utiliza un termómetro patrón y se toma en distintas horas del día la temperatura con el instrumento patrón y el sistema en funcionamiento, completando los campos de la siguiente planilla:

Día	Hora	Valor Termómetro	Valor Sist.	Diferencia
09/11/2015	07:00	23 ºC	22 ºC	1 ºC
09/11/2015	11:00	26 ºC	25 ºC	1 ºC
09/11/2015	15:00	27 ºC	26 ºC	1 ºC
09/11/2015	19:00	26 ºC	25 ºC	1 ºC
09/11/2015	23:00	26 ºC	25 ºC	1 ºC
10/11/2015	07:00	22 ºC	23 ºC	1 ºC
10/11/2015	11:00	25 ºC	24 ºC	1 ºC
10/11/2015	15:00	27 ºC	26 ºC	1 ºC
10/11/2015	19:00	27 ºC	26 ºC	1 ºC
10/11/2015	23:00	26 ºC	25 ºC	1 ºC

Como se observa la diferencia es de 1 ºC, lo cual implica que nuestro sistema está midiendo por debajo de la temperatura que arroja el instrumento patrón.

Para solucionar este error se vuelve a calibrar el sensor y aumentándose en una unidad cada vez que se presenta la temperatura sensada por pantalla.

La línea de código antes del test era:

```
valorADC = (valorADC * 1.22);
```

Luego del test de sistema la línea varia de la siguiente forma:

```
valorADC = (valorADC * 1.22)+1;
```

Además se considera el error que se tiene en el módulo del ADC ya que no se toman todos los bits de resolución del mismo, sino los 8 bits más significativos.

5.2.2 Test de disipación

Este simple test consiste en subir la temperatura del sensor y luego tomar el tiempo que tarda el sistema en disminuir la temperatura.

El procedimiento es el siguiente:

a. Se mide la temperatura ambiente. Es importante destacar ya que el sistema nunca podrá disminuir por debajo de esta temperatura.

b. Se setea la temperatura limite en un valor alto (por ejemplo 88ºC)

c. Se calienta el aire cerca del sensor (mediante alguna fuente de calor externa) hasta que marque 5 grados más que la temperatura ambiente.

d. Se setea la temperatura limite en un valor que sea al menos Temperatura ambiente + 4. En este momento el ventilador se enciende.

e. Se toma el tiempo que tarda el sistema en enfriar hasta que se apaga el ventilador

Día	Hora	Temperatura ambiente (ºC)	Temperatura seteada (ºC)	Temperatura medida (ºC)	Tiempo de enfriamiento (segundos)
09/11/2015	07:00	23	27	28	4.5
09/11/2015	11:00	26	30	31	5
09/11/2015	15:00	27	31	32	5.2
09/11/2015	19:00	26	30	31	6.5
09/11/2015	23:00	26	30	31	5.5
10/11/2015	07:00	22	26	27	4.7
10/11/2015	11:00	25	29	30	5.1
10/11/2015	15:00	27	31	32	5.9
10/11/2015	19:00	27	31	32	7.1
10/11/2015	23:00	26	30	31	5.6

Se puede observar que no hay una relación directa entre la temperatura y el tiempo necesario para enfriar. Depende también de la humedad y otros factores atmosféricos, pero de todas formas, observando la tabla, se ve que todavía hay un margen aceptable con respecto al requerimiento planteado [LPC6], ante cualquier eventualidad que pudiera demorar el enfriamiento unos segundos más.

6 Conclusión

Se espera ratificar las expectativas del cliente mediante este trabajo, el cual se desarrollara en un ambiente académico, y poder a su vez demostrar los conocimientos en forma práctica y teórica basados en elementos y ciencia de la ingeniería, los cuales serán usados para la solución de algún problema específico, cumpliendo con la finalidad que tienen en su defectos los sistemas embebidos.

Luego se puede validar el producto debido que se lograron realizar los distintos test ya sean unitarios y de sistema en sí. Se debe tener en cuenta que existen errores de hardware ya que se utilizan para la elaboración del producto materiales como las Protoboard y los distintos cables que no terminan siendo eficientes a la hora de la implementación.

7 Bibliografía

- User Manual 10360 LPCxpresso
- Manual de Programación del Lenguaje C
- http://anhnvnguyen.blogspot.com.ar/2010/04/lpc17xx-gpio-basic_05.html
- http://www.eeherald.com/section/design-guide/esmod6a.html
- Filminas Clases Electrónica Digital III – UNC
- Datasheet LM35