

**Universidad Nacional De Educación A Distancia**



**Tesis Doctoral**

**Prototipo Para La Determinación De Propiedades  
Térmicas En Alimentos Semisólidos**

**Por: Rudi Radrigán Ewoldt**

**Ing. Agroindustrial Mg.Scie.**

**Tesis propuesta como cumplimiento parcial de  
los requisitos para el doctorado en:**

**Sistemas De Ingeniería Eléctrica, Electrónica y De Control**

**De la  
Escuela Técnica Superior de Ingenieros Industriales**

**Madrid -2007**



Departamento de Ingeniería Eléctrica,  
Electrónica y de Control

## **TESIS DOCTORAL**

Prototipo para la Determinación de Propiedades  
Térmicas en Alimentos Semisólidos

Por:

Rudi Radrigán Ewoldt

Ing. Agroindustrial Mg.Scie.

Director:

Dr. D. Francisco Mur Pérez.





## Agradecimientos

Deseo expresar el más sincero agradecimiento a los profesores Mur y Castro por su colaboración en el desarrollo del tema presentado. Además, manifiesto mi agradecimiento especial a la Agencia de Cooperación Internacional de España, quien financio por medio de la beca Mutis IIB, los estudios preliminares de este trabajo, así como a las empresas Maxim y Microchip quienes aportaron con suministros electrónicos especiales para el desarrollo del prototipo. Gracias también Alvaro Gajardo por sus valiosos comentarios y aportaciones.

Un especial agradecimiento a la Sra. Nora Plaza C. Decano de la Facultad de Ciencias de la Salud y los Alimentos de la Universidad del Bío Bío, por su incondicional apoyo y confianza.

Y a todas las personas que de una u otra forma aportaron con sus comentarios a la culminación de éste trabajo.



## Dedicatoria

El presente trabajo esta dedicado a mi familia, mi esposa Cecilia y mis pequeños, quienes me apoyaron con su paciencia para completar con éxito una etapa más en vida.

Y a la memoria de mi padre, que no pudo estar de cuerpo presente para culminar éste camino.

*Cotidianus faceren vita nostrum*



**ÍNDICE GENERAL**

Lista de Tablas.....	ix
Lista de Figuras.....	xi
Lista de Ecuaciones.....	xvii
Resumen.....	xix
Abstrac.....	xxi
Capítulo I: .....	1
Introducción .....	1
Descripción del proyecto.....	1
Base teórica .....	3
Transferencia de calor por conducción.....	4
Ley de Fourier.....	7
Transferencia de calor por conducción.....	4
Conducción de calor en estado estable unidimensional.....	10
Balances diferenciales: placa.....	11
Placa plana sin generación en estado estable.....	14
Transferencia en la interfase.....	15
Efecto convectivo .....	15
Radiación .....	16
Manantial calórico Eléctrico.....	17
Manantial calórico Viscoso.....	18
Manantial calórico Químico .....	20
Manantial calórico Nuclear.....	21

## **Índice**

Técnicas de medición.....	21
Método Estado Estacionario .....	22
Método Cilindros concéntricos .....	22
Método Flujo de calor .....	23
Técnicas en estado casi estacionario .....	25
Método de Fitch.....	25
Método Cenco-Fitch.....	26
Método modificado de Zuritz .....	26
Método Fitch-Rahman .....	27
Difusividad térmica.....	29
Técnicas de medición.....	29
Determinación directa .....	30
Calor específico .....	30
Método de la mezcla .....	31
Método comparativo .....	31
Método adiabático.....	31
Método Calorimetría Diferencial de Barrido .....	32
Capítulo II: El Hardware de control.....	33
Introducción .....	33
Adquisición de datos.....	33
Microcontroladores.....	34
Arquitectura de los Pic .....	35
Arquitectura Harvard.....	37
Diagrama de bloques .....	38

## **Índice**

---

Mapas de memoria RAM .....	39
Memoria de programa.....	41
Registro de funciones.....	42
Contador de Programa .....	44
Stack.....	45
Palabra de Estado del Procesador .....	47
Otros registros especiales .....	48
Puertos de entrada/salida.....	49
Circuito equivalente.....	49
Temporizador/Contador.....	51
Interrupciones.....	53
Funcionamiento.....	54
Fuentes .....	55
Programa Fuente .....	55
Conjunto de Interrupciones.....	57
Instrucciones de Byte.....	58
Instrucciones de Byte sobre W y Literales.....	63
Instrucciones de Bit.....	65
Instruccines especiales .....	71
Resume de instrucciones .....	73
Métodos de direccionamineto.....	74
Direcciomaniento de memoria .....	76
Ordenador.....	78
Protocolo de comunicaciones.....	79

## **Índice**

---

Protocolo USB .....	79
Características USB .....	79
Especificaciones técnicas.....	81
Topología del bus.....	81
Capa física .....	84
El Host USB.....	85
El controlador del Host.....	87
Dispositivo USB.....	89
HUBs.....	91
Arquitectura Externa .....	92
Arquitectura Interna.....	94
La capa lógica.....	98
Relación Software cliente-función.....	100
Flujo de datos USB .....	101
Endpoints y direcciones de dispositivo .....	101
Tuberías.....	102
Mensajes .....	103
Frames y microframes .....	104
Tipos de Transferencias.....	105
Transferencias de control.....	106
Transferencias Isócronas.....	107
Interrupciones.....	107
Transferencias de Bultos .....	108
Capa protocolo .....	109



## **Índice**

---

Formatos de campos e identificadores .....	112
Campo dirección .....	112
Campo endpoint.....	113
Campo frame .....	113
Campo datos.....	113
CRC.....	113
Paquetes Token .....	114
Paquetes SOF.....	115
Paquetes Datos .....	115
Paquetes Handshake .....	116
Transacción OUT.....	118
Transacción SETUP .....	119
Transacción Split .....	119
Formato Split .....	120
Protocolo y transferencias.....	121
Transferencia de Bulk .....	122
Transferencias de control .....	123
Transferencias de interrupciones.....	124
Transferencias isócronas.....	124
Identificación de la velocidad del dispositivo.....	125
Codificación de datos.....	126
Relleno de bits.....	127
Sync.....	128
EOP .....	128

## **Índice**

---

La mecánica.....	129
Cable estándar de quita y pon .....	130
Cable fijo de velocidad alta y media.....	131
Protocolo USART.....	132
Primer Bloque.....	135
Segundo Bloque.....	136
Tercer Bloque .....	137
Protocolo I2C .....	139
Interrupción I2C .....	140
Concepto del Bus I2C .....	140
Generalidades .....	142
Transferencia de Bit .....	143
Interrupción del I2C .....	144
Conexiones Start y Stop .....	145
Transferencia de datos.....	146
Reconocimiento.....	147
Sincronización .....	148
Arbitraje.....	149
Diseño de la Placa .....	151
Placa Usb.....	152
Placa Sensores.....	155
Termopares.....	157
¿Cómo funcionan? .....	158
Temperatura relativa .....	159

## **Índice**

Como llegar a la respuesta .....	162
Codigos de color de los termopares.....	166
Voltímetro .....	170
Diagrama final.....	171
Capítulo III: Software de control .....	173
Microcontrolador y sensores.....	173
Software de control e interfase .....	173
Requerimientos del sistema.....	173
Requerimientos no funcionales.....	174
Casos de Uso.....	175
Casos de Uso Usuario-Aplicación .....	175
Casos de Uso Aplicación-Instrumento .....	188
Diagrama casos de uso Operador Aplicación .....	190
Diagrama de secuencia.....	193
Implementación.....	204
Capítulo IV: Pruebas y Ensayos .....	205
Pruebas y ensayos.....	205
Ingreso de datos .....	205
Captura de datos.....	206
Análisis de datos .....	206
Impresión de reportes.....	207
Almacenamiento de datos .....	209
Comparación de resultados.....	214
Capítulo V: Conclusiones.....	215

## **Índice**

---

Capítulo VI: Aportaciones .....	219
Glosario.....	225
Bibliografía.....	227
Apéndice 1:Frimware.....	233
Apéndice 2: Aspectos de Visual Basic 6.....	294
Apéndice 3: Programa de instalación.....	CD

**LISTA DE TABLAS**

<i>Número</i>	<i>Página</i>
Resumen de instrucciones .....	73
Paquetes de clases.....	110
2.1 Transacción de paquetes .....	117
2.2 Respuesta del Host.....	118
2.3 Comportamiento del endpoit .....	119
2.5 Tipos de endpoint.....	121
2.6 Codificación del cable USB.....	132
2.7 Terminología del Bus I2C .....	141
2.8 Estado de transferencia de datos.....	146
2.9 Coeficiente de seebeck.....	163
2.10 Datos técnicos de referencia de las termocuplas.....	165
3.1 Requerimientos funcionales.....	174
Ficha 1 Adquirir Datos .....	176
Ficha 2 Agregar materia prima.....	176
Ficha 3 Agregar Usuario .....	177
Ficha 4 Análisis del filtro .....	178
Ficha 5 Análisis lineal.....	178
Ficha 6 Análisis del termograma.....	179
Ficha 7 Análisis termopar.....	179
Ficha 8 calibración del instrumento .....	180
Ficha 9 Cerrar sesión.....	180

## **Índice**

---

Ficha 10 Eliminar Materia prima.....	181
Ficha 11 Eliminar Usuario .....	181
Ficha 12 Informe Log File .....	182
Ficha 13 Informe materias primas.....	182
Ficha 14 Informe resumen de datos .....	183
Ficha 15 Informe Termograma.....	183
Ficha 16 Informe Termopar.....	184
Ficha 17 Informe Usuarios .....	185
Ficha 18 Iniciar sesión .....	185
Ficha 19 Modificar materia prima.....	186
Ficha 20 Modificar Usuario.....	187
Ficha A1 Cerrar conexión del instrumento.....	188
Ficha A2 Transmisión de datos.....	188
Ficha A3 Verificar conexión del instrumento.....	189
Ficha A4 Verificar estado del instrumento.....	190
4.1 Comparación de resultados con otras metodologías .....	214

**LISTA DE FIGURAS**

<i>Número</i>	<i>Página</i>
1. Etapas del proyecto.....	2
1.1a Distribución de caras .....	5
1.1b Distribución de temperaturas para t .....	5
1.2 Gradiente de temperatura con histéresis .....	7
1.3 Gradiente de temperatura.....	7
1.4 Flujo de calor en eje z.....	13
1.5 Espectro parcial.....	17
1.6 Perfil de velocidad en cilindros.....	19
1.7 Manantial nuclear .....	21
2.1 Arquitectura Von Newmann .....	36
2.2 Arquitectura Harvard.....	38
2.3 Microcontroladores pic16f8xx.....	39
2.4 Organización interna RAM.....	40
2.5 Organización memoria familia pic16cxx .....	41
2.6a Estructura del microprocesador .....	42
2.6b Estructura de un pic .....	42
2.7 Circuito equivalente de I/O.....	49
2.8 Diagrama lógico PIC16F8X.....	50
2.9 Circuito equivalente RTCC.....	52
2.10 Lógica de interrupciones PIC16F8X .....	54
2.11 Direccionamiento GOTO .....	76

## **Índice**

---

2.12 Direccionamiento Call.....	77
2.13 Estructura de capas del USB .....	83
2.14 Direccionamiento del Flujo .....	84
2.15 Topología física del bus USB.....	85
2.16 Vista conceptual del controlador USB .....	88
2.17 Típico HUB USB.....	91
2.18 Arquitectura externa USB .....	93
2.19 Esquema del HUB.....	94
2.20 Sistema de conexión .....	96
2.21 Esquema de funcionamiento del HOST.....	98
2.22 Lógica del HOST .....	99
2.23 Diagrama de conexión cliente función.....	100
2.24 Transacción SPLIT .....	120
2.25 Transferencia en bulk.....	122
2.26 Transferencia de control .....	123
2.27 Transferencias de interrupciones .....	124
2.28 Transferencias isócronas .....	125
2.29 Esquema eléctrico de conexión por velocidad de transmisión.....	126
2.30 Codificación cero no invertido.....	127
2.31 Sincronismo NRZI.....	128
2.32 Detalles de conectores AyB.....	130
2.33 Cable USB A-B .....	131
2.34 Cable tipo A.....	132
2.35 Esquema de conexión RS232.....	135



## **Índice**

2.36 Conector DB25 y 9 para DTE.....	138
2.37 Conector DB25 y 9 para DCE.....	139
2.38 Esquema I2C.....	141
2.39 Transferencia I2C .....	144
2.40 Condiciones Start y Stop.....	145
2.41 Reconocimiento .....	147
2.42 Diagrama de Bloques PL2330.....	152
2.43 Esquema de Conexión USB-RS232.....	154
2.44 Esquema de conexión del pic con termopares.....	155
2.45 Esquema de conexión Pic comercial.....	156
2.46 Esquema pic SX48BD/TQ.....	157
2.47 a Termopar lineal.....	159
2.47 b Termopar de dos hilos.....	159
2.48 Termopar normal.....	161
2.49 Termopar com bloque isométrico .....	161
2.50 Termopar tipo K.....	162
2.51 Termopar digital tipo K .....	170
2.52 Diagrama final placa sensores.....	171
3.1 Diagrama de casos de uso operador aplicación.....	191
3.2 Diagrama casos de uso Administrador aplicación.....	192
3.3 Diagrama casos de uso Aplicación instrumento.....	192
3.4 Diagrama de secuencia para el inicio de sesión .....	193
3.5 Diagrama de secuencia para el cierre de sesión .....	194
3.6 Diagrama de secuencia para agregar materias primas.....	194

## **Índice**

3.7 Diagrama de secuencia para eliminar materias primas .....	195
3.8 Diagrama de secuencia para modificar materias primas.....	195
3.9 Diagrama de secuencia para adquirir datos.....	196
3.10 Diagrama de secuencia para análisis de filtro .....	197
3.11 Diagrama de secuencia para análisis lineal.....	197
3.12 Diagrama de secuencia para análisis de termopar.....	198
3.13 Diagrama de secuencia para análisis de termograma.....	198
3.14 Diagrama de secuencia para imprimir resumen .....	199
3.15 Diagrama de secuencia para imprimir termograma .....	199
3.16 Diagrama de secuencia para imprimir termopar(gráfico).....	200
3.17 Diagrama de secuencia para agregar usuario(Administrador).....	200
3.18 Diagrama de secuencia para eliminar un usuario .....	201
3.19 Diagrama de secuencia para modificar usuario .....	201
3.20 Diagrama de secuencia para informe de usuarios.....	202
3.21 Diagrama de secuencia para informe Log File .....	202
3.22 Diagrama de Clases.....	203
4.1 Ingreso de datos.....	205
4.2 Captura de datos .....	206
4.3 Interfase de trabajo .....	207
4.4 a Informe Gráfico.....	208
4.4 b Análisis de regresión .....	208
4.5 Resumen de propiedades térmicas.....	208
4.6 Termograma puré de manzana a 60 seg. ....	209
4.7 Termograma puré de pera a 60 seg. ....	210

## **Índice**

---

4.8 Termograma puré de pera a 120 seg.....	210
4.9 Termograma teórico .....	211
4.10 A Microscopia electrónica de harina de trigo .....	211
4.10 B Microscopia electrónica de lechuga.....	211
4.11 A Microscopia de pulpa de durazno.....	212
4.11 B Microscopia de galleta .....	212
4.12 Red proteica vegetal.....	212
4.13 Formación de vórtice de Koch .....	213



**LISTA DE ECUACIONES**

<i>Número</i>	<i>Página</i>
1.1 Flujo de calor .....	6
1.2 Velocidad de flujo .....	8
1.3 Ley de Fourier .....	9
1.4 Vector densidad de flujo .....	9
1.5 Velocidad Vectorial.....	9
1.6 De fourier análoga a Ohm .....	10
1.7 Balance general.....	12
1.8 Volumen de elemento.....	14
1.9 Distribución de temperatura en placas .....	14
1.10 Flujo de calor a través de paredes .....	15
1.11 Flujo de calor en la superficie .....	16
1.12 Manantial eléctrico .....	18
1.13 Perfil de velocidad.....	19
1.14 Manantial catalítico .....	20
1.15 Manantial nuclear .....	21
1.16 Calor unidimensional.....	22
1.17 Conductividad radial.....	23
1.18 Flujo global .....	24
1.19 Coeficiente térmico .....	24
1.20 Coeficiente individual .....	24
1.21 Conductividad térmica efectiva .....	24

## **Índice**

---

1.22 Conductividad Efectiva.....	24
1.23 Calor en estado casi estacionario.....	25
1.24 De Fitch .....	25
1.25 De Fitch-Rahman .....	28
1.26 De Fuente lineal.....	28
1.27 Difusividad Térmica .....	29
1.28 Número de Biot .....	30
1.29 Entalpía .....	30
1.30 Calor específico por mezcla.....	31
1.31 Calor específico adiabático.....	32

## RESUMEN

Los sistemas productivos en la actualidad han evolucionado cada día y la ingeniería de procesos debe ser más activa en lo que al control de procesos lo exige un mundo globalizado, ya que permite optimizar las materias primas y las fuentes energéticas necesarias para producir un alimento.

Las propiedades térmicas de alimentos son fundamentales para la industria agroalimentaria, permiten o dificultan la transferencia térmica entre la fuente calefactora y el alimento en si; la característica principal de anisotropismo de la fibra vegetal dificulta el modelamiento o predicción del comportamiento de dicho producto por tanto se han trabajado con cierto error asociado.

El prototipo planteado en éste estudio demostró ser mucho más eficiente en tiempo y recursos de materias primas además de mantener su replicabilidad en 96.5% con respecto a las metodologías clásicas en las que se apoya este trabajo.

Por otra parte la incorporación de la formación de vórtices de Koch (meteorología de los fractales), así como es estudio completo de los tres fenómenos asociados a la transmisión de calor dio como resultado una simulación en dos dimensiones de las distintas distribuciones térmicas al interior de un cilindro infinito, que comparadas con las microscopías electrónicas de barrido muestran una consistencia irrefutable de los postulados utilizados.

Para la recolección de datos se utilizó una fuente lineal de calor y se construyó la adquisición de datos por medio del protocolo USB para las distintas zonas de temperatura con termopares digitales (DS2760) tipo K, que a su vez conectaban al pic central SX48DB/TQ con protocolo RS232 al pic PL2330 SSOP que transforma el UART a USB 1.1 y este a la interfase de trabajo desarrollada en Visual Basic 6 ®, dicha interfase permite la captura y el análisis de los datos recolectados.

El prototipo final mostró gran versatilidad en la captura y análisis de los datos, pudiendo ser utilizado en docencia como en investigación, abriendo una ventana para una futura comercialización como instrumento de análisis en los distintos laboratorios de propiedades térmicas de alimentos o en líneas de procesos industriales.



## **ABSTRACT**

*At the present time, the productive systems have been evolving every day and the process engineering should be more active about the process control as the globalize world demands it, since it allows optimizing the raw materials and the power sources necessary to produce a food.*

*The food's thermal properties are fundamental for the agro-alimentary industry. This thermal properties allows or makes difficult the thermal transference between the heating source and food it self; the main characteristic aniseed tropism of the vegetal fiber makes difficult the behavioral modeling or the prediction of such product, therefore has been worked with certain associate error:*

*The proposed prototype shows to be much more efficient in the matter of time and resources of raw materials, moreover; the prototype's replication is around the 96.5% in every respect from the classics methodologies in which this work has been supported.*

*On the other hand, the incorporation of the Koch's vortexes (fractals meteorology), as well of the full study of the three associated phenomena in the heat transfer; gave as result a two dimensional simulation of the different thermal distributions at the inside of a infinite cylinder that compared with the electronic sweeping microcopies is irrefutable consistent than the used postulates.*

***For the data collection it was used a linear heat source. The data acquisition it was realized through/by mean the USB protocol for the distinct temperatures zones with K type digitals thermocouples (DS2760), that were connect to a central PIC SX48DB/TQ with RS232 protocol. The PIC SX48DB/TQ was connected to a PL2330 SSOP PIC that converts de UART signal to USB 1.1. The USB 1.1 signal is captured for software interface that was developed with Microsoft Visual Basic 6.0®. The interface makes possible the collection and the analysis of the collected data.***

***The final prototype shows great versatility in the capture and the data analysis, being able to be used in teaching as in investigation, opening a window for a future commercialization as instrument of analysis on the different laboratories of food's thermal properties or in industrial process lines.***

## ***Capítulo 1***

### **1 INTRODUCCIÓN.**

En la actualidad los procesos de manufactura de productos alimenticios, los procesos de transferencia térmica han sido un verdadero desafío para el ingeniero de procesos, supeditado a las experiencias previas, es decir a la aplicación de estos tratamientos por tanteo o aproximación subjetiva; este problema radica que los materiales con que se fabrican estos productos son diversos y heterogéneos en sus propiedades, si además se incorpora la poca información existente hace que los procesos de fabricación tengan mermas considerables por defectos; pocas empresas de manufacturas poseen la tecnología y los recursos suficientes para analizar previamente las materias primas. Por ello es imperativo contar con un instrumento que permita subsanar el inconveniente tecnológico actual.

#### **1.1 Descripción del proyecto.**

El objetivo de este proyecto es construir el sistema lógico y físico para la medición de temperatura en alimentos independiente del sentido de las fibras vegetales. La meta es determinar la velocidad de conducción térmica al interior de un alimento ( $k$ ), difusividad térmica ( $\alpha$ ) y calor específico ( $C_p$ ), para la mejora de procesos tecnológicos.

El centro del proyecto lo constituye el diseño y construcción del firmware y la aplicación que interactúa con el usuario.

El sistema debe cumplir con ciertas características acordes con el uso que se la va a dar.

- Sencillo con el mínimo de componentes básicos.
- Robustez y estabilidad del sistema.
- Versatilidad de muestreo

Las diferentes partes del proyecto serán las correspondientes a las de la siguiente gráfica.

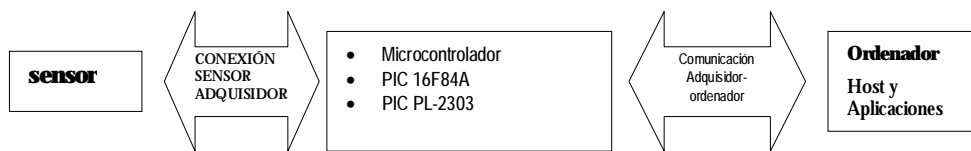


Figura 1 Etapas del proyecto

- Sensores.
- Conexión sensor-adquisidor.
- Adquisidor de datos
- Comunicación adquisidor-ordenador
- Aplicaciones en el ordenador

También se puede dividir el proyecto en etapas tales como:

- Aplicación.
- Comunicación USB.
- Comunicación UART.

## **1.2 BASE TEÓRICA.**

### **Introducción a la transferencia de calor.**

De los procesos de transporte, el transporte de calor es probablemente el más familiar dado que es parte de nuestra experiencia diaria, por ejemplo cuando se nos enfría la sopa o el café. Procesos que emplean transporte de calor aparecen frecuentemente en la industria química: Calentamiento del petróleo crudo (u otra mezcla líquida) hasta su punto de ebullición para separarlo en fracciones en una columna de destilación o la remoción del calor generado en una reacción química [26]. En cualquier caso necesitamos hallar la velocidad a la cual ocurre la transferencia de calor para calcular el tamaño del equipo requerido o para mejorar el ya existente.

De otra parte debemos recordar que el calor es solo una de las formas de la energía y que es esta y no el calor la que se conserva de acuerdo a la primera ley de la termodinámica [4]. La energía como propiedad se utiliza en termodinámica para ayudar a especificar el estado de un sistema. De otra parte la energía se transfiere a través de los límites de un sistema termodinámico en forma de trabajo o de calor. Transferencia de calor es la expresión usada para indicar el transporte de energía originado en una diferencia de temperatura. Crank [6] menciona que la "Velocidad de Transferencia de Calor" o "Flujo de Calor" ( $Q$ , [W] o [Btu/h]), es la expresión de la energía térmica transportada por unidad de tiempo, y "Densidad de Flujo de Calor" o "Flux de Calor" ( $q$ , [W/m<sup>2</sup>] o [Btu/hr.pie<sup>2</sup>]), es la velocidad de transferencia de calor por unidad de área. El cálculo de las velocidades locales de

transferencia de calor requiere conocer las distribuciones locales de temperatura, las cuales proveen el potencial para la transferencia de calor.

Existen tres mecanismos diferentes por los cuales ocurre esta transferencia de calor:

- i. Conducción, en donde el calor pasa a través de la sustancia misma del cuerpo.
- ii. Convección, en el cual el calor es transferido por el movimiento relativo de partes del cuerpo calentado, y
- iii. Radiación, mecanismo por el que el calor se transfiere directamente entre partes distantes del cuerpo por radiación electromagnética.

En gases y líquidos la convección y la radiación tienen importancia destacada, pero en los sólidos la convección puede considerarse ausente y la radiación generalmente es despreciable.

### **1.2. 1 Transferencia de calor por conducción.**

La teoría matemática de la conducción del calor según Thomson, puede basarse en una hipótesis sugerida por el siguiente experimento: Tomemos una placa de algún sólido limitada por dos superficies planas paralelas de una extensión tal que, desde el punto de vista de las partes entre los dos planos, puedan suponerse infinitos [37] (ver figura 1.1<sup>a</sup>).

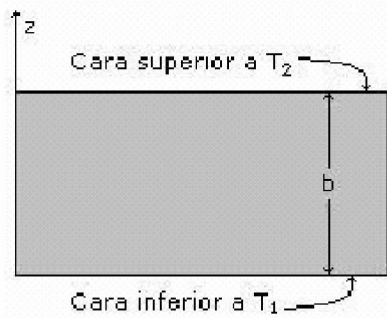


Fig. 1.1a

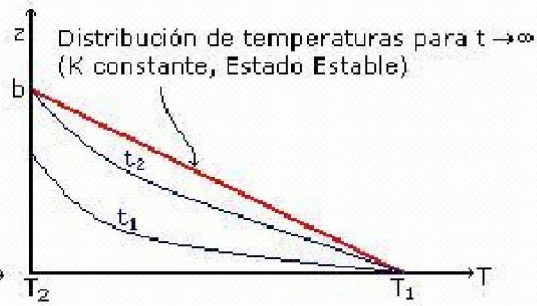


Fig. 1.1b

En la práctica esta condición puede acercarse usando una placa plana, de dimensiones finitas donde sus caras menores han sido aisladas térmicamente de forma tal que solo existan gradientes de temperatura en la dirección perpendicular a las caras mayores. En este caso la diferencia de temperatura ocurre entre planos perpendiculares al eje  $z$  causando transporte en la dirección  $z$ . El hecho de que la placa es muy delgada en la dirección  $z$  y muy ancha en las direcciones  $x$  e  $y$  indica que hay pérdidas despreciables en los extremos perpendiculares a los ejes  $x$  e  $y$ . De esta forma  $q_x$  y  $q_y$  son cero. En general la velocidad de conducción de calor en cualquier punto en un material se caracteriza por un vector de flux de calor  $q$  el cual puede resolverse en componentes a lo largo de los tres ejes coordenados.

Podemos ignorar la naturaleza vectorial de  $q$  y considerar solo su componente escalar  $z$  para un simple caso de conducción unidimensional de calor.

Los dos planos se mantienen a temperaturas diferentes sin que esta diferencia de temperaturas sea tan grande como para causar un cambio sensible en las propiedades del sólido. Por ejemplo, mientras la superficie superior se mantiene a la temperatura de una mezcla hielo agua, la inferior se mantiene a la

temperatura de una corriente de agua caliente que fluye constantemente por allí. Después de mantener estas condiciones durante suficiente tiempo, las temperaturas de los diferentes puntos del sólido alcanzaran valores estables, la temperatura siendo igual para planos paralelos a la superficie de la placa (despreciando los efectos terminales).

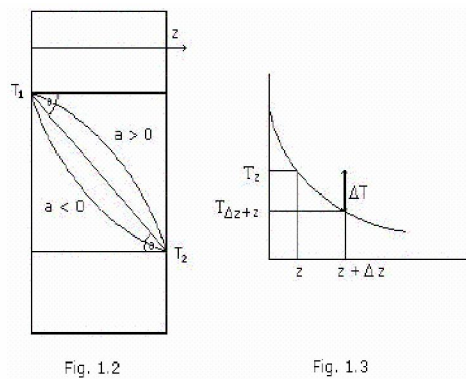
Supongamos que la temperatura de la superficie inferior es  $T_1$  y la de la superficie superior es  $T_2$  ( $T_1 > T_2$ ), y consideremos que el sólido está inicialmente a temperatura uniforme  $T_2$ . La placa tiene un espesor  $b$ . Los resultados de los experimentos sugieren que, cuando se ha alcanzado el estado estable, la cantidad de calor que fluye a través de la placa en un tiempo  $t$  a través de un área  $S_z$  perpendicular a la dirección  $z$  es igual a:

$$Q = \frac{k S_z (T_1 - T_2)}{b} \quad (1.1)$$

El coeficiente de proporcionalidad  $k$  es la conductividad térmica. Estrictamente hablando la conductividad térmica no es una constante sino que, de hecho, es una función de la temperatura para todas las fases y en líquidos y gases depende también de la presión, especialmente cerca al estado crítico. La conductividad térmica en la madera y cristales varía también en forma ostensible con la dirección. Esta es una de las Propiedades de Transporte de los materiales. La dependencia de la conductividad térmica con la temperatura para rangos de temperatura pequeños puede expresarse en forma aceptable como  $k = k_0(1 + aT)$ , donde  $k_0$  es el valor de la conductividad térmica en alguna condición de referencia



y  $a$  es el coeficiente de la temperatura que es positivo o negativo dependiendo del material en cuestión. La figura 1.2 muestra el efecto en el gradiente de temperatura (para estado estable) en una placa plan como resultado que este sea positivo o negativo. Se resalta el que el gradiente de temperatura será lineal solo cuando la conductividad térmica sea constante.



### 1.2.2 Ley de Fourier.

En la sección anterior se considera el caso especial de conducción de calor unidimensional en estado estable en una geometría rectangular. La ecuación (1.1) es válida sólo para este caso especial y no puede usarse en otras situaciones tales como geometría cilíndrica o estado transitorio. Tampoco puede usarse para predecir la variación de la temperatura con la posición dentro del medio. Por esta razón es necesario desarrollar una ecuación más general que sea aplicable en cualquier punto, en cualquier geometría y para condiciones estables o inestables (cuando el estado físico de un sistema no cambia con el tiempo, se dice que el sistema se encuentra en estado estable). Con este propósito retomamos del gráfico

1.1b una línea de temperatura contra posición en cualquier momento arbitrario (ver figura 1.3) [9].

Se puede relacionar la velocidad de flujo de calor  $Q_z$  en cualquier posición arbitraria  $z$  a la densidad de flujo de calor  $q_z$  en la misma posición usando la definición  $Q_z = q_z S_z$ .

Comencemos por reconocer que la velocidad de flujo de calor puede escribirse a partir de la ecuación (1.1) como:

$$\frac{Q_z}{S_z} = \frac{k(T_1 - T_2)}{b} = q_z \quad 1.2)$$

Si aplicamos (1.2) a un pequeño incremento  $\Delta z$ ,  $b$  será reemplazado por  $\Delta z$ , y  $(T_1 - T_2)$  por  $-\Delta T$ . El signo menos es necesario de acuerdo a la definición del operador diferencia:

$$\Delta T = T_{z+\Delta z} - T_z$$

Entonces el flujo promedio de calor a través de una distancia  $\Delta z$  es:

$$q_z = k \frac{\Delta T}{\Delta z} = k \frac{T_{(z+\Delta z, T)} - T_{(z, T)}}{\Delta z}$$

De la figura 1.3 se observa que  $-\Delta T / \Delta z$  representa la pendiente promedia sobre la región  $\Delta z$  de la curva  $T$  vs  $z$ . También observamos que si hacemos  $\Delta z$  cada vez más pequeño obtenemos una mejor aproximación de la pendiente en  $z$ . En el límite cuando  $\Delta z$  tiende a cero, obtenemos la derivada parcial de  $T$  respecto a  $z$  según el teorema fundamental del cálculo. Así, para estado transitorio, podemos escribir en cualquier localización:

$$q_z = -k \frac{\partial T}{\partial z} = \frac{Q_z}{S_z} \quad (1.3)$$

La cual es llamada ley de Fourier para conducción de calor en una dimensión, en honor al matemático francés Jean Baptiste Fourier a quien se le atribuye. En el caso de tratarse de estado estable en una dimensión, T sería solo función de z y la derivada sería total.

En el caso general, donde hay flujo de calor en las tres direcciones coordenadas, T es función de más de una variable independiente y:

$$q_x = -k \left( \frac{\partial T}{\partial x} \right) \quad q_y = -k \left( \frac{\partial T}{\partial y} \right) \quad q_z = -k \left( \frac{\partial T}{\partial z} \right)$$

serán las componentes del vector densidad de flujo de calor.

$$\mathbf{q} = i q_x + j q_y + k q_z \quad \text{o} \quad \mathbf{q} = -K \nabla T \quad (1.4)$$

Aquí q es una cantidad vectorial. También i, j, k son los vectores unitarios en las direcciones x, y, z. El operador  $\nabla$  (nabla) puede operar sobre cualquier escalar. Usando T como ejemplo, el término  $\nabla$  es:

$$\nabla T = i \left( \frac{\partial T}{\partial x} \right) + j \left( \frac{\partial T}{\partial y} \right) + k \left( \frac{\partial T}{\partial z} \right) \quad (1.5)$$

La ecuación (1.4) es una ecuación para la ley de Fourier en notación vectorial Gibbs o forma vectorial. Es válida para cualquier sistema isotrópico, o sea que la conductividad es la misma independientemente de la dirección. El signo

menos indica que el calor solo se transfiere en la dirección en la que decrece la temperatura como lo predice la segunda ley de la termodinámica [6].

Es interesante hacer notar que la ecuación de Fourier para conducción unidireccional de calor es exactamente análoga a la ley de Ohm para un conductor eléctrico, la cual puede expresarse como:

$$i = -k_e S \frac{\partial E}{\partial n} \quad (1.6)$$

En esta ecuación la corriente eléctrica  $i$  corresponde al flujo de calor  $Q$ ; el potencial eléctrico  $E$  corresponde al potencial térmico  $T$ , y la conductividad eléctrica  $k_e$  ( $k_e = 1/\rho_e$ , donde  $\rho_e$  es la resistividad eléctrica) corresponde a la conductividad térmica  $k$ . Como las ecuaciones (1.3) y (1.6) tienen la misma forma, el campo de temperatura dentro del cuerpo calentado, y el campo de potencial eléctrico en un cuerpo de la misma forma, corresponden no al otro siempre que la distribución de temperatura en la superficie corresponda a la distribución superficial del potencial eléctrico. Esta analogía nos capacita para estudiar problemas de conducción de calor en detalle a través de modelos eléctricos similares [5].

### **1.2.3 Conducción de calor en estado estable unidimensional.**

Podemos examinar en primera instancia las aplicaciones de la ley de Fourier para conducción de calor, calculando el flujo de calor en algunos sistemas unidimensionales sencillos. Varias formas físicas diferentes pueden caer en la

categoría de sistemas unidimensionales: los sistemas cilíndricos y esféricos son unidimensionales cuando la temperatura en el cuerpo es una función únicamente de la distancia radial y es independiente del ángulo azimutal o de la distancia axial.

En algunos problemas bidimensionales, el efecto de una segunda coordenada en el espacio puede ser tan pequeño que se justifique despreciarlo, y el problema de flujo de calor multidimensional puede aproximarse con un análisis unidimensional [15].

En estos casos, las ecuaciones diferenciales se simplifican y se obtiene una solución fácil como resultado de la simplificación.

#### **1.2.4 Aplicación de los balances diferenciales a la transferencia de calor por conducción.**

- **La pared plana.**

Para una placa de conductividad térmica constante y espesor  $b$ , (Fig.1.1.a) extendida al infinito en las otras dimensiones de tal manera que el flujo de calor en la región considerada es efectivamente unidimensional, es conveniente tratar el problema en el sistema de coordenadas rectangulares. En el caso general la

temperatura puede estar cambiando con el tiempo y puede haber fuentes de calor dentro del cuerpo. Es posible hacer el siguiente balance de energía en el elemento con espesor  $\partial z$ :

[Energía calorífica conducida en la dirección positiva de  $z$  por la cara superior] – [Energía calorífica conducida en la dirección positiva de  $z$  por la cara inferior] + [Cambio de energía interna (acumulación de energía calorífica)] = [Calor generado dentro del elemento de volumen  $S_z \partial z$ ]

Podemos simplificar como

- SALIDA – ENTRADA + ACUMULACIÓN = GENERACIÓN (1.7)
- Estas cantidades de energía están dadas como sigue:
- Calor entrando por la cara ubicada en  $z$ :  $Q_z|_z$
- Calor saliendo por la cara ubicada en  $z + \partial z$ :

$$Q_z|_{z+\partial z} = Q_z|_z + \frac{\partial Q_z}{\partial z} \partial z$$

Esto surge de la definición básica de la derivada siendo  $Q_z$  función de  $z$  como lo podemos revisar con la ayuda de la figura 1.4. La definición de la derivada  $\partial Q / \partial z$  es el límite de  $Q / z$  cuando  $z$  tiende a cero. Así el valor de  $Q$  en el punto  $z + \partial z$ , es decir  $Q_{(z + \partial z)}$ , es igual al valor de  $Q_{(z)}$  más la derivada por  $\partial z$ . En otras palabras, la derivada multiplicada por  $\partial z$  es realmente  $\partial Q$ .

A partir de la ley de Fourier encontramos el flujo de calor entrando por la cara ubicada en  $z$ :

$$Q_z|_z = -kS_z \frac{\partial T}{\partial z}|_z$$

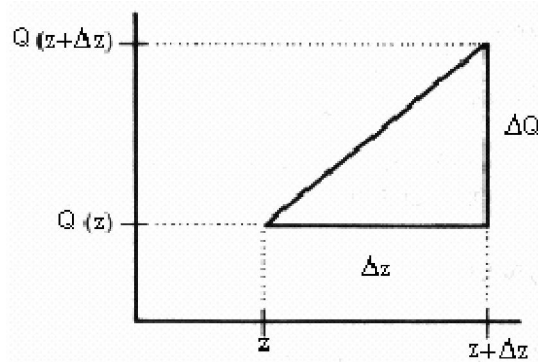


Fig. 1.4

La acumulación:  $\rho C_p (S_z dz) \frac{\partial T}{\partial t}$

En el caso de sólidos y líquidos los calores específicos a presión y volumen constantes son iguales y la velocidad de incremento de la energía interna se refleja en la velocidad de almacenamiento de energía calorífica en el elemento de volumen [6] .

La energía generada dentro del elemento es  $H S_z \partial z$  donde  $H$  es la generación de energía por unidad de volumen y unidad de tiempo por fuentes de calor distribuidas.

Combinando las relaciones anteriores, eliminando términos semejantes y dividiendo por el volumen del elemento  $S_z \partial z$ , ya que el área transversal  $S_z$  es constante:

$$-\frac{\partial}{\partial z} \left[ k \frac{\partial T}{\partial z} \right] + \rho C_p \frac{\partial T}{\partial t} = \phi_H \quad (1.8)$$

Estos términos resumen un balance térmico que expresa la primera ley de la termodinámica o ley de la conservación de la energía. Debido a que es un balance de energía térmica y no de energía total, aparece el término de generación de energía, por conversión (o degradación) de otras formas de energía en energía calorífica.

- **Placa plana sin generación en estado estable.**

Si no hay fuentes de calor dentro de la placa, y además, la conductividad térmica  $k$  es constante y el flujo de calor es estable ( $\partial T / \partial t = 0$ ) y unidimensional, la ecuación (1.8) se convierte en  $(\partial^2 T) / (\partial z^2) = 0$ , la cuál fácilmente se resuelve para dar:  $T = C_1 z + C_2$ . Las constantes  $C_1$  y  $C_2$  pueden evaluarse a partir de las condiciones límite que nos indican las temperaturas de las superficies en  $z = 0$  y  $z = b$ . Aplicando estas condiciones se obtiene una expresión para la distribución de temperaturas en la placa:

$$\frac{(T - T_1)}{(T - T_2)} = \frac{z}{b} \quad (1.9)$$



El flujo de calor a través de la placa se obtiene por la ley de Fourier de la conducción:

$$Q_z = -kS_z \frac{\partial T}{\partial z} = -\left(\frac{kS_z}{b}\right)(T_2 - T_1) = \frac{(T_1 - T_2)}{\left(\frac{b}{kS_z}\right)} \quad (1.10)$$

Es interesante resaltar la similitud entre esta ecuación y la que normalmente establece la ley de Ohm. El término  $b/kS_z$  es equivalente a la resistencia eléctrica y se denomina adecuadamente la resistencia térmica. Si la conductividad térmica varía con la temperatura de acuerdo con alguna relación, como la lineal  $k = k_0(1 + aT)$  (ver fig.1.2), la ecuación (1.8) deberá integrarse teniendo en cuenta esta variación [4].

### 1.2.5 TRANSFERENCIA EN LA INTERFASE.

Hasta ahora hemos supuesto las temperaturas de las superficies externas constantes y conocidas. Sin embargo el sólido puede estar intercambiando calor con el medio que lo rodea por convección y/o por radiación.

- **Efecto convectivo.**

El fluido que está en contacto con la superficie del sólido puede estar en movimiento laminar, o en movimiento turbulento, y éste movimiento puede ser causado por fuerzas externas, es decir, ser convección forzada; o por gradientes de densidad inducidos por las diferencias de temperatura, y será convección natural. Además puede estar cambiando de fase (ebullición o condensación).

En cualquiera de los casos anteriores el mecanismo de transferencia es complejo. Independientemente de la naturaleza particular del proceso de transferencia, el flujo de calor en la superficie se expresa como:

$$Q = hA_s(T_s - T_\infty) = \frac{(T_s - T_\infty)}{1/hA_s} \quad (1.11)$$

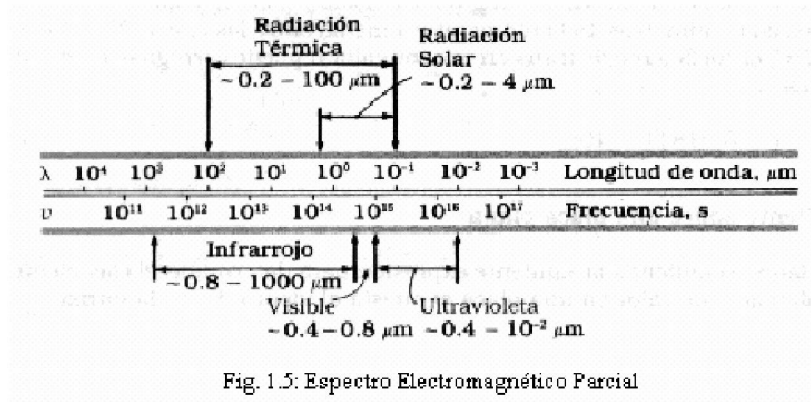
Proporcional a la diferencia de temperaturas de la superficie y del fluido respectivamente, y h es el coeficiente convectivo de transferencia de calor o coeficiente pelicular [15].

Esta ecuación se conoce como la ley de enfriamiento de Newton, y más que una ley fenomenológica, define el coeficiente de transferencia de calor h. Cualquier estudio sobre convección se reduce en últimas al estudio de los medios por los cuales puede determinarse h, el cual depende de las características de la capa límite, que a su vez está influenciada por la geometría de la superficie, por la naturaleza del movimiento del fluido y de una variedad de propiedades termodinámicas y de transporte del fluido.

## **1.2.6 RADIACIÓN.**

Todo cuerpo a una temperatura absoluta finita emite radiación electromagnética. Esta radiación, cuando está en el rango de longitud de onda comprendido entre los 0.2 y los 100  $\mu\text{m}$  se denomina térmica. Cualitativamente puede explicarse su origen a variaciones en los estados electrónico, vibracional y

rotacional de átomos o moléculas. Conforman solo una pequeña parte de todo el espectro de radiación e incluye parte de la radiación ultravioleta, la radiación visible (0.35 a 0.78  $\mu\text{m}$ ) y parte del infrarrojo [37].



La radiación no solo ocurre en superficies sólidas, ésta puede ocurrir también en líquidos y gases.

### 1.2.7 MANANTIALES CALORÍFICOS.

- **Manantial calorífico de origen eléctrico.**

La transmisión de la corriente eléctrica Según Thomson es un proceso irreversible, y parte de la energía eléctrica se transforma en calor. Consideremos un alambre conductor de sección circular con radio  $R$ . Por él circula una corriente eléctrica  $i$  amperios, con una caída de voltaje  $V$  a lo largo del alambre [37]. La velocidad de generación de calor será :

$$Q [W] = V [\text{voltios}] \times i [\text{amperios}].$$

A partir de la ley de ohm se define la resistencia eléctrica  $R_e$  de la varilla como:

$$R_e [\text{ohmio}] = V [\text{voltios}] / i [\text{amperios}]. \text{ Por esto, también } Q [W] = i^2 [\text{amp}^2] \cdot R_e [\Omega].$$

La resistencia eléctrica de la varilla es directamente proporcional a la longitud  $L$ , a la resistividad específica del material  $\rho_e [\Omega m]$  e inversamente proporcional al área transversal  $A$ . Entonces la velocidad de producción de calor debido a la disipación eléctrica por unidad de volumen viene dada por la expresión:

$$\Phi_{HC} = \frac{I^2 \rho_e L}{A} \cdot \frac{1}{AL} = \frac{I^2 \rho_e}{A^2} = I^2 \rho_e \quad \left[ \frac{W}{m^3} \right] \quad (1.12)$$

$I$  densidad de corriente eléctrica.

Si se supone que el aumento de temperatura en el alambre no es grande, no es preciso tener en cuenta la variación de las conductividades eléctrica y calorífica con la temperatura.

- **Manantial calorífico de origen viscoso.**

Consideremos el flujo de un fluido Newtoniano incompresible a través del espacio comprendido entre dos cilindros coaxiales. Al girar el cilindro exterior, las

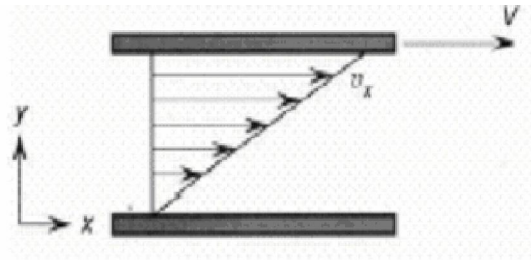
capas cilíndricas del fluido rozan con las capas de fluido adyacentes dando lugar a una producción de calor, es decir que la energía mecánica se degrada a energía calorífica [15]. La magnitud de la intensidad de manantial calorífico depende del gradiente local de velocidad; cuanto más rápidamente se mueva una capa de fluido respecto de otra adyacente, mayor será el calentamiento producido por disipación viscosa.  $T$  será función solo de  $r$ . Si el espesor  $b$  de la rendija es pequeño comparado con el radio  $R$  del cilindro exterior, el problema puede resolverse aproximadamente utilizando un sistema simplificado, es decir, despreciando los efectos de la curvatura y resolver el problema en coordenadas cartesianas.

En este caso el manantial calorífico de origen viscoso viene dado por:

$$\Phi_H = \mu \left( \frac{\partial v}{\partial r} \right)^2$$

$w$ : velocidad angular;

$v_x = R \cdot w$  (velocidad tangencial)



**Fig 1.6. Perfil de velocidades para dos cilindros concéntricos, uno de ellos fijo.**

El perfil de velocidad para el flujo laminar estacionario de un fluido de viscosidad constante en una rendija de espesor  $b$ , es lineal:  $v_x = (y/b)V$  por tanto:

$$Q = \mu \left( \frac{V}{b} \right)^2 \quad (1.13)$$

$$\Phi_H = \mu \left( V/b \right)^2$$

Al calcular el perfil de temperatura aparece espontáneamente la expresión:

$$Br = \left[ \frac{\mu V^2}{k(T_b - T_o)} \right]$$

O número de Brinkman, que es una medida de la importancia del calentamiento viscoso con relación al flujo de calor que resulta de la diferencia de temperatura comunicada ( $T_b - T_o$ ).

Para  $Br > 2$ , existe una temperatura máxima en un punto comprendido entre las dos paredes.

- **Manantial calorífico de origen químico.**

En una reacción química se produce o consume energía calorífica debido a la reordenación de los átomos de las moléculas reaccionantes para dar lugar a los productos de reacción. La velocidad de producción de energía calorífica por unidad de volumen en las reacciones químicas es generalmente una función compleja de la presión, temperatura, composición y actividad del catalizador.

Si consideramos  $\Phi_{HC}$  como una función exclusiva de la temperatura, y suponemos variación lineal con esta variable [25].

$$\Phi_{HC} = \Phi_0 \left[ \frac{T - T_0}{T_1 - T_0} \right] \quad (1.14)$$

Donde  $T$  es la temperatura local del lecho catalítico, supuesta igual para el catalizador y el fluido (aunque la diferencia de temperatura entre el catalizador y el fluido no siempre es despreciable), y  $\Phi_0$  y  $T_0$  son constantes empíricas para

unas determinadas condiciones de entrada al reactor. Usando métodos de cálculo adecuados pueden obtenerse expresiones más realistas para  $\Phi_{HN}$

- **Manantial calorífico de origen nuclear.**

Consideremos un elemento de combustible nuclear de forma esférica de radio  $R_f$ , revestido con una vaina esférica de aluminio, cuyo radio externo es  $R_c$ . La principal fuente de energía calorífica en el reactor se debe a las colisiones entre los fragmentos de fisión producidos, que poseen energías cinéticas elevadas, y los átomos del material fisionable.

Este manantial volumétrico de energía calorífica no será uniforme en el interior de la esfera del material fisionable, sino máximo en el centro de la misma. Podríamos representar este manantial mediante una función parabólica:

$$\Phi_{HN} = \Phi_{N0} \left[ 1 - m \left( \frac{r}{R_f} \right)^2 \right] \quad (1.15)$$

$\Phi_{N0}$  es la velocidad volumétrica de producción de calor en el centro de la esfera, y  $m$  es una constante adimensional comprendida entre cero y uno [5].

### **1.2.8 TÉCNICAS DE MEDICIÓN DE LA CONDUCTIVIDAD TÉRMICA.**

Las mediciones de conductividad térmica se dividen en tres grupos [20]; A. Técnicas en estado estacionario, B Técnicas en estado casi estacionario, y C Técnicas transcientes.

### **1.2.8.1 Técnicas en estado estacionario.**

- **Método de la placa caliente en paralelo.**

Este método ha sido utilizado ampliamente en materiales no alimenticios, una fuente de calor es rodeada por la muestra a la que se aplica un pulso de calor; dos protectores o aislantes están situados en ambos extremos axiales para evitar las pérdidas de calor.

La resultante del estado estacionario unidireccional de la conducción de calor en materiales isotrópicos homogéneos sin la generación interna de calor puede ser escrita como:

$$Q = \frac{kA(T_1 - T_2)}{L} \quad (1.16)$$

El valor de k puede ser calculado por medio de la cantidad de calor Q, las temperaturas internas y externas  $T_1$  y  $T_2$ , y del espesor de la muestra L.

- **Método de cilindros concéntricos.**

Este aparato consiste en dos cilindros concéntricos que pasean la muestra entre ellos y sus extremos se encuentra aislados, el calentador está situado generalmente en el cilindro externo. El líquido refrigerador se pasa a través de cilindro interno. Se asume que el calor del líquido refrigerante absorbe el calor de



la muestra mientras ésta absorbe de la fuente calórica, la tasa de conductividad térmica es calculada por la ecuación unidireccional del traspaso térmico radial como:

$$Q = A_{ou} \left[ \frac{T_{ou} - T_{in}}{r_{ou} \ln(r_{in}/r_{ou})} \right] \quad (1.17)$$

- **Método de flujo de calor.**

Este es un medidor comercial que mide el flujo de calor por medio del gradiente de la temperatura a través de la muestra, este método es conveniente para materiales con conductividades (k/l) inferiores a 11,3 W/m<sup>2</sup>K [12].

Tong y Sheen en 1992, propusieron una técnica para la determinación de la conductividad termal efectiva por medio de un recipiente plástico dividido en varias capas por una tubería de cobre de 1/8 de pulgada de diámetro, la cual se enrolla como espiral al interior [38].

La técnica según los autores consiste en introducir el recipiente plástico en una cama de agua con un agitador externo y circular al interior de la tubería una corriente de agua, además posee un medidor de flujo de calor el cual amplifica la señal 50 veces; se registran las temperaturas de la cama  $T_2$  y temperatura al interior de la muestra  $T_1$ , al comparar las temperaturas de la cama y de la salida ( $T_{ou}$ ) de la tubería se obtiene de 5 a 7 °C de diferencia. La temo conductividad es evaluada por  $(T_{in} + T_{ou})/2$ .

En estado estacionario, el flujo de calor se expresa por:

$$W = K(T_{ou} - T_{in}) \quad (1.18)$$

Donde W es el flujo de calor (W/m<sup>2</sup>) y U es el coeficiente global de calor (W/m<sup>2</sup>K). Este coeficiente se escribe como:

$$\frac{1}{U} = \frac{1}{\lambda_{in}} + \frac{1}{\lambda_{ou}} + \sum_{j=1}^N \left( \frac{l_j}{k_j} \right) \quad (1.19)$$

Donde  $\lambda_{in}$  y  $\lambda_{ou}$  son los coeficientes de calor interno y externo (W/m<sup>2</sup>K), y  $l_i$  y  $k_i$  son el espesor (m), la conductividad térmica (W/mK) para cada una de las N capas si  $\lambda_{in}$  y  $\lambda_{ou}$  son muy grande la ecuación (1.19) se reduce a:

$$\frac{1}{U} = \sum_{j=1}^N \left( \frac{l_j}{k_j} \right) \quad (1.20)$$

Si no es posible la medición de la conductividad térmica por cada capa, se puede utilizar la siguiente ecuación para la conductividad térmica efectiva.

$$\sum_{j=1}^N \left( \frac{l_j}{k_j} \right) = \frac{l_{con}}{k_e} \quad (1.21)$$

Donde  $l_{con}$  y  $k_e$  son el espesor y conductividad térmica del contenedor plástico.

Combinando las ecuaciones 1.18, 1.19, 1.20 se obtiene la ecuación para determinar la conductividad efectiva del recipiente contenedor multicapa.

$$k_e = \frac{W l_{con}}{(T_{ou} - T_{in})} \quad (1.22)$$

Las ventajas y limitantes de estos métodos son: simplicidad matemática, necesitan mayor tiempo en estabilización, deben tener una geometría definida, presencia de líquidos refrigerantes y calefactores, y no son muy útiles en sistemas biológicos.

### 1.2.8.2 Técnicas en estado casi estacionario.

- **Método de Fitch.**

Este método es uno de los métodos más utilizados en materiales de conductividad baja, Fitch en 1935 planteó un recipiente con base metálica, el cual posee una cierta cantidad de líquido a temperatura constante, otra placa completamente aislada menos la cara que da al recipiente y la muestra es colocada entre las dos placas. Este método es simple y no requiere mayor equipamiento [8].

**a) Teoría de Fitch.**

La conducción de calor en estado casi estacionario esta definida por la ecuación:

$$\frac{Ak_e(T - T_s)}{l} = m_{cp} C_{cp} \frac{\partial T}{\partial t} \quad (1.23)$$

con las condiciones iniciales:

$$t=0, \quad T=T_0$$

Resolviendo la ecuación se tiene:

$$\ln \left[ \frac{T_0 - T_s}{T - T_s} \right] = \frac{Ak_e t}{lm_{cp} C_{cp}} \quad (1.24)$$

La termo conductividad se calcula por medio de la pendiente del grafico de  $\ln(T_o-T_s)/(T-T_s)$  vs t.

Mohsenin, [19] y Rahma, [28], plantean que existe una diferencia entre el valor real y el valor calculado siendo necesaria la calibración del sistema cada vez que se realiza una medición, ya que es una función cuadrática que depende del espesor de la muestra y del área de contacto.

- **Método de Cenco-Fitch.**

La mejora radica en el mejor aislamiento y doble fuente calorífica del sistema, Bennet menciona que este método es una modificación del método de Fitch, que reduce el error a un 15% como máximo posible[3]. Las razones son:

- a) Los errores impredecibles que se presentan por la presión aplicada a causa del liquido, o cambiar el espesor y/o características de la muestra.
- b) Poca precisión del flujo de calor radial.
- c) Considerable pérdida de calor y prácticamente inmedible por el sistema de prensa.

- **Método de Fitch modificado por Zuritz.**

Zuritz modifíco el aparato diseñado por Fitch, para muestras de alimentos de menor tamaño, el método es mucho más simple que el original, la conductividad térmica es difícil obtenerla en ciertos materiales agropecuarios como granos y mariscos, los cuales no se puede cortarlos de manera cilíndrica o

en láminas delgadas para su medición. La característica de material biológico y aire, se puede considerar como un cuerpo poroso y puede ser medido como tal, claro está que la determinación de la conductividad propia del alimento es importante en los procesos de transferencia de calor en sistemas estáticos [45].

Este método es útil en pequeñas proporciones y conductividades inferiores a 1,04 W/mK, con un error del 6,9%.

- **Método de Fitch- Rahman.**

Rahamn se basó en la experiencia de Zuritz [45], para alimentos frescos y refrigerados, las ventajas principales de esta modificación son:

1. Las pérdidas de calor por los bordes de la muestra son insignificantes.
2. Al provocar una convección radial, esta provoca una reducción en el grosor de plancha transmisora, haciendo más eficiente el trapazo térmico.

El error de este método fue predicho por la media independiente de muestras de manzanas, zanahorias, peras y calamares con un espesor de 4mm, después de calibrarlo el error fue de 5% por encima de cero grados y de 9% por debajo de éste [27].

Del punto de vista teórico la aplicación del error depende directamente de la ecuación 1.24, la cual debe satisfacer todos sus componentes en estado casi estacionario, la pendiente de la curva de éste método puede ser escrita como:

$$S = \frac{AK_e}{lm_{CP}C_{CP}} \quad (1.25)$$

### 1.2.8.3 Técnicas en estado transiente.

- **Método de la fuente en línea.**

Este método es el más usado en alimentos por que no considera geometrías ni tiempos de prolongados de medición, siendo de bajo coste.

#### **Teoría.**

De acuerdo con Hooper and Lepper [13], Van der Held and Van Drunen [44] desarrollaron una aplicación correcta al principio de medición térmica por fuente en línea, el detalle matemático deriva de la solución de cambios en la temperatura en un punto y se considera una línea de calor como:

$$T = B + \left[ \frac{q}{4\pi k} \right] \ln t \quad (1.26)$$

Donde  $q$  es el calor de entrada por unidad de longitud (W/m),  $t$  es el tiempo del experimento,  $B$  es el intercepto,  $T$  la temperatura,  $K$  la conductividad de fuente.

Se debe graficar  $T$  vs  $\ln t$  para linealizar la función y trabajar con la pendiente de la recta.

Kontani [18] uso una aguja hipodérmica para irradiar el calor en muestras y otra para medir la temperatura de muestra situadas a 1 cm de distancia una de otra, y circuló una tensión continua por 3 minutos, posteriormente Rahman [27] fusionó las dos hipodérmicas para introducir una sola aguja en la muestra.

### **1.2.9 DIFUSIVIDAD TÉRMICA**

La tasa a la que se difunde el calor depende de la relación que existe entre el material y la conductividad de este; pudiendo definirlo de manera:

$$\alpha = \frac{k}{\sigma \cdot Cp} \quad (1.27)$$

Donde la difusividad ( $\alpha$ ) se expresa en  $m^2/s$ , la densidad ( $\sigma$ ) en  $kg/m^3$ , el calor específico ( $Cp$ ), en  $J/kg \cdot ^\circ K$  y la conductividad ( $k$ ) en  $W/m \cdot ^\circ K$ .

### **1.2.10 TÉCNICAS DE MEDICIÓN DE LA DIFUSIVIDAD TÉRMICA**

Para la determinación de la difusividad se han utilizado métodos de determinación experimental directa luego de obtener la conductividad, densidad y calor específico, o la utilización de ecuaciones predictoras en función a la

composición del material. Si bien la metodología es aceptada podemos determinarlos de forma experimental utilizando los siguientes principios:

- **Determinación Directa.**

Para esta determinación se utiliza el principio de conducción de una dirección en estado no estacionario del calor, lo que nos permite hacer una relación con el número de Biot, (resistencia interna/resistencia externa).

$$Bi = \frac{hR}{k} \quad (1.28)$$

### **1.2.11 CALOR ESPECÍFICO**

El calor específico se define como el calor necesario para elevar la temperatura de un cuerpo en una unidad de masa por un grado; siendo la unidad de medida J/kg°K, el calor específico depende de la naturaleza y del proceso de adición de temperatura sea por presión o por aumento de calor del material, en condiciones normales de operación el calor específico representa una cualidad importante en el desarrollo de los sistemas de producción.[19]; el término de entalpía puede ser escrito por:

$$H = \int C dT \quad (1.29)$$

Siendo C la constante de temperatura de la ecuación.

### **1.2.12 TÉCNICAS DE MEDICIÓN DEL CALOR ESPECÍFICO**



Para la determinación del calor específico hay cuatro metodologías: A) método de la mezcla, B) método comparativo, C) método adiabático, D) método de la calorimetría diferencial.

- **Método de la mezcla**

Este método se basa en la diferencia de temperatura entre un cuerpo con calor específico como el agua y un cuerpo desconocido (muestra a analizar), lo único conocido es la masa de los dos cuerpos, dado una ecuación con una incógnita.

$$C_{sa} = \frac{C_w(W_w + e)(T_{oc} - T_{em})}{W_{sa}(T_{os} - T_{em})} \quad 1.30$$

Donde:

$C_{sa}$  = calor específico de la muestra.

$T_{oc} - T_{em}$  = Rango de temperatura del experimento.

$C_w$  = Calor específico del agua.

$W_w$  = Masa del agua.

$W_{sa}$  = Masa de la muestra.

$E$  = Temperatura de equilibrio de la mezcla.

- **Método comparativo.**

Es una corrección al método anterior, varios autores han propuesto factores de corrección a la ecuación, pero todas concuerdan en la grafica cruzada entre la evolución de temperatura de la mezcla versus la evolución de temperatura de la sustancia patrón, donde se interceptan las curvas, ese es el valor del calor específico de la muestra.

- **Método adiabático**

Mohsenin [19] presenta este método para la medición del calor específico, en un recipiente aislado provisto de una resistencia eléctrica que la envuelve, suministrando el calor necesario para la medición, al interior del recipiente se deposita la muestra provista por una termocupla en el centro de la misma; Para calcular el calor específico de la muestra es necesario comparar la energía suministrada versus la energía absorbido dando como resultado la siguiente ecuación:

$$C_{sa} = \frac{VIt}{W_{sa}(T_{es} - T_{os})} \quad 1.31$$

Donde:

$C_{sa}$ = calor específico de la muestra.

$T_{es} - T_{em}$  = Rango de temperatura del experimento.

V= Voltaje.

I= Intensidad.

T= Tiempo.

$W_{sa}$ = Masa de la muestra.

- **Método de la calorimetría diferencial (DSC)**

La temperatura se incrementa gradualmente en una celda térmica a una cantidad de masa determinada, a una tasa de transferencia conocida y con la ecuación de balance térmico se encuentra la capacidad calórica, varios autores han modificado la forma del experimento obteniendo los mismos resultados, a este método también se lo conoce como variación de entalpía [28].

## ***Capítulo 2***

### **El Hardware de control.**

#### **2.1 Introducción.**

Los requisitos del sistema de adquisición de datos que se debe implementar:

- Debe ser capaz de almacenar datos sobre más de un sensor para determinar la conductividad del alimento.
- Los periodos de muestra deben ser flexibles y en el orden de segundos y fracción de estos.
- Los datos deben ser almacenados en un ordenador, el cual permite gestionar adecuadamente la visualización gráfica, almacenamiento en soporte físico para su posterior tratamiento, etc.

#### **2.2 Adquisidor de datos.**

El adquisidor de datos será el cuerpo del proyecto que se pretende explicar aquí; Dicho adquisidor podría ser implementado usando un modelo comercial existente siempre que cumpla las especificaciones citadas.

En primera instancia se pensó en un ordenador 386 o 486, pero no soportan la tecnología Universal Serial Bus (USB), así que el requerimiento mínimo es un

Pentium®III con sistema operativo Windows XP, ya que se desarrollaría una interfase la cual se conectara como cualquier otro periférico USB como cámara de video, escáner o impresora.

Se plantea trabajar con USB 1.1 el cual permite una transferencia de 1.5Mbps, el cual es apropiado para sistema de adquisición de temperatura, además que es soportado por todos los equipos con puertos USB desde la primera generación hasta la actual USB On to Go.

El sistema adquisidor estará compuesto por dos placas y dos microcontroladores, la primera encargada de la comunicación entre el ordenador y el periférico, la segunda tendrá la tarea de tomar los datos desde los sensores y transformar la señal para ser distribuida por la primera placa, esto presenta la ventaja de que puede ser ampliada las prestaciones del equipo sin la necesidad de construir todo el sistema nuevamente.

### **2.2.1 Microcontroladores**

El mercado actual de los microcontroladores está experimentando en los últimos años un gran crecimiento en cuanto a las prestaciones que dichos dispositivos poseen [1].

Actualmente se pueden encontrar microcontroladores a muy bajo precio que poseen periféricos para múltiples tareas. Las especificaciones que debe cumplir el adquisidor pueden ser implementadas por el microcontrolador o bien

por dispositivos externos integrados en la misma placa. Las funciones que se tienen que implementar son:

- Convertidor analógico a digital.
- Capacidad de comunicación con el ordenador.
- Gestión de entrada y salida con los usuarios.
- Posibilidad de diferentes configuraciones.

Dentro de toda la gran gama que hay en el mercado existen multitud de microcontroladores, fabricados por muchas empresas importantes como Motorola, Siemens, Mitsubishi, Zilog o Intel, entre otras. Lo que hizo inclinar la balanza hacia los PIC's no fue su arquitectura optimizada para aumentar al máximo las capacidades de un dispositivo, sino todo el conjunto de cualidades que permite el desarrollo al igual que la información disponible de los microcontroladores [2].

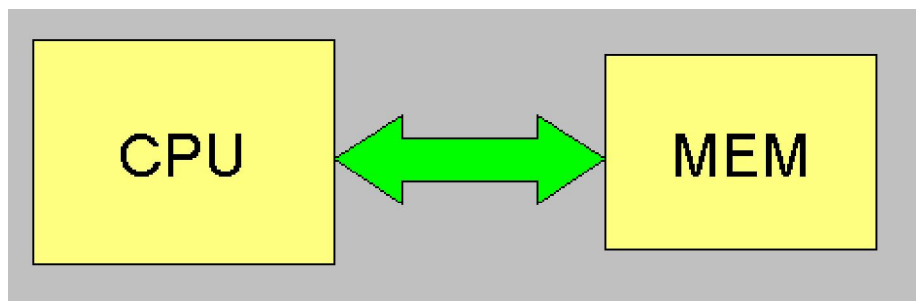
### **2.2.2 Arquitectura de los PIC.**

La arquitectura tradicional de computadoras y microprocesadores se basa en el esquema propuesto por John Von Neumann, en el cual la unidad central de proceso, (CPU), esta conectada a una memoria única que contiene las instrucciones del programa y los datos (figura 2.1). El tamaño de la unidad de datos o instrucciones esta fijado por el ancho del bus de la memoria. Es decir que un microprocesador de 8 bits, que tiene además un bus de 8 bits que lo conecta con la memoria, deberá manejar datos e instrucciones de una o más unidades de 8 bits (bytes) de longitud. Cuando deba acceder a una instrucción o dato de más de un byte de longitud, deberá realizar más de un acceso a la

memoria. Por otro lado este bus único limita la velocidad de operación del microprocesador, ya que no se puede buscar de memoria una nueva instrucción, antes de que finalicen las transferencias de datos que pudieran resultar de la instrucción anterior [7]. Es decir que las dos principales limitaciones de esta arquitectura tradicional son:

- a) Que la longitud de las instrucciones esta limitada por la unidad de longitud de los datos, por lo tanto el microprocesador debe hacer varios accesos a memoria para buscar instrucciones complejas.
- b) Que la velocidad de operación (o ancho de banda de operación) esta limitada por el efecto de cuello de botella que significa un bus único para datos e instrucciones que impide superponer ambos tiempos de acceso.

La arquitectura Von Neumann permite el diseño de programas con código auto modificable, práctica bastante usada en las antiguas computadoras que solo tenían acumulador y pocos modos de direccionamiento, pero innecesaria, en las computadoras modernas.



***Figura 2.1 Arquitectura Von Neumann***

### **2.2.2.1 Arquitectura Harvard.**

La arquitectura conocida como Harvard, consiste simplemente en un esquema en el que el CPU esta conectado a dos memorias por intermedio de dos buses separados. Una de las memorias contiene solamente las instrucciones del programa, y es llamada Memoria de Programa. La otra memoria solo almacena los datos y es llamada Memoria de Datos (figura 2.2). Ambos buses son totalmente independientes y pueden ser de distintos anchos. Para un procesador de Set de Instrucciones Reducido, o RISC (Reduced Instrucción Set Computer), el set de instrucciones y el bus de la memoria de programa pueden diseñarse de manera tal que todas las instrucciones tengan una sola posición de memoria de programa de longitud. Además, como los buses son independientes, el CPU puede estar accediendo a los datos para completar la ejecución de una instrucción, y al mismo tiempo estar leyendo la próxima instrucción a ejecutar [10]. Se puede observar claramente que las principales ventajas de esta arquitectura son:

- a) que el tamaño de las instrucciones no esta relacionado con el de los datos, y por lo tanto puede ser optimizado para que cualquier instrucción ocupe una sola posición de memoria de programa, logrando así mayor velocidad y menor longitud de programa,

- b) que el tiempo de acceso a las instrucciones puede superponerse con el de los datos, logrando una mayor velocidad de operación.

Una pequeña desventaja de los procesadores con arquitectura Harvard, es que deben poseer instrucciones especiales para acceder a tablas de valores constantes que pueda ser necesario incluir en los programas, ya que estas tablas se encontraran físicamente en la memoria de programa (por ejemplo en la EPROM de un microprocesador).

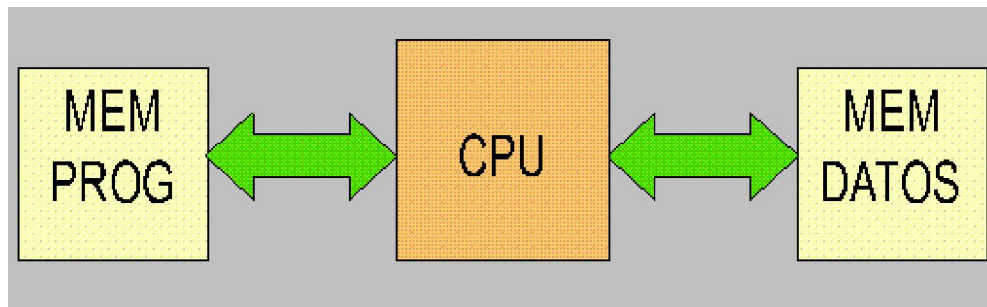


Figura 2.2 Arquitectura Harvard

Los microcontroladores PIC 16C5X, 16CXX y 17CXX poseen arquitectura Harvard, con una memoria de datos de 8 bits, y una memoria de programa que, según el modelo, puede ser de 12 bits para los 16C5X, 14 bits para los 16CXX y 16 bits para los 17CXX [7].

#### **2.2.2.2 Diagrama de bloques**



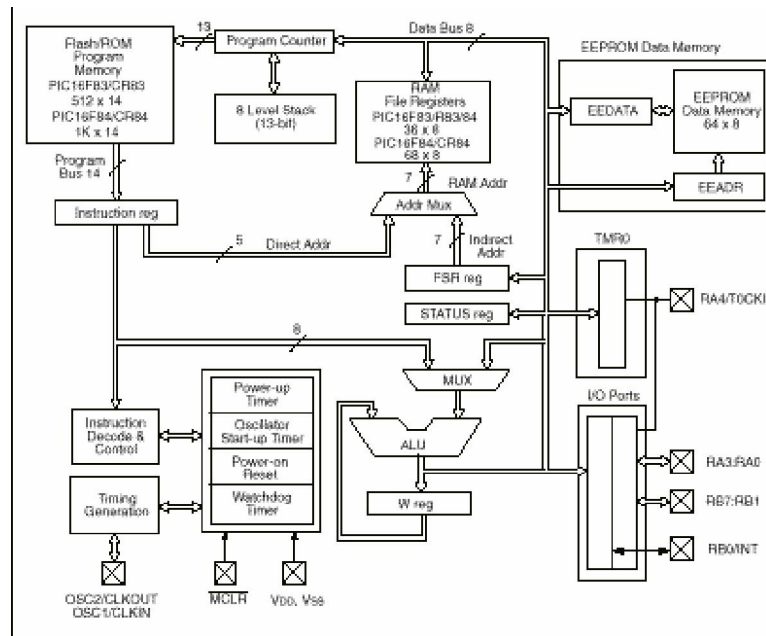


Figura 2.3 Diagrama de bloques de los microcontroladores PIC16F8X

### 2.2.2.3 Mapas de memoria Interna (RAM)

#### A) Organización

La memoria interna de datos, también llamada archivo de registros (register file), esta dividida en dos grupos: los registros especiales, y los registros de propósito generales. Los primeros ocupan las 11 posiciones primeras que van desde la 00 a la 0B, y los segundos las posiciones que siguen, o sea de la 0C a la 4F.

Los registros especiales contienen la palabra de estado (STATUS), los registros de datos de los tres puertos de entrada salida (Puerto A, Puerto B, Puerto C), los 8 bits menos significativos del program counter (PC), el contador del Real Time Clock/Counter (RTCC) y un registro puntero llamado File Select Register (FSR). La posición 00 no contiene ningún registro en especial y es utilizada en el mecanismo de direccionamiento indirecto.

Los registros de propósito general se dividen en dos grupos: los registros de posición fija y los bancos de registros. Los primeros ocupan las 8 posiciones que van de la 08 a la 0F. los bancos de registros consisten en hasta cuatro grupos o bancos de 16 registros cada uno, que se encuentran superpuestos en las direcciones que van de la 10 a la 1F. Se puede operar con un solo banco a la vez, el cual se selecciona mediante los bits 5 y 6 del File Select Register (FSR)

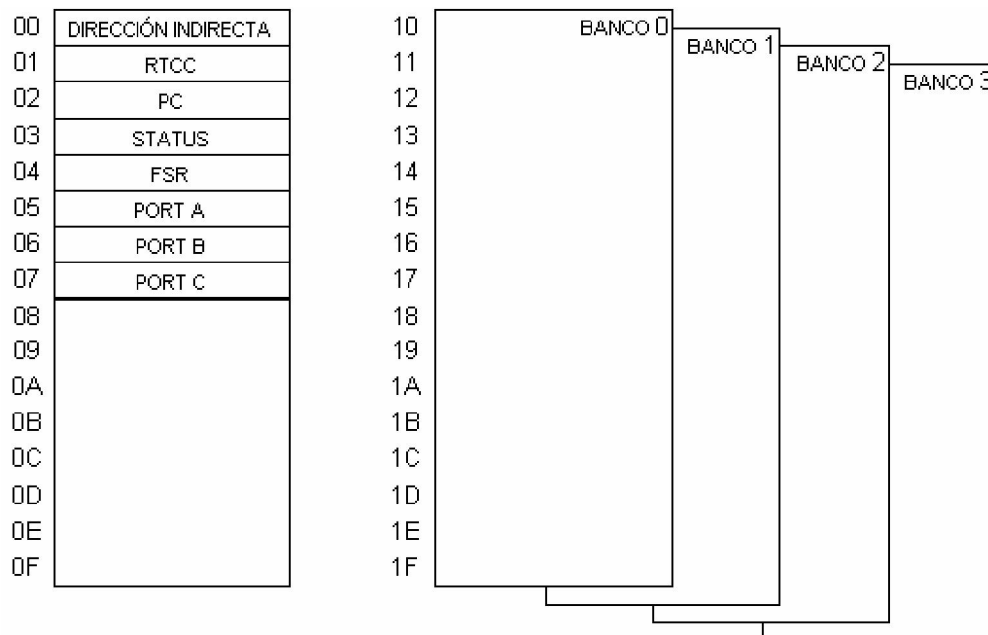


Figura 2.4 Organización de la memoria Interna (RAM) en la familia PIC16CXX [21].

### B) Memoria de Programa Organización

La memoria de programa, que en los PIC16CXX puede ser de 512 a 2K instrucciones, debe ser considerada a los efectos de la programación, como compuesta por secciones o páginas de 512 posiciones. A su vez cada página debe considerarse dividida en dos mitades de 128 posiciones cada una. Esto se debe, como se verá, a las limitaciones de direccionamiento de las instrucciones de salto [22].

	000
----	0FF
----- PAGINA 0 -----	100
	1FF
	200
----	2FF
----- PAGINA 1 -----	300
	3FF
	400
----	4FF
----- PAGINA 2 -----	500
	5FF
	600
----	6FF
----- PAGINA 3 -----	700
	7FF

Figura 2.5 Organización de la memoria de programa en la familia PIC16CXX

#### 2.2.2.4 Registros de funciones especiales Camino de los datos y registro W

La figura 2.6 b) representa un diagrama simplificado de la arquitectura interna del camino de los datos en el CPU de los microcontroladores PIC. Este diagrama puede no representar con exactitud el circuito interno de estos microcontroladores, pero es exacto y claro desde la óptica del programador. La figura 2.6 a) representa el mismo diagrama para un microprocesador ficticio de arquitectura tradicional. Se puede observar que la principal diferencia entre ambos radica en la ubicación del registro de trabajo, que para los PIC's se denomina W (Working Register), y para los tradicionales es el Acumulador (A).

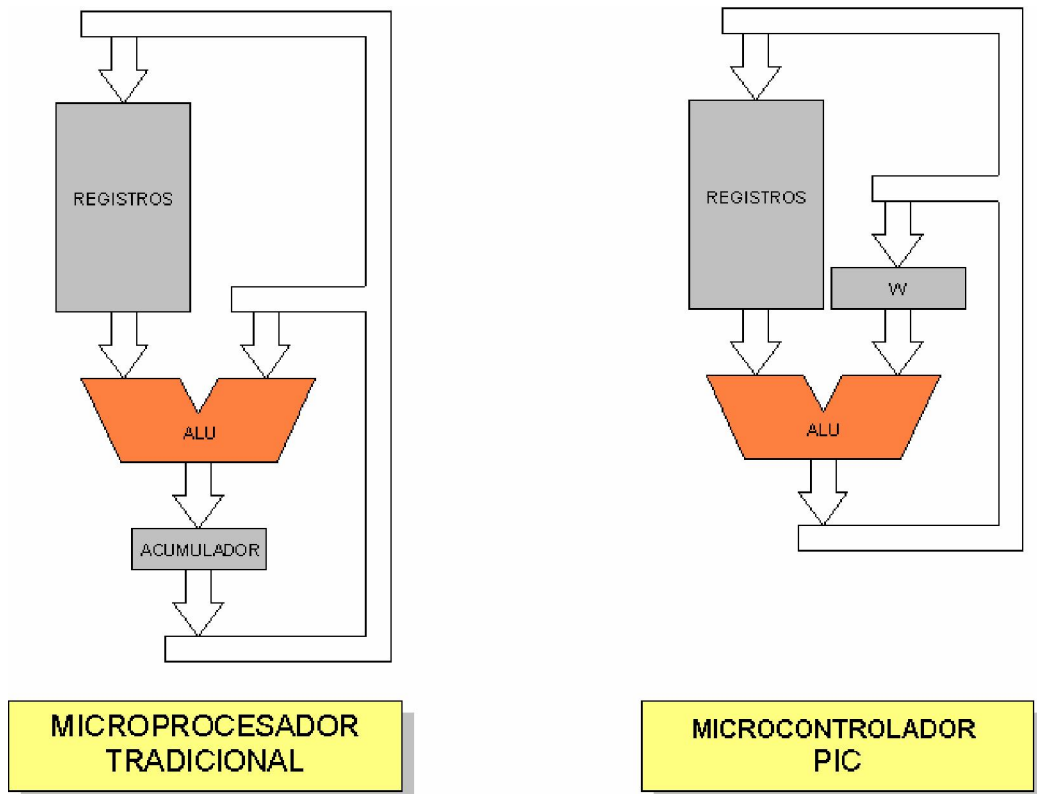


Figura 2.6 a) Estructura de un Microprocesador

Figura 2.6 b) Estructura de un PIC

En los microcontroladores tradicionales todas las operaciones se realizan sobre el acumulador. La salida del acumulador esta conectada a una de las entradas de la Unidad Aritmética y Lógica (ALU), y por lo tanto éste es siempre uno de los dos operandos de cualquier instrucción. Por convención, las instrucciones de simple operando (borrar, incrementar, decrementar, complementar), actúan sobre el acumulador. La salida de la ALU va solamente a la entrada del acumulador, por lo tanto el resultado de cualquier operación siempre quedara en este registro. Para operar sobre un dato de memoria, luego realizar la operación siempre hay que mover el acumulador a la memoria con una instrucción adicional.

En los microcontroladores PIC, la salida de la ALU va al registro W y también a la memoria de datos, por lo tanto el resultado puede guardarse en cualquiera de los dos destinos. En las instrucciones de doble operando, uno de los dos datos siempre debe estar en el registro W, como ocurría en el modelo tradicional con el acumulador. En las instrucciones de simple operando el dato en este caso se toma de la memoria (también por convención). La gran ventaja de esta arquitectura es que permite un gran ahorro de instrucciones ya que el resultado de cualquier instrucción que opere con la memoria, ya sea de simple o doble operando, puede dejarse en la misma posición de memoria o en el registro W, según se seleccione con un bit de la misma instrucción. Las operaciones con constantes provenientes de la memoria de programa (literales) se realizan solo sobre el registro W.

En la memoria de datos de los PIC's se encuentran ubicados casi todos los registros de control del microprocesador y sus periféricos autocontenidos, y también las posiciones de memoria de usos generales. En el caso de los 16C5X, algunos registros especiales de solo escritura (TRIS y OPTION) no están accesibles dentro del bloque de memoria de datos, sino que solo se pueden cargar desde el registro W por medio de instrucciones especiales.

#### **2.2.2.5 Contador de Programa**

Este registro, normalmente denominado PC, es totalmente equivalente al de todos los microprocesadores y contiene la dirección de la próxima instrucción a ejecutar. Se incrementa automáticamente al ejecutar cada instrucción, de manera que la secuencia natural de ejecución del programa es lineal, una instrucción después de la otra. Algunas instrucciones que llamaremos de control, cambian el contenido del PC alterando la secuencia lineal de ejecución. Dentro de estas instrucciones se encuentran el GOTO y el CALL que permiten cargar en forma directa un valor constante en el PC haciendo que el programa salte a cualquier posición de la memoria. Otras instrucciones de control son los SKIP o “salteos” condicionales, que producen un incremento adicional del PC si se cumple una condición específica, haciendo que el programa saltee, sin ejecutar, la instrucción siguiente [30].

El PC por ejemplo es un registro de 9 bits en los 16C54/55, 10 bits en el 16C56, y 11 bits en el 16C57, lo que permite direccionar respectivamente 512, 1024 o 2048 posiciones de memoria de programa.

Al resetearse el microprocesador, todos los bits del PC toman valor 1, de manera que la dirección de arranque del programa es siempre la última posición de memoria de programa. En esta posición se deberá poner una instrucción de salto al punto donde verdaderamente se inicia el programa.

A diferencia de la mayoría de los microprocesadores convencionales, el PC es también accesible al programador como registro de memoria interna de datos, en la posición de 02. Es decir que cualquier instrucción común que opere sobre registros puede ser utilizada para alterar el PC y desviar la ejecución del programa. El uso indiscriminado de este tipo de instrucciones complica el programa y puede ser muy peligroso, ya que puede producir comportamientos difíciles de predecir. Sin embargo, algunas de estas instrucciones utilizadas con cierto método, pueden ser muy útiles para implementar poderosas estructuras de control tales como el goto computado. Como el microprocesador opera con datos de 8 bits, y la memoria de datos es también de 8 bits, estas instrucciones solo pueden leer o modificar los bits 0 a 7 del PC [36].

#### **2.2.2.6 Stack**

En los microcontroladores PIC el stack es una memoria interna dedicada, de tamaño limitado, separada de las memorias de datos y de programa, inaccesible al programador, y organizada en forma de pila, que es utilizada

solamente, y en forma automática, para guardar las direcciones de retorno de subrutinas e interrupciones. Cada posición es de 11 bits y permite guardar una copia completa del PC. Como en toda memoria tipo pila, los datos son accedidos de manera tal que el primero que entra es el ultimo que sale.

En los 16CXX el stack es de solo dos posiciones, mientras que en los 16CXX es de 8 posiciones y en los 17CXX es de 16 posiciones. Esto representa, en cierta medida, una limitación de estos microcontroladores, ya que no permite hacer uso intensivo del anidamiento de subrutinas. En los 16CXX, solo se pueden anidar dos niveles de subrutinas, es decir que una subrutina que es llamada desde el programa principal, puede a su vez llamar a otra subrutina, pero esta ultima no puede llamar a una tercera, porque se desborda la capacidad del stack, que solo puede almacenar dos direcciones de retorno. Esto de hecho representa una traba para el programador y además parece impedir o dificultar la programación estructurada, sin embargo es una buena solución de compromiso ya que estos microcontroladores están diseñados para aplicaciones de alta velocidad en tiempo real, en las que el overhead (demoras adicionales) que ocasiona un excesivo anidamiento de subrutinas es inaceptable. Por otra parte existen técnicas de organización del programa que permiten mantener la claridad de la programación estructurada, sin necesidad de utilizar tantas subrutinas anidadas [35].

Como ya se menciona anteriormente, el stack y el puntero interno que lo direcciona, son invisibles para el programador, solo se los accede automáticamente para guardar o rescatar las direcciones de programa cuando se



ejecutan las instrucciones de llamada o retorno de subrutinas, o cuando se produce una interrupción o se ejecuta una instrucción de retorno de ella.

### **2.2.2.7 Palabra de Estado del Procesador**

La palabra de estado del procesador contiene los tres bits de estado de la ALU (C, DC y Z), y otros bits que por comodidad se incluyeron en este registro.

7 6 5 4 3 2 1 0 Registro STATUS El bit Z indica que el resultado de la última operación fue CERO. El bit C indica acarreo del bit más significativo (bit 7) del resultado de la última operación de suma. En el caso de la resta se comporta a la inversa, C resulta 1 si no hubo pedido de préstamo. El bit DC (digit carry) indica acarreo del cuarto bit (bit 3) del resultado de la última operación de suma o resta, con un comportamiento análogo al del bit C, y es útil para operar en BCD (para sumar o restar números en código BCD empaquetado). El bit C es usado además en las operaciones de rotación derecha o izquierda como un paso intermedio entre el bit 0 y el bit 7 [10].

El bit PD (POWER DOWN) sirve para detectar si la alimentación fue apagada y encendida nuevamente, tiene que ver con la secuencia de inicialización, el watch dog timer y la instrucción sleep, y su uso se detallara en la sección referida al modo POWER DOWN. El bit TO (TIME-OUT) sirve para detectar si una condición de reset fue producida por el watch dog timer, esta relacionado con los mismos elementos que el bit anterior y su uso se detallara en la sección referida al WATCH DOG TIMER. Los bits de selección de pagina

PA0/PA1/PA2 se utilizan en las instrucciones de salto GOTO y CALL, y se explicaran con detalle en la sección referida a las instrucciones de control, y a la organización de la memoria de programa[7].

### **2.2.2.8 OTROS REGISTROS ESPECIALES**

Las ocho primeras posiciones del área de datos están reservadas para alojar registros de propósito especial, quedando las restantes libres para contener los datos u operandos que se desee (registros de propósito general).

El registro INDF que ocupa la posición 0 no está implementando físicamente y, como se ha explicado, se le referencia en el direccionamiento indirecto de datos aunque se utiliza el contenido de FSR.

En la dirección esta el registro TAR0 (Temporizador) que puede ser leído y escrito como cualquier otro registro. Puede incrementar su valor con una señal externa aplicada al pin T0CKI o mediante el oscilador interno.

El PC ocupa la posición 2 del área de datos en donde se halla el registro PCL al que se añaden 3 bits auxiliares y se conectan con los dos niveles de la Pila en las instrucciones CALL y RETLW.

El registro de Estado ocupa la posición 3 y entre sus bits se encuentran los señalizadores C, DC y Z y los bits PA1 y PA0 que seleccionan la página en la memoria de programa. El bit 7 (PA2) no está implementando en los PIC de la gama baja [35].

FRS se ubica en la dirección 4 y puede usarse para contener las direcciones del dato en las instrucciones con direccionamiento indirecto y también para guardar operandos en sus 5 bits de menos peso.

Los registros que ocupan las posiciones 5 ,6 y 7 soportan los Puertos A, B y C de E/S. Pueden ser leídos y escritos como cualquier otro registro y manejan los valores de los bits que entran y salen por los pines de E/S del microcontrolador.

### 2.2.2.9 Puertos de entrada / salida

Los microprocesadores PIC16CXX tienen dos o tres puertos de entrada/salida paralelo de usos generales llamados Puerto A, Puerto B y Puerto C. El Puerto A es de cuatro bits y los demás son de 8 bits cada uno.

### 2.2.2.10 Circuito equivalente

El circuito equivalente de un bit cualquiera de un puerto de entrada salida es el siguiente

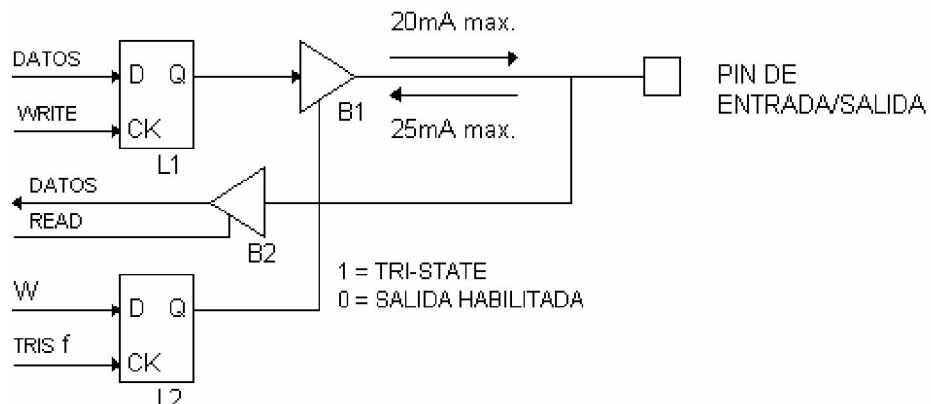


Figura 2.7 Circuito equivalente de puerto I/O

El latch L1 corresponde a un bit del registro de datos del puerto, mientras que L2 es un bit del registro de control de tris-tate del mismo. B1 es el buffer tri-state de salida que tiene capacidad de entregar 20 mA y drenar 25 mA. B1 es controlado por L2. Si L2 tiene cargado un “1”, B1 se encuentra en tri-state, es decir con la salida desconectada (en alta impedancia), y el puerto puede ser usado como entrada. Si L2 tiene cargado un “0”, la salida de B1 esta conectada (baja impedancia) y el puerto esta en modo de salida. B2 es el buffer de entrada, es decir el que pone los datos en el bus interno del microcontrolador cuando se lee el registro de datos del puerto. Puede verse que el dato leído es directamente el estado del pin de entrada [36]

### 2.2.2.11 Diagrama lógico

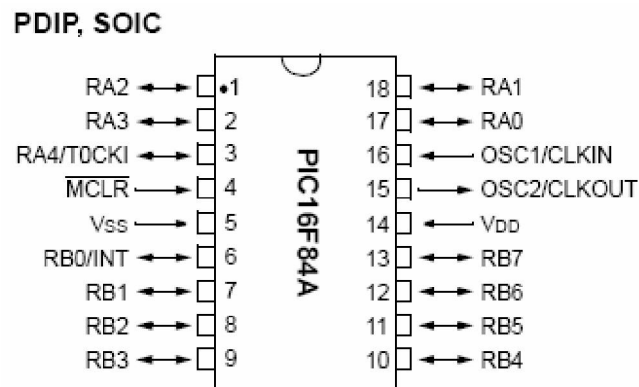


Figura 2.8 Diagrama lógico para los microcontroladores PIC16F8X

### **2.2.2.12 Temporizador/Contador (RTCC).**

Este dispositivo, llamado Real Time Clock / Counter, es básicamente un contador de 8 bits, constituido por el registro operacional RTCC que se encuentra en la posición 01 de la memoria de datos. Este registro puede usarse para contar eventos externos por medio de un pin de entrada especial (modo contador) o para contar pulsos internos de reloj de frecuencia constante (modo timer). Además, en cualquiera de los dos modos, se puede insertar un prescaler, es decir un divisor de frecuencia programable que puede dividir por 2, 4, 8, 16, 32, 64, 128 o 256. Este divisor puede ser utilizado alternativamente como prescaler del RTCC o como postscaler del Watch Dog Timer, según se lo programe [7].

Para su programación se dispone de dos registros: el RTCC ya mencionado y el registro OPTION. Este último no es accesible como memoria de datos, no se lo puede leer de ninguna manera, y solo se lo puede escribir con la instrucción especial OPTION (familia PIC16F8X). Este registro contiene los bits necesarios para seleccionar modo contador o modo timer, flanco de conteo en modo contador, prescaler para RTCC o para WDT y constante de división del prescaler, según el siguiente esquema:

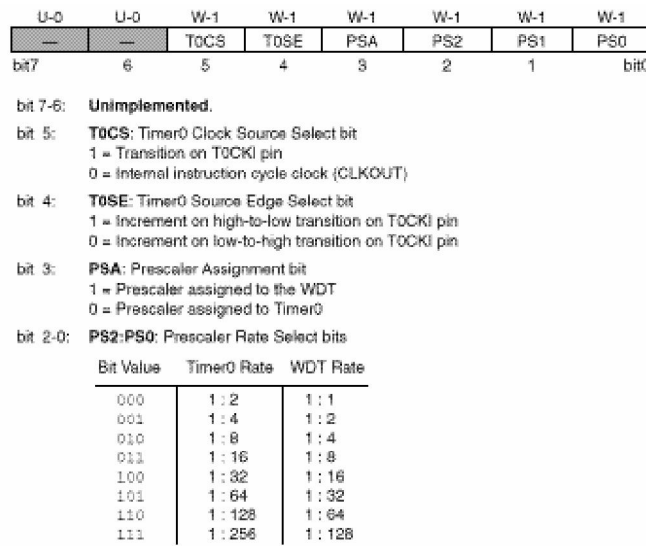
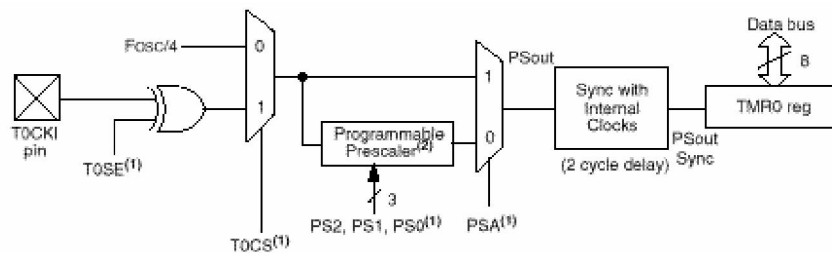


Figura 2.9 A continuación se muestra un circuito equivalente del RTCC (TMR0) y el prescaler.



En el esquema se puede observar claramente como operan los bits de configuración T0SE, T0CS y PSA, en cualquiera de sus combinaciones. Se observa además que en la entrada del contador RTCC hay un circuito de sincronización que introduce una demora de dos ciclos del clock de instrucciones ( $F_{osc} / 4$ ). Al escribir sobre el RTCC automáticamente se resetea este circuito, por lo tanto solo se incrementará dos ciclos después.

El prescaler es un contador asincrónico de 8 bits más un multiplexor 8 a 1 comandado por los bits PS0 a PS2, que permite seleccionar como salida a cualquiera de los bits del contador. Al escribir sobre el RTCC, si este está programado para operar con prescaler (PSA=0), se borra automáticamente el prescaler. Las instrucciones CLRWDT y SLEEP borran el prescaler, si este está programado para operar como postscaler del watch dog timer (PSA=1)[7].

### **2.2.2.13 Interrupciones.**

Los 16FXX agregan la posibilidad de contar con sistema de interrupciones. Este sistema consiste en un mecanismo por el cual un evento interno o externo, asincrónico respecto del programa, puede interrumpir la ejecución de éste produciendo automáticamente un salto a una subrutina de atención, de manera que pueda atender inmediatamente el evento, y retomar luego la ejecución del programa exactamente en donde estaba al momento de ser interrumpido. Este mecanismo es muy útil por ejemplo para el manejo de timers o rutinas que deben repetirse periódicamente (refresh de display, antirebote de teclado, etc.), detección de pulsos externos, recepción de datos, etc.

Existen de tres a doce eventos que pueden generar interrupciones en los PIC16FXX existentes hasta el momento, pero nada impide que puedan agregarse más en versiones futuras.

### **2.2.2.14 Funcionamiento**

En los 16FXX las interrupciones se comportan casi exactamente igual que las subrutinas. Desde el punto de vista del control del programa, al producirse una interrupción se produce el mismo efecto que ocurriría si el programa tuviese un CALL 0004h en el punto en que se produjo la interrupción. En uno de los registros de control del sistema de interrupciones existe un bit de habilitación general de interrupciones GIE, que debe ser programado en 1 para que las interrupciones puedan actuar. Al producirse una interrupción, este bit se borra automáticamente para evitar nuevas interrupciones. La instrucción RETFIE que se utiliza al final de la rutina de interrupción, es idéntica a un retorno de subrutina, salvo que además coloca en uno automáticamente el bit GIE volviendo a habilitar las interrupciones. Dentro de la rutina de interrupción, el programa deberá probar el estado de los flags de interrupción de cada una de las fuentes habilitadas, para detectar cual fue la que causó la interrupción y así decidir que acción tomar.

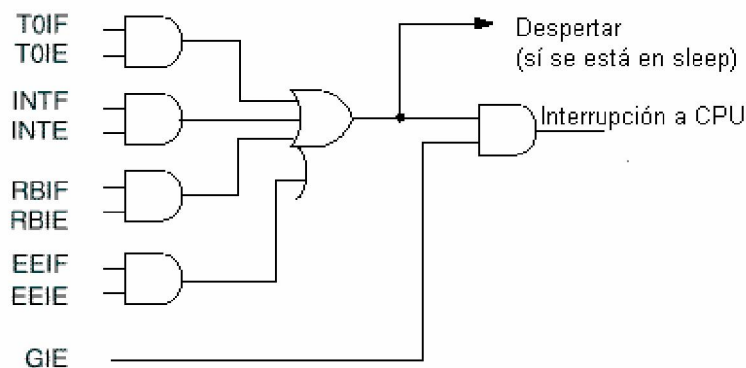


Figura 2.10 Lógica de interrupciones para los controladores PIC16F8X

### 2.2.2.15 Fuentes



La señal que produce la interrupción es en realidad una sola, que resulta de la combinación de todas las fuentes posibles y de los bits de habilitación. Existen dos grupos de fuentes, unas que se habilitan con solo colocar en uno el bit GIE, y otras que además necesitan que este puesto a uno el bit PEIE. En algunas versiones de los 16FXX solo existe el primer grupo. Además, cada fuente de interrupciones tiene su respectivo bit de habilitación individual.

Las fuentes de interrupción varían con cada versión, y pueden ser por ejemplo:

- Interrupción externa por pin RB0/INT
- Desborde del Timer 0 o RTCC
- Cambio en el estado de los bits 4 a 7 del puerto B
- Desborde del timer 1
- Desborde del timer 2
- Interrupción del capture/compare 1
- Interrupción del capture/compare 2
- transmisión o recepción de un caracter por la interface serie síncrona
- transmisión o recepción de un caracter por la interface serie asíncrona
- Fin de conversión A/D
- Lectura/escritura del puerto paralelo de comunicación con otros microprocesadores
- Escritura de EEPROM finalizada

#### **2.2.2.16 Programa fuente:**

El programa fuente esta compuesto por una sucesión de líneas de programa. Cada línea de programa esta compuesta por 4 campos separados por uno o más espacios o tabulaciones. Estos campos son:

<b>[Etiqueta]</b>	<b>Comando</b>	<b>[Operando(s)]</b>	<b>[:Comentario]</b>
-------------------	----------------	----------------------	----------------------

La etiqueta es opcional. El comando puede ser un mnemónico del conjunto de instrucciones. El operando esta asociado al comando, si no hay comando no hay operando, e inclusive algunos comandos no llevan operando. El comentario es opcional para el compilador aunque es buena práctica considerarlo obligatorio para el programador [21].

La etiqueta, es el campo que empieza en la primer posición de la línea no se pueden insertar espacios o tabulaciones antes de la etiqueta sino será considerado comando. Identifica la línea de programa haciendo que el compilador le asigne un valor automáticamente. Si se trata de una línea cuyo comando es una instrucción de programa del microcontrolador, se le asigna el valor de la dirección de memoria correspondiente a dicha instrucción (location counter). En otros casos se le asigna un valor de una constante, o la dirección de una variable, o será el nombre de una macroinstrucción, etc.

El comando puede ser un código mnemónico de instrucción del microcontrolador, o una directiva o pseudoinstrucción para el compilador. En el primer caso será directamente traducido a código de maquina, en el segundo caso será interpretado por el compilador y realizara alguna acción en tiempo de compilación como ser asignar un valor a una etiqueta, etc.

El campo de parámetros puede contener uno o más parámetros separados por comas. Los parámetros dependen de la instrucción o directiva. Pueden ser números o literales que representen constantes o direcciones.

El campo de comentario debe comenzar con un caracter punto y coma no necesita tener espacios o tabulaciones separándolo del campo anterior, e incluso puede empezar en la primera posición de la línea. El compilador ignora todo el texto que contenga la línea después de un caracter punto y coma. De esta manera pueden incluirse líneas que contengan solo comentarios, y es muy buena práctica hacer uso y abuso de esta posibilidad para que los programas resulten autodocumentados.

### **2.2.2.17 Conjunto de instrucciones**

El conjunto de instrucciones de los microprocesadores PIC 16FXX consiste en un pequeño repertorio de solo 33 instrucciones de 12 bits, que pueden ser agrupadas para su estudio en tres a cinco grupos. En este curso se ha optado por clasificarlas, desde el punto de vista del programador, en cinco categorías bien definidas de acuerdo con la función y el tipo de operandos involucrados. En primer lugar se agrupan las instrucciones que operan con bytes y que involucran algún registro de la memoria interna. En segundo lugar se analizarán las instrucciones que operan solo sobre el registro W y que permiten cargarle una constante implícita o incluida literalmente en la instrucción (literales). En tercer lugar se agrupan las instrucciones que operan sobre bits

individuales de los registros de la memoria interna. En cuarto lugar se clasifican las instrucciones de control de flujo del programa, es decir las que permiten alterar la secuencia lineal de ejecución de las instrucciones. Por último se agrupan unas pocas instrucciones que llamaremos especiales, cuyas funciones o tipos de operandos son muy específicos y no encajan en ninguna de las clasificaciones anteriores [22].

### 2.2.2.18 Instrucciones de Byte que operan con Registros

Estas instrucciones pueden ser de simple o doble operando de origen. El primer operando de origen será siempre el registro seleccionado en la instrucción, el segundo, en caso de existir, será el registro W. El destino, es decir donde se guardara el resultado, será el registro seleccionado o el W, según se seleccione con un bit de la instrucción.

El formato genérico de estas instrucciones es el siguiente:

0	1	2	3	4	5	<b>6</b>	<b>7</b>	<b>8</b>	<b>9</b>	<b>10</b>	<b>11</b>
						<b>d</b>	<b>f</b>	<b>F</b>	<b>f</b>	<b>f</b>	<b>f</b>

Los bits 0 a 4 (5 bits), denominados “f” permiten seleccionar uno de 32 registros de la memoria interna. El bit 5, denominado “d”, permite especificar el destino del resultado. Si  $d = 1$  el resultado se guardara en el registro

seleccionado. Si  $d = 0$  el resultado se guardara en W. Los bits 6 a 11 identifican la instrucción específica a realizar.

Las instrucciones siguientes son las tres operaciones lógicas de doble operando:

**ANDWF f,d ;operación AND lógica, destino = W f**  
**IORWF f,d ;operación OR lógica, destino = W f**  
**XORWF f,d ;operación XOR lógica, destino = W f**

Los nombres mnemónicos de estas instrucciones provienen de: AND W con F, Inclusive OR W con F y XOR W con F.

Las que siguen son las cuatro operaciones aritméticas y lógicas sencillas de simple operando:

**MOVF f,d ;movimiento de datos, destino = f**  
**COMF f,d ;complemento lógico, destino = NOT f**  
**INCF f,d ;incremento aritmético, destino = f + 1**  
**DECF f,d ;decremento aritmético, destino = f - 1**

Los mnemónicos de estas instrucciones provienen de: MOVE File, COMplement File, INCrement File y DECrement File.

En las siete instrucciones anteriores el único bit afectado de la palabra de estado del procesador es el Z, que se pone en 1 si el resultado de la operación es 00000000, y se pone en 0 si el resultado tiene cualquier otro valor.

A continuación siguen las dos instrucciones de rotación de bits a través del CARRY:

**RLF f,d ;rotación a la izquierda, destino = f ROT**

**RRF f,d ;rotación a la derecha, destino = f ROT**

En estas operaciones (Rotate Left File y Rotate Right File) los bits son desplazados de cada posición a la siguiente, en sentido derecho o izquierdo. El desplazamiento es cerrado, formando un anillo, con el bit C (CARRY) de la palabra de estado.

En estas dos instrucciones, el único bit afectado de la palabra de estado del procesador es el bit C, que tomará el valor que tenía el bit 7 o el bit 0, según sea el sentido del desplazamiento.

Estas instrucciones son muy útiles para la manipulación de bits, y además para realizar operaciones aritméticas, ya que en numeración binaria, desplazar un número a la izquierda es equivalente a multiplicarlo por 2, y hacia la derecha, a dividirlo por 2.

La instrucción siguiente realiza el intercambio de posiciones entre los cuatro bits menos significativos y los cuatro más significativos (nibble bajo y nibble alto).

**SWAPF f,d ;intercambia nibbles, destino = SWAP f**

Esta instrucción (SWAP File) no afecta ninguno de los bits de la palabra de estado del procesador.

Esta instrucción es muy útil para el manipuleo de números BCD empaquetados, en los que en un solo byte se guardan dos dígitos BCD (uno en cada nibble)

Las dos operaciones que siguen son la suma y la resta aritméticas :

**ADDWF f,d ;suma aritmética, destino = f + W**

**SUBWF f,d ;resta aritmética, destino = f - W**

Estas operaciones (ADD W a F y SUBstract W de F) afectan a los tres bits de estado C, DC y Z.

El bit Z se pone en 1 si el resultado de la operación es 00000000, y se pone en 0 si el resultado tiene cualquier otro valor.

La suma se realiza en aritmética binaria pura sin signo. Si hay un acarreo del bit 7, es decir que el resultado es mayor que 255, el bit C (carry) resulta 1, en caso contrario resulta 0. Si hay un acarreo del bit 3, es decir que la suma de las dos mitades (nibbles) menos significativas (bits 0 a 3) resulta mayor que 15, se pone en 1 el bit DC (digit carry), en caso contrario se pone en 0.

Ejemplos :

1010 0010	1101 0000
+ 0100 1111 <b>C DC Z</b>	+ 0110 1111 <b>C DC Z</b>
1111 0001 0 1 0	0011 1111 1 0 0

La resta se realiza sumando, en binario puro sin signo, el registro f más el complemento a dos de W (el complemento a 1, o complemento lógico, más 1)

Ejemplos :

f	0100 0100	0010 1000
W	- <u>0010 1000</u> <b>C DC Z</b>	- <u>0100 0100</u> <b>C DC Z</b>
	0001 1100 1 0 0	1110 0100 0 1 0

equivalente a :

f	0100 0100	0010 1000
comp 2W	+ <u>1101 1000</u> <b>C DC Z</b>	+ <u>1011 1100</u> <b>C DC Z</b>
	0001 1100 1 0 0	1110 0100 0 1 0

Los bits de estado C y DC toman el valor normal correspondiente a la suma de f con el complemento a 2 de W. De esta manera el significado para la operación de resta resulta invertido, es decir que C (carry) es 1 si no hubo desborde en la resta, o dicho de otra manera, si el contenido de W es menor que el de f. El bit DC se comporta de manera similar, es decir que DC es 1 si no hubo desborde en la mitad menos significativa, lo que equivale a decir que el nibble bajo del contenido de W es menor que el del registro f [21].

Las instrucciones que siguen son de simple operando, pero son casos especiales ya que el destino es siempre el registro seleccionado :



**CLRF f ;borrado de contenido, f = 0**

**MOVWF f ;copia contenido W f, f = W**

La instrucción CLRF (CLear File) afecta solo al bit Z que resulta siempre 0.

La instrucción MOVWF (MOVeW a F) no afecta ningún bit de la palabra de estado [22].

### 2.2.2.19 Instrucciones de Byte que operan sobre W y Literales

Estas instrucciones se refieren todas al registro W , es decir que uno de los operandos de origen y el operando de destino son siempre el registro W .En las instrucciones de este grupo que tienen un segundo operando de origen, este es siempre una constante de programa literalmente incluida en la instrucción, llamada constante literal o simplemente literal.

El formato genérico de estas instrucciones es el siguiente :

0	1	2	3	4	5	6	7	8	9	10	11
				k	k	k	k	k	k	k	k

Los bits 0 a 7 especifican la constante literal de 8 bits que se utilizara en la operación.

Las tres instrucciones que siguen son las operaciones lógicas tradicionales, similares a las que ya vimos anteriormente, pero realizadas entre una constante de programa y el registro W :

**IORLW k ; operación OR lógica,  $W = W \vee k$**

**ANDLW k ; operación AND lógica,  $W = W \wedge k$**

**XORLW k ; operación XOR lógica,  $W = W \oplus k$**

En estas tres instrucciones (Inclusive OR LiteralW , AND LiteralW y XOR LiteralW ) el único bit afectado de la palabra de estado del procesador es el Z , que se pone en 1 si el resultado de la operación es 00000000 , y se pone en 0 si el resultado tiene cualquier otro valor.

La instrucción que sigue sirve para cargar una constante de programa en el registroW :

**MOVLW k ; carga constante en W,  $W = K$**

Esta (MOVE LiteralW ) instrucción no afecta ninguno de los bits de estado del procesador.

La instrucción que sigue (CLEARW ) no correspondería incluirla en este grupo, y pertenece en realidad al primero, el de las instrucciones que operan sobre registros, ya que se trata de un caso especial de la instrucción CLRF , con destino W , y f = 0. La incluimos aquí porque como se le ha asignado un mnemónico particular referido específicamente al registro W , creemos que, desde el punto de vista del programador, es más útil verla dentro del grupo de instrucciones referidas aW .

**CLRW ; borra el contenido de W,  $W = 0$**

Al igual que en la instrucción CLRF, el único bit de estado afectado es el Z que resulta 1.

### 2.2.2.20 Instrucciones de Bit

El formato genérico de estas instrucciones es el siguiente :

0	1	2	3	4	5	6	7	8	9	10	11
				<b>b</b>	<b>b</b>	<b>b</b>	<b>f</b>	<b>f</b>	<b>f</b>	<b>f</b>	<b>f</b>

Los bits 0 a 4 (5 bits), denominados “f”, permiten seleccionar uno de 32 registros de la memoria interna. Los bits 5 a 7, denominados “b”, permiten especificar el número de bit (0 a 7) sobre el que se operará. Estas instrucciones operan solamente sobre el bit especificado, el resto de los bits del registro no son alterados. Estas instrucciones no tienen especificación de destino, ya que el mismo es siempre el registro seleccionado.

**BCF f,b ;borra el bit b de f ;bit f(b) = 0**

**BSF f,b ;coloca en uno el bit b de f ;bit f(b) = 1**

Estas instrucciones (BitClearFile y BitSetFile) no afectan ningún bit de la palabra de estado del procesador.

### 2.2.2.21 Instrucciones de Control

**GOTO k ;salto a la posición k (9 bits) del programa**

Esta es la típica instrucción de salto incondicional a cualquier posición de la memoria de programa (que en la mayoría de los microprocesadores convencionales se llama JUMP). La constante literal  $k$  es la dirección de destino del salto, es decir la nueva dirección de memoria de programa a partir de la cual comenzarán a leerse las instrucciones después de ejecutar la instrucción GOTO. Esta instrucción simplemente carga la constante  $k$  en el registro PC (contador de programa). La única complicación de esta instrucción es que la constante  $k$  es de solo 9 bits, mientras que el registro PC es de 11 bits, ya que en el 16C57 debe permitir direccionar una memoria de programa de 2 K. Los dos bits faltantes, bit 9 y 10 del PC, son tomados respectivamente de los bits de selección de página PA0 y PA1 de la palabra de estado. Este comportamiento particular hace que la memoria de programa aparezca como dividida en páginas de 512 posiciones como se verá más adelante. El programador debe tener en cuenta que antes de ejecutar una instrucción GOTO es posible que haya que programar los bits PA0 y PA1 [21].

La que sigue es la instrucción de llamado a subrutina:

### **CALL $k$ ; salto a la subrutina en la posición $k$ (8 bits)**

Su comportamiento es muy similar al de la instrucción GOTO, salvo que además de saltar guarda en el stack la dirección de retorno de la subrutina (para la instrucción RETLW). Esto lo hace simplemente guardando en el stack una copia del PC incrementado, antes de que el mismo sea cargado con la nueva dirección  $k$ . La única diferencia con la instrucción GOTO respecto de la forma en la que se realiza el salto, es que en la instrucción CALL la constante  $k$  tiene

so lo 8 bits en vez de 9. En este caso también se utilizan PA0 y PA1 para cargar los bits 9 y 10 del PC, pero además el bit 8 del PC es cargado siempre con 0. Esto hace que los saltos a subrutina solo puedan realizarse a posiciones que estén en las primeras mitades de las páginas mencionadas. El programador debe tener en cuenta este comportamiento y asegurarse de ubicar las posiciones de inicio de las subrutinas en las primeras mitades de las páginas.

La instrucción que aparece a continuación es la de retorno de subrutina:

**RETLW k ;retorno de subrutina con constante k, W = k**

Esta (RETurn con Literal in W) instrucción produce el retorno de subrutina con una constante literal k en el registro W. La operación que realiza consiste simplemente en sacar del stack un valor y cargarlo en el PC. Ese valor es el PC incrementado antes de realizar el salto, de la última instrucción CALL ejecutada, por lo tanto es la dirección de la instrucción siguiente a dicho CALL. Dado que el stack es de 11 bits, el valor cargado en el PC es una dirección completa, y por lo tanto se puede retornar a cualquier posición de la memoria de programa, sin importar como estén los bits de selección de página. Esta instrucción además carga siempre una constante literal en el registro W. Ya que esta es la única instrucción de retorno de subrutina de los PIC16FXX, no hay en estos microprocesadores forma de retornar de una subrutina sin alterar el registro W. Por otro lado, y con un método lógico especial de programación, un conjunto de sucesivas instrucciones RETLW puede ser usado como una tabla de valores constantes incluida en el programa (Ej.: tablas BCD / 7 seg., hexa/ASCII, etc.) [22].

A continuación se presentan las dos únicas instrucciones de “salteo” (skip) condicional. Estas instrucciones son los únicos medios para implementar bifurcaciones condicionales en un programa. Son muy generales y muy poderosas ya que permiten al programa tomar decisiones en función de cualquier bit de cualquier posición de la memoria interna de datos, y eso incluye a los registros de periféricos, los puertos de entrada/salida e incluso la palabra de estado del procesador. Estas dos instrucciones reemplazan y superan a todo el conjunto de instrucciones de salto condicional que poseen los microprocesadores sencillos convencionales (salto por cero, por no cero, por carry, etc.).

**BTFSK f,b ;salteo si bit = 0, bit = f(0) saltea**

**BTFSK f,b ;salteo si bit = 1, bit = f(1) saltea**

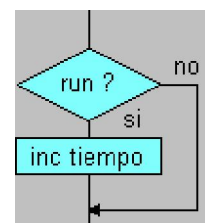
BTFSK (Bit Test File and Skip if Clear) saltea la próxima instrucción si el bit b del registro f es cero. La instrucción BTFSK (Bit Test File and Skip if Set) saltea si el bit es 1. Estas instrucciones pueden usarse para realizar o no una acción según sea el estado de un bit, o, en combinación con GOTO, para realizar una bifurcación condicional.

### Ejemplo 1 :

```

-----
-----
btfsk flags,run ;sí ha arrancado el reloj
incf tiempo ;incremento contador de
tiempo

```

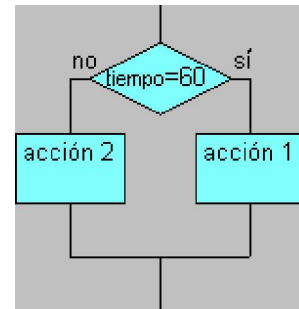


## Ejemplo 2:

```

movf tiempo,w    ;testeo por tiempo = 60
xorlw 60
btfss STATUS,Z
goto acc_2        ;salto si tiempo <> 60

```



```

----- acción 1

```

```

goto acc_fin

```

```

acc_2

```

```

----- acción 2

```

```

acc_fin  acá se unen los caminos

```

Las instrucciones que siguen son casos especiales de las de incremento y decremento vistas anteriormente. Estas instrucciones podrían categorizarse dentro del grupo de instrucciones orientadas a byte sobre registros (primer grupo), ya que efectivamente operan sobre los mismos, y el formato del código

de la instrucción responde al de ese grupo, pero, a diferencia de las otras, pueden además alterar el flujo lineal del programa y por eso se les incluyó en este grupo.

**DECFSZ f,d ;decrementa y saltea si 0, destino= f - 1, = 0 saltea**

**INCFSZ f,d ;incrementa y saltea si 0, destino= f + 1, = 0 saltea**

Estas dos instrucciones (Decrement File and Skip if Zero, e Increment File and Skip if Zero) se comportan de manera similar a DECF e NCF, salvo que no afectan a ningún bit de la palabra de estado. Una vez realizado el incremento o decremento, si el resultado es 00000000, el microprocesador saltará la próxima instrucción del programa. Estas instrucciones se utilizan generalmente en combinación con una instrucción de salto (GOTO), para el diseño de ciclos o lazos (loops) de instrucciones que deben repetirse una cantidad determinada de veces.

### **Ejemplo:**

```

    clrf 10          ;pongo cero en la posición 10 de la memoria interna
loop      ;lo que sigue se ejecutará 256 veces
.....
.....
.....

    incfsz 10,1      ;incremento la posición 10 hasta que llegue a 0
    goto loop        ;si no llego a cero voy a repetir la secuencia
                    ;cuando llegue a cero salteo el goto
..... y sigue la continuación del programa
    
```



.....

.....

### **2.2.2.22 Instrucciones Especiales**

En este grupo se reunieron las instrucciones que controlan funciones específicas del microprocesador o que actúan sobre registros especiales no direccionados como memoria interna normal.

La instrucción que sigue es la típica NO OPERATION, existente en casi todos los microprocesadores.

#### **NOP ;no hace nada, consume tiempo**

Esta instrucción solo sirve para introducir una demora en el programa, equivalente al tiempo de ejecución de una instrucción. No afecta ningún bit de la palabra de estado.

La siguiente es una instrucción específica de control de los puertos de entrada/salida.

#### **TRIS f ;carga el tristate control, TRISf = W**

Esta instrucción (TRISf) carga el registro de control de los buffers tristate de un puerto de entrada salida (data dirección register), con el valor contenido en W. El parámetro f debe ser la dirección de memoria interna del

puerto, aunque el valor  $W$  no será cargado en el puerto sino en el registro de tristate del mismo.

La siguiente instrucción sirve para programar el registro `OPTION` que controla el RTCC y prescaler

### **OPTION ;carga el registro OPTION, OPTION = W**

El registro `OPTION` no es accesible como memoria interna y solo se lo puede programar con esta instrucción. Esta instrucción no afecta ningún bit de la palabra de estado.

La instrucción que sigue borra el contador del watch dog timer. Este registro tampoco es accesible como memoria, y esta es la única instrucción que lo modifica.

### **CLRWDT ;borra el watch dog timer, WDT = 0**

Esta instrucción, además, coloca en uno los bits `PD` (power down) y `TO` (time-out) de la palabra de estado.

La siguiente es una instrucción especial de control del microcontrolador que lo pone en el modo power down. En este modo el microprocesador se detiene, el oscilador se apaga, los registros y puertos conservan su estado, y el consumo se reduce al mínimo. La única forma de salir de este estado es por medio de un reseto por time-out del watch dog timer.

### **SLEEP ;coloca el $\mu$ C en modo sleep, WDT = 0**

Esta instrucción, además, borra el bit PD (power down) y setea el bit TO (time-out) de la palabra de estado.

### 2.2.2.23 Resumen de instrucciones (clasificación según el fabricante en tres grupos):

#### *Instrucciones orientadas a byte:*

Mnemonic, Operands	Description	Cycles	12-Bit Opcode		Status Affected
			MSb	LSb	
ADDWF f,d	Add W and f	1	0001	11df ffff	C,DC,Z
ANDWF f,d	AND W with f	1	0001	01df ffff	Z
CLRF f	Clear f	1	0000	011f ffff	Z
CLRW –	Clear W	1	0000	0100 0000	Z
COMF f,d	Complement f	1	0010	01df ffff	Z
DECF f,d	Decrement f	1	0000	11df ffff	Z
DECFSZ f,d	Decrement f, Skip if 0	1(2)	0010	11df ffff	None
INCF f,d	Increment f	1	0010	10df ffff	Z
INCFSZ f,d	Increment f, Skip if 0	1(2)	0011	11df ffff	None
IORWF f,d	Inclusive OR W with f	1	0001	00df ffff	Z
MOVF f,d	Move f	1	0010	00df ffff	Z
MOVWF f	Move W to f	1	0000	001f ffff	None
NOP –	No Operation	1	0000	0000 0000	None
RLF f,d	Rotate left f through Carry	1	0011	01df ffff	C
RRF f,d	Rotate right f through Carry	1	0011	00df ffff	C
SUBWF f,d	Subtract W from f	1	0000	10df ffff	C,DC,Z
SWAPF f,d	Swap f	1	0011	10df ffff	None
XORWF f,d	Exclusive OR W with f	1	0001	10df ffff	Z

#### *Instrucciones orientadas a bit:*

Mnemonic, Operands	Description	Cycles	12-Bit Opcode		Status Affected
			MSb	LSb	
BCF f,b	Bit Clear f	1	0100	bbbf ffff	None
BSF f,b	Bit Set f	1	0101	bbbf ffff	None
BTFSC f,b	Bit Test f, Skip if Clear	1 (2)	0110	bbbf ffff	None
BTFSS f,b	Bit Test f, Skip if Set	1 (2)	0111	bbbf ffff	None

#### *Instrucciones orientadas a literal y control:*

Mnemonic, Operands	Description	Cycles	12-Bit Opcode			Status Affected
			MSb	LSb		
ANDLW k	AND literal with W	1	1110	kkkk	kkkk	Z
CALL k	Call subroutine	2	1001	kkkk	kkkk	None
CLRWDT k	Clear Watchdog Timer	1	0000	0000	0100	$\overline{TO}$ , $\overline{PD}$
GOTO k	Unconditional branch	2	101k	kkkk	kkkk	None
IORLW k	Inclusive OR Literal with W	1	1101	kkkk	kkkk	Z
MOVLW k	Move Literal to W	1	1100	kkkk	kkkk	None
OPTION k	Load OPTION register	1	0000	0000	0010	None
RETLW k	Return, place Literal in W	2	1000	kkkk	kkkk	None
SLEEP —	Go into standby mode	1	0000	0000	0011	$\overline{TO}$ , $\overline{PD}$
TRIS f	Load TRIS register	1	0000	0000	0fff	None
XORLW k	Exclusive OR Literal to W	1	1111	kkkk	kkkk	Z

esta tabla de resumen del conjunto de instrucciones se pueden observar los mnemónicos, la explicación, el número de ciclos, el código de máquina y los bits afectados del registro STATUS para cada una de las instrucciones [21].

### 2.2.3 Modos de direccionamiento.

#### 2.2.3.1 Direccionamiento de la memoria de datos (RAM)

La memoria interna se direcciona en forma directa por medio de los 5 bits “f” contenidos en las instrucciones que operan sobre registros. De esta manera se puede direccionar cualquier posición desde la 00 a la 1F. Como se vio en el capítulo correspondiente a los mapas de memoria, las direcciones 10 a 1F corresponden a los bancos de registros, por lo tanto, en los microcontroladores que tengan más de un banco, antes de acceder a alguna variable que se encuentre en esta zona, el programador deberá asegurarse de haber programado los bits de

selección de banco en el registro FSR . Los registros especiales y de uso general de la posición 00 a la 0f están presentes en todos los PIC16FXX , al igual que el banco 0 de registros. Los bancos 1, 2 y 3 de registros están presentes solo en el 16C57 [30].

El registro FSR , además de servir para seleccionar el banco activo , sirve como puntero para direccionamiento indirecto . La posición 00 del mapa de RAM es la llamada dirección indirecta. Si en cualquier instrucción se opera con la dirección 00 , en realidad se estará operando con la dirección a donde apunte el contenido del FSR . Por ejemplo si el FSR contiene el valor 14 , una instrucción que opere sobre la dirección 0 , operara en realidad sobre la dirección 14 . Se puede decir en este ejemplo que la posición 14 de memoria fue direccionada en forma indirecta a través del puntero FSR .

***Ejemplo :***

; Esta porción de programa borra 5 posiciones de memoria a partir de la dirección 12

FSR equ 04 ;(definición al comienzo del programa)

.....

.....

movlw 5 ;prepara para repetir 5 veces

movwf 08 ;(el registro 08 es el contador del bop)

movlw 12h ;apunta a la dirección 12h

```

mowf FSR ;
bop:
    clrf 0      ;borra una posición de memoria
    incf FSR    ;apunta a la siguiente
    decfsz 08   ;si todavía no borra todas
    goto bop    ;sigue borrando

```

.....

.....

El direccionamiento indirecto es muy útil para el procesamiento de posiciones consecutivas de memoria, como en el ejemplo, o para el direccionamiento de datos en subrutinas.

### 2.2.3.2 Direccionamiento de la memoria de programa (EPROM, OTP)

La instrucción GOTO dispone solo de 9 bits en el código de operación para especificar la dirección de destino del salto. Al ejecutar una instrucción GOTO el microprocesador toma los dos bits que restan para completar la dirección de 11 bits, de los bits 5 y 6 de la palabra de estado. Estos últimos son llamados bits de selección de página (PA0 y PA1). El programador deberá asegurarse de que estos dos bits tengan el valor correcto antes de toda instrucción GOTO. Ver figura 2.11

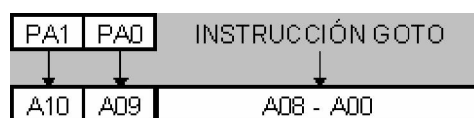


Figura 2.11 Direccionamiento directo con instrucción GOTO

Deberá tenerse en cuenta además que es posible avanzar de una página a otra en forma automática cuando el PC se incrementa. Esto ocurre si el programa empieza en una página y sigue en la siguiente. Sin embargo, al incrementarse el PC desde la última posición de una página a la primera de la siguiente, **los bits PA0 y PA1 no se modifican**, y por lo tanto si se ejecuta una instrucción GOTO, CALL o alguna que actúe sobre el PC, esta producirá un salto a la página anterior, a menos que el programador tenga la precaución de actualizar el valor de dichos bits. Por este motivo es conveniente dividir el programa en módulos o rutinas que estén confinados a una página.

En el caso de la instrucción CALL, el direccionamiento se complica un poco más, ya que la misma solo dispone de 8 bits para especificar la dirección de destino salto. En este caso también se utilizan los mismos bits de selección de página para completar los bits décimo y decimoprimero de la dirección, pero falta el noveno bit. En estas instrucciones este bit se carga siempre con 0, lo que implica que solo se pueden realizar saltos a subrutina a las mitades inferiores de cada página. En este caso también el programador deberá asegurarse que el estado de los bits PA0 y PA1 sea el correcto al momento de ejecutarse la instrucción.



Figura 2.12 Direccionamiento directo con instrucción CALL

Las instrucciones que operan sobre el PC como registro y alteran su contenido provocando un salto, responden a un mecanismo muy similar al de las instrucciones CALL para la formación de la dirección de destino. En este caso los bits 0 a 7 son el resultado de la instrucción, el bit 8 es 0 y los bits restantes se toman de PA0 y PA1 [30].

Este mecanismo se llama paginado, y a pesar de que representa una complicación bastante molesta para el programador, resulta muy útil ya que permite ampliar la capacidad de direccionamiento de memoria de programa para las instrucciones de salto.

### **2.3 Ordenador.**

La aplicación desarrollada por el ordenador debe ser capaz de leer los datos de las adquisiciones realizadas. Estos datos deben poder ser almacenados en un soporte físico para independizarlos de la aplicación. Además opcionalmente la aplicación podría recuperarlos del soporte físico y además mostrarlos al usuario y de manera gráfica.

Por tanto el ordenador también debe poseer los niveles físicos que soporta el USB.

### **2.4 Protocolo de Comunicaciones.**



El protocolo de comunicación implementado deberá desarrollarse en tres etapas, la primera la comunicación entre los pic, la segunda entre el pic y el pc y la tercera entre el pic y los sensores.

La comunicación entre los pic será USART ya que no todos los pic soportan USB o la capacidad de memoria no permite otra aplicación, la comunicación entre el pic y el ordenador se utilizará USB tipo BULK asincrónico, y protocolo I2C entre el pic y los sensores.

#### **2.4.1 Protocolo de Comunicaciones USB.**

##### **2.4.2.1 Características generales del USB**

La especificación del USB [42] proporciona una serie de características que pueden ser distribuidas en categorías. Estas características son comunes para todas las versiones (desde la 1.0 hasta la 2.0)

##### *Fácil uso para los usuarios :*

- Modelo simple para el cableado y los conectores
- Detalles eléctricos aislados del usuario (terminaciones del bus)
- Periféricos auto-identificativos
- Periféricos acoplados y reconfigurados dinámicamente (HotSwappable)

### *Flexibilidad*

- Amplio rango de tamaños de paquetes, permitiendo variedad de opciones de buffering de dispositivos
- Gran variedad de tasas de datos de dispositivos acomodando el tamaño de buffer para los paquetes y las latencias
- Control de flujo para el manejo del buffer construido en el protocolo.

### *Ancho de banda isócrono*

- Se garantiza un ancho de banda y bajas latencias apropiadas para telefonía, audio, etc.
- Cantidad de trabajo isócrono que puede usar el ancho de banda completo del bus.
- Control de flujo para el manejo del buffer construido en el protocolo.

### *Amplia gama de aplicaciones y cargas de trabajo*

- Adecuando el ancho de banda desde unos pocos kbs hasta varios Mbs
- Soporta tanto el tipo de transferencia isócrono como el asíncrono sobre el mismo conjunto de cables.
- Conexiones múltiples, soportando operaciones concurrentes de varios dispositivos.
- Soporta hasta 127 dispositivos físicos.
- Soporta la transferencia de múltiples datos y flujos de mensajes entre el host y los dispositivos.

### *Robustez*

- Manejo de errores y mecanismos de recuperación ante fallos implementados en el protocolo.

- Inserción dinámica de dispositivos

- Soporte para la identificación de dispositivos defectuosos.

### *Implementación de bajo coste*

- Sub canal de bajo coste a 1.5 Mbs

- Conectores y cables de bajo coste

- Adecuado para el desarrollo de periféricos de bajo coste

En los siguientes apartados vamos a ver algunos de los detalles de cada una de las versiones que es lo que las diferencia unas de otras. Estas diferencias serán mas bien escasas en cuanto a características generales.

#### **2.4.2.2 Especificaciones técnicas**

En esta sección se describe algunos aspectos básicos en el funcionamiento de USB , tales como :

- La topología

- El flujo de datos

- La capa de protocolo

- La eléctrica

- La mecánica

##### ***a) La topología del bus***

El Universal Serial Bus conecta los dispositivos USB con el host USB.

La interconexión física USB es una topología de estrellas apiladas donde un hub es el centro de cada estrella. Cada segmento de cable es una conexión punto-a-punto entre el host y los hubs o función, o un hub conectado a otro hub o función.

El número máximo de dispositivos que puede conectar USB es de 127, pero debido a las constantes de tiempo permitidas para los tiempos de propagación del hub y el cable, el número máximo de capas permitido es de siete (incluida la capa raíz) con un máximo de longitud de cable entre el hub y el dispositivo de 5 metros. Cabe destacar que en siete capas, sólo se soportan cinco hubs que no sean raíz en una ruta de comunicación entre el host y cualquier dispositivo. Un dispositivo compuesto ocupa dos capas, por eso, no puede ser activado si está acoplado en la última capa de nivel siete. Sólo las funciones pueden ponerse en este nivel.

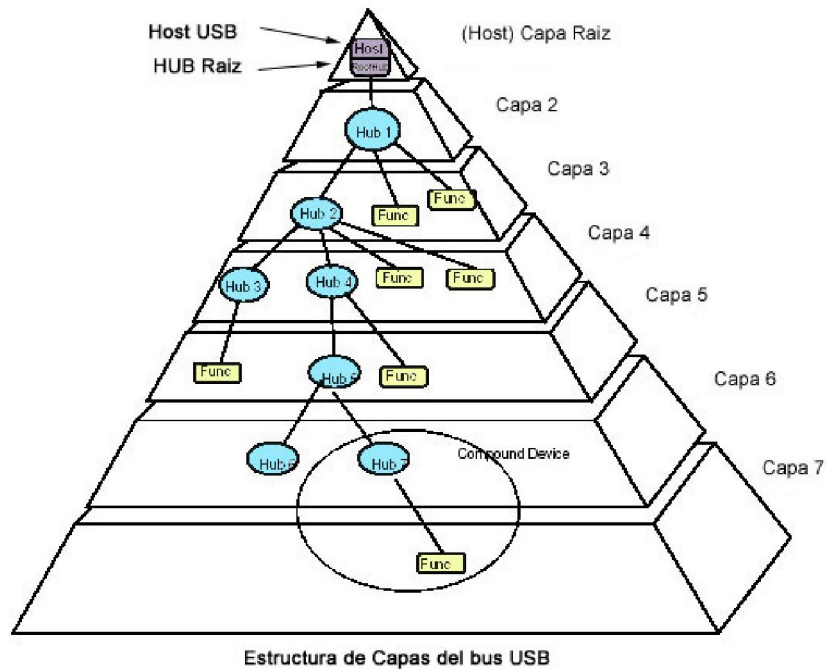


Figura 2.13 Estructura de capas del bus USB

La topología del bus USB se puede dividir en tres partes:

La capa física: Como están conectados los elementos físicamente

La capa lógica: Los roles y las responsabilidades de los elementos USB

La relación software del cliente-función: Como se ven mutuamente el software del cliente y los interfaces de las funciones relacionadas

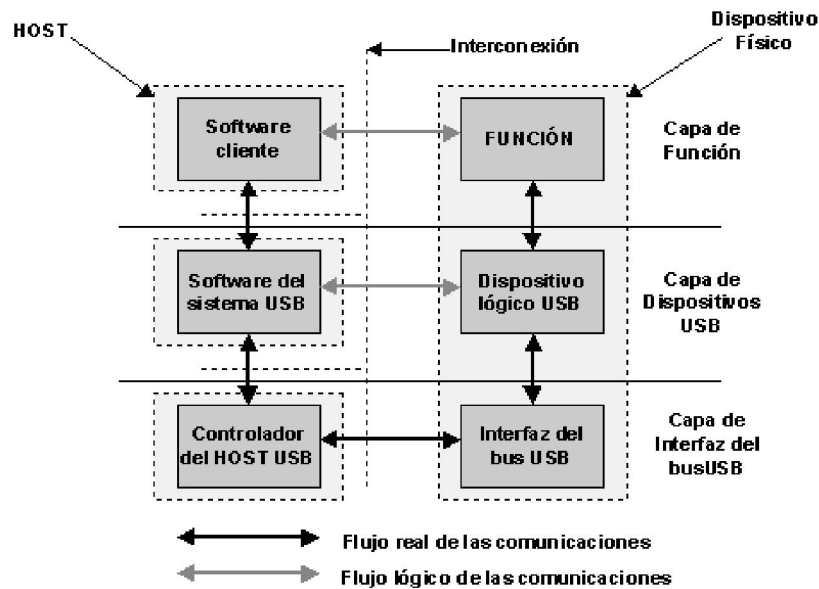


Figura 2.14 Direcciónamiento directo de flujo del USB

### ***La capa física***

La arquitectura física del USB se centra en las piezas de plástico y de metal con las que el usuario debe tratar para construir un entorno USB. Cada entorno físico USB está compuesto por cinco tipos de componentes:

El host

El controlador del host

Los enlaces

Los dispositivos

Los hubs

Los dispositivos están conectados físicamente al host a través de una topología en estrella, como se ilustra en la figura. Los puntos de acople están provistos de una clase de dispositivos USB llamados hubs, los cuales tienen

puntos de acople adicionales llamados puertos. Estos hubs se conectan a otros dispositivos a través de enlaces (cables de cuatro hilos).

El host proporciona uno o mas puntos de acople a través del hub raíz. Para prevenir los acoples circulares, se impone una estructura ordenada por capas en la topología de estrella y como resultado se obtiene una configuración al estilo de un árbol como se ve en la figura 2.15.

Todas las comunicaciones físicas son iniciadas por el host. Esto quiere decir que cada milisegundo, o en cada ventana de tiempo que el bus lo permita, el host preguntará por nuevos dispositivos en el bus USB. Además el host inicia todas las transacciones físicas y soporta todas las transferencias de datos sobre la capa física

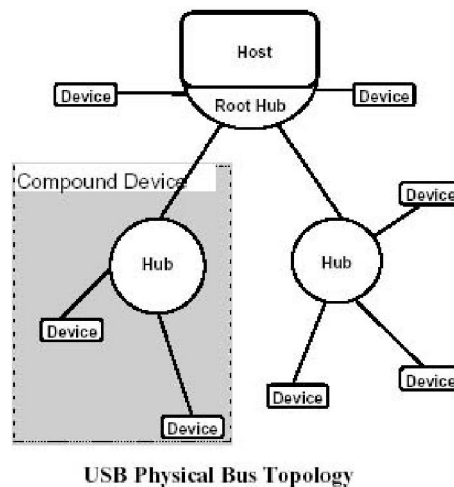


Figura 2.15 Topología física del bus USB

### ***El host USB***

El host es el sistema de computación completo, incluyendo el software y el hardware, sobre el cual se sostiene el USB.

El host tiene la habilidad de procesar y gestionar los cambios de configuración que puedan ocurrir en el bus durante su funcionamiento. El host gestiona el sistema y los recursos del bus como el uso de la memoria del sistema, la asignación del ancho de banda del bus y la alimentación del bus. El host también ayuda al usuario con la configuración automática de los dispositivos conectados y reaccionando cuando son desconectados.

Un host puede soportar uno o más buses USB. El host gestiona cada bus independientemente de los demás. Los recursos específicos del bus como el ancho de banda asignado son únicos a cada bus. Cada bus está conectado al host a través de un controlador del host.

Sólo hay un host en cualquier sistema USB. Desde el interfaz USB hasta el sistema de host del ordenador es lo que se le llama controlador de host y puede estar implementado como combinación de hardware, firmware o software. Integrado dentro del sistema de host hay un **hub raíz** que provee de un mayor número de puntos de acople al sistema.

Siempre que es posible, el software del USB usa el interfaz existente del sistema de host para gestionar las interacciones superiores. Por ejemplo, si un sistema de host usa la Gestión de Energía Avanzada (APM), el software del USB conecta al APM para interceptar, suspender las notificaciones.



### ***El controlador del host***

El controlador de host está formado por el hardware y el software que permite a los dispositivos USB ser conectados al host. Este controlador es el agente iniciador del bus, es decir es el que comienza las transferencias en el bus.

El controlador de bus es el maestro en un bus USB. Otros buses como PCI, permiten la presencia de múltiples maestros donde cada uno arbitra sus accesos al bus. En la arquitectura USB sólo hay un controlador de host por cada bus USB y por eso no hay arbitración para el acceso al bus.

Como las transferencias de datos de los dispositivos pueden ser basadas en datos o en la disponibilidad espacial del dispositivo, la mayoría de los controladores de host están implementados como dispositivos maestros de bus PCI. Esto permite al controlador de host iniciar una transferencia de datos en el bus del sistema cuando le sea necesario, sin requerir la intervención del host de la CPU para cada transferencia. El controlador se comporta como un bus maestro PCI multicanal programable para dar soporte a las necesidades de transferencia de datos de múltiples dispositivos conectados al bus USB.

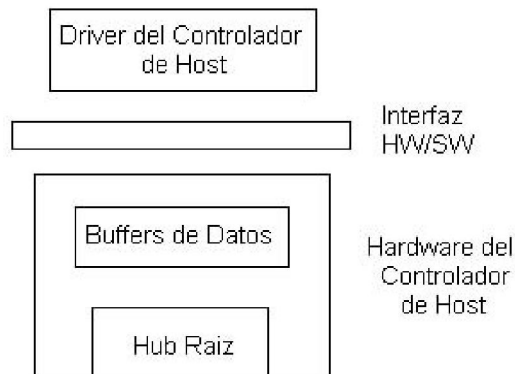


Figura 2.16 Vista conceptual del controlador USB

Esta figura muestra una vista conceptual del controlador de bus USB.

La parte software consiste en el driver del controlador de host (HCD).

Este software interactúa con el hardware del controlador de host a través de la interfaz hardware/software.

La parte hardware del controlador de host consiste en un hub raíz que proporciona los puertos USB y los buffers de datos (colas) donde son almacenadas cuando son movidas a/desde memoria.

El host USB interactúa con los dispositivos USB a través del controlador.

Las funciones básicas del controlador de host son:

- Detectar la inserción o desconexión de dispositivos USB
- Gestionar el flujo de control entre el host y los dispositivos
- Gestionar el flujo de datos entre el host y los dispositivos

- Coleccionar estadísticas de actividad y estado
- Proveer una cantidad limitada de energía a los dispositivos conectados

Hay dos implementaciones estandarizadas de la parte hardware de los controladores de host USB. Ambas proporcionan la misma funcionalidad y rendimiento para la interconexión. Estas implementaciones son:

El Universal Host Controller Interface (UHCI) definido por Intel

Open Host Controller Interface (OpenHCI o OHCI) definido por Microsoft.

UHCI está definido como que la parte software tiene una gran responsabilidad para mantener el hardware en funcionamiento. Esto permite a esta implementación ser relativamente simple y realizarse con un bajo número de puertas.

OHCI está definida como que la parte hardware tiene más responsabilidad por el mantenimiento del flujo de datos, para que la parte software tenga menos trabajo que hacer. Esta otra implementación tiende a ser más compleja y tiene una cantidad mayor de puertas que la UHCI

### ***Dispositivos USB***

Un dispositivo es una colección de funcionalidad que lleva a cabo algún propósito de utilidad. Por ejemplo, un dispositivo podría ser un ratón, un teclado, una cámara, etc. Puede haber múltiples dispositivos simultáneamente en el mismo bus. Cada dispositivo lleva consigo información que puede ser útil para identificar sus características. La información que describe al dispositivo se encuentra asociada con el canal de control. Esta información se divide en tres categorías:

**Estándar:** Esta es la información cuya definición es común a todos los dispositivos USB e incluye elementos como la identificación del fabricante, la clase, la gestión de energía.

**Clase:** La definición de esta información varía dependiendo del aparato. Es una clasificación de los dispositivos en cuanto a sus prestaciones.

**USB Vendor:** El fabricante del periférico puede poner aquí cualquier información deseada.

El software del host es capaz de determinar el tipo de dispositivo conectado haciendo uso de esta información y de un direccionamiento individual. Todos los dispositivos USB son accedidos por una dirección USB que es asignada dinámicamente cuando se conecta, asignándole también un número. Cada aparato soporta además uno o más canales a través de los cuales el host puede comunicarse con el dispositivo. Una vez ha sido reconocido e identificado el

dispositivo, el software del host puede hacer que los drivers del dispositivo apropiados obtengan el control del nuevo dispositivo conectado.

Cuando desconectamos el dispositivo, la dirección puede ser reutilizada para el próximo dispositivo conectado.

En cuanto a los tipos de dispositivos nos encontramos con dos clases:

Hubs, que proporcionan los puntos de acople adicionales al USB.

Funciones, que le dan al sistema la funcionalidad (HIDs, impresoras, unidades de almacenamiento...)

## HUBs

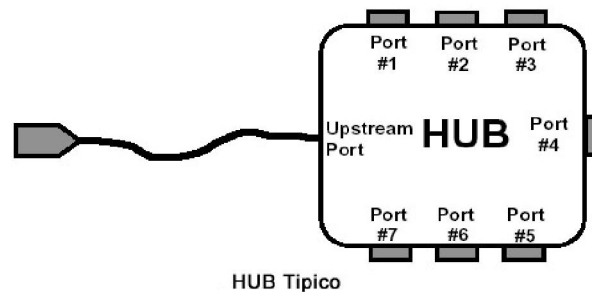


Figura 2.17 Típico HUB USB

Los hubs son un elemento clave en la arquitectura Plug and Play del USB, un bus tradicional puede ser dividido en segmentos de bus individuales conectados por puentes. Cada segmento tiene un número limitado de puntos de conexión debido a la limitada energía y la carga del bus. En la arquitectura USB, el número

de puntos de conexión de dispositivos se expande añadiendo dispositivos únicos llamados **HUBs** o concentradores. Estos dispositivos expanden la arquitectura del USB de dos formas:

Incrementando los puntos de conexión a través de puertos adicionales

Proporcionando energía a los dispositivos al expandir el bus.

Los HUBs sirven para simplificar la conectividad USB desde la perspectiva del usuario y proporcionar robustez y complejidad a un precio relativamente bajo. Según esta arquitectura, se pueden expandir el bus acoplando hubs adicionales, interconectando dichos hubs mediante enlaces. Cada hub convierte un punto simple de conexión en múltiples puntos, donde estos puntos de conexión se les llaman puertos.

### **Arquitectura Externa**

Un hub requiere tener un puerto de subida y de 1 a N de bajada.

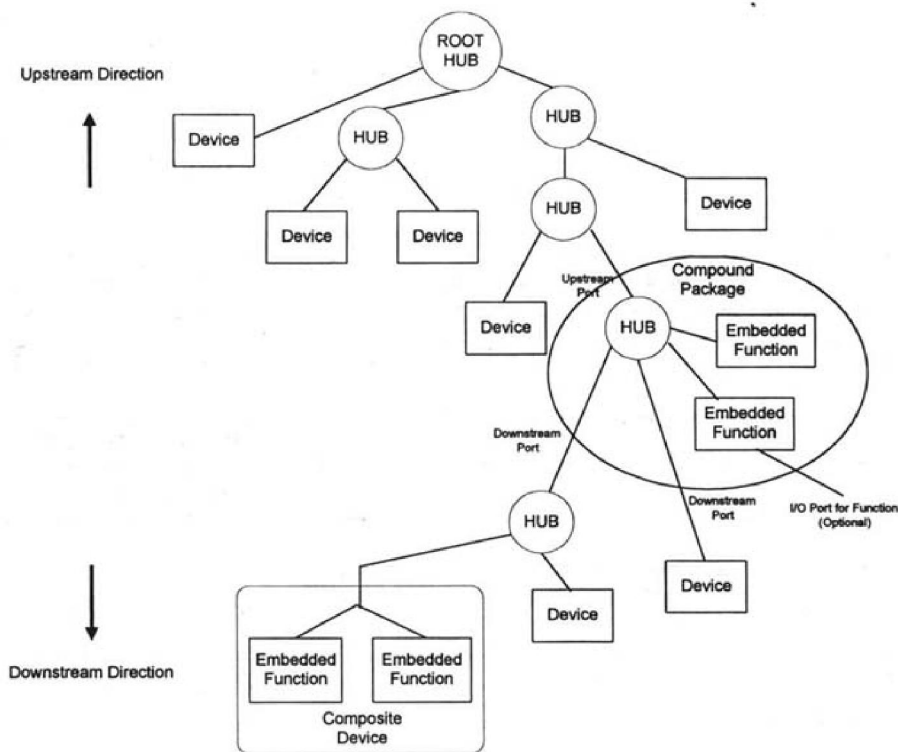


Figura 2.18 Arquitectura externa USB

El *puerto de subida* (upstream port) de un hub conecta el hub al host. Es el que eléctricamente está más cerca del controlador de host. Está numerado como el puerto 0.

Los *puertos de bajada* permiten la conexión a otro hub o a una función. Están numerados del 1 al N. Son los que más lejos están del controlador de host. Los hubs pueden detectar conexiones y desconexiones en cada puerto de bajada y activar la distribución de energía para cada dispositivo de bajada. Cada puerto puede ser individualmente activado para cada dispositivo high-full- o low-speed.

Los hubs tienen dos formas de obtención de la energía: A través del puerto de subida (bus-powered) o de una fuente externa (self-powered).

### Arquitectura Interna

Un hub USB 1.x consiste en:

El repetidor HUB, que es el responsable de gestionar la conectividad entre el puerto de subida y el de bajada, los cuales están operando a la misma velocidad.

El controlador del hub hace posible el acceso del hub al host y viceversa. Para la versión USB 2.0 hay un tercer elemento: Traductor de transacciones que proporciona los mecanismos que dan soporte a los dispositivos full-/low-speed tras el hub mientras se transmiten todos los datos entre el host y el hub en el modo hi-speed

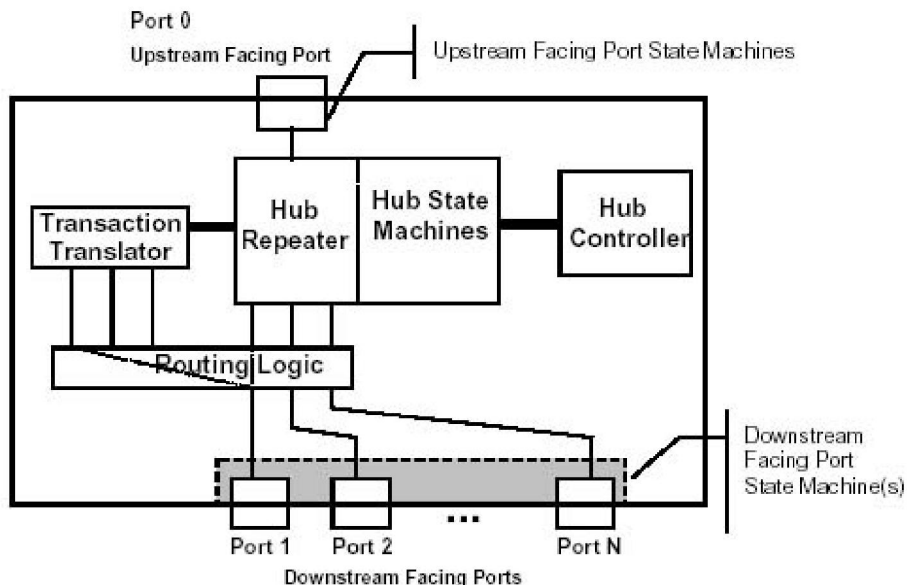


Figura 2.19 Esquema del HUB



Los hubs tienen diferente comportamiento en cuanto a conectividad dependiendo si están propagando el tráfico de paquetes, reanudando la señalización o están en estado idle.

Conectividad de Señalización de Paquetes (Fig 2.20 A); El repetidor del hub contiene un puerto que debe estar siempre conectado en la dirección de subida, que es la conexión al host y uno o más puertos de bajada (la de los dispositivos). La conectividad de subida es la conectividad hacia el host y la de bajada hacia el dispositivo. Un hub también tiene un estado idle durante el cual el hub no está conectado. En este estado, todos los puertos del hub están en modo recepción esperando empezar el siguiente paquete.

Si un puerto de bajada está activo (en un estado donde puede propagar la señal a través del hub) y el hub detecta el comienzo de un paquete en ese puerto, se establece la conexión en dirección al puerto de subida del hub, pero no a cualquier otro puerto de bajada. Esto significa que cuando un dispositivo o un hub transmite un paquete de subida, solo los hubs en línea entre el dispositivo que transmite y el host verán el paquete.

En la dirección de bajada, los hubs operan en modo broadcast. Cuando un hub detecta el comienzo de un paquete en su puerto de subida, establece la conexión a todos los puertos activos de bajada. Si un puerto no está activo no se propaga la señal de bajada por él.

Reanudación de la conexión (Fig 2 20 B); Los hubs tienen distinto comportamiento para la reanudación de la señal de subida y bajada. Un hub que está suspendido emite la señal de continuidad de su puerto de subida a todos sus puertos de bajada activos.

Si un hub está suspendido y detecta la reanudación de la señal de un puerto de bajada suspendido o activo, el hub emite esa señal de subida a todos sus puertos activos incluyendo el puerto que inició la secuencia de reanudación, pero no a los desactivados o suspendidos.

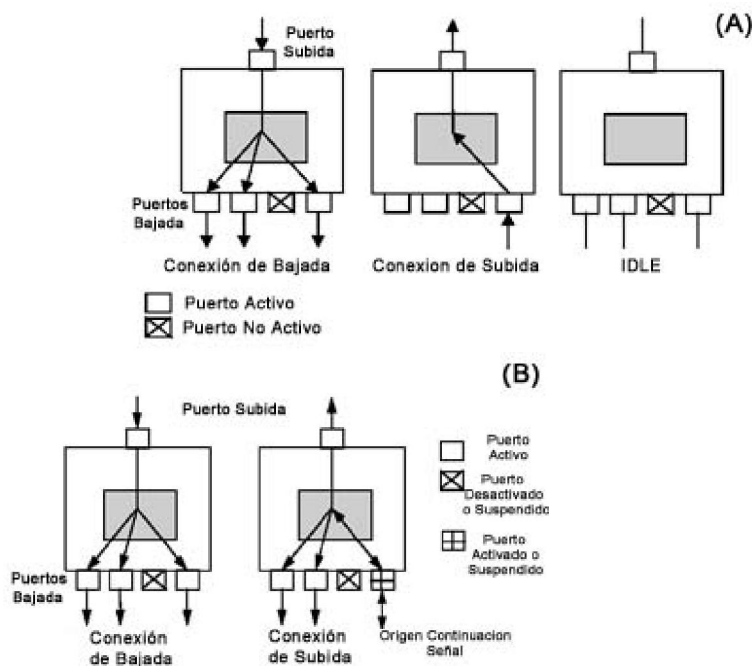


Figura 2 20 A ) sistema de conexión por paquetes B ) reactivación del sistema

## Funciones

Una función es un dispositivo USB que es capaz de transmitir y recibir datos o información de control sobre el bus. Típicamente se implementa como un periférico separado con un cable que se conecta en un puerto del hub. Sin embargo hay una gran flexibilidad a la hora de construir dispositivos. Una función simple pueden dar una funcionalidad simple (un microfono, unos altavoces...) o puede estar compuesto en distintos tipos de funcionalidad, como unos altavoces con un panel LCD. Este tipo de dispositivos se les llama función múltiples o composite device (dispositivo compuesto).

Otra forma de construir productos con múltiples funciones es creando un compound device (que significa también dispositivo compuesto). Este es el término usado cuando un hub está acoplado junto a múltiples dispositivos USB dentro de un mismo paquete. El usuario verá una sola unidad en el extremo del cable, pero internamente tiene un hub y varios dispositivos. Este tipo de "paquetes" tienen una dirección de bus para cada uno de los componentes, en contraposición a los composite devices que tienen una única dirección.

Un buen ejemplo de un dispositivo compuesto sería un teclado USB que tuviera una conexión adicional para ratón. A pesar de que el teclado es un periférico, en este caso se le puede acoplar un ratón y por supuesto se necesitaría de un hub interno en el teclado para que esto pueda funcionar

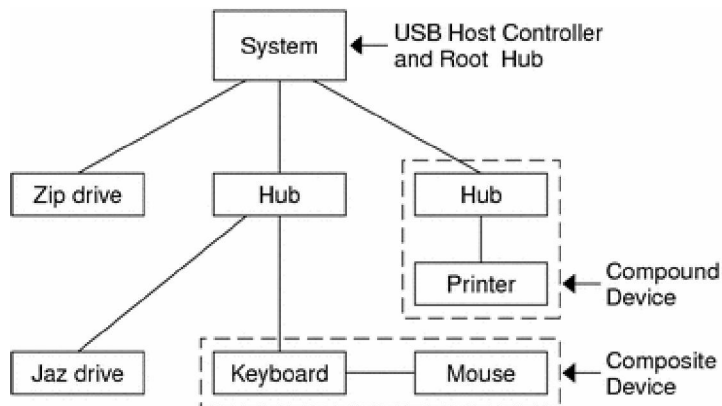


Figura 2.21 Esquema de funcionamiento y control del HOST

Cada función contiene la información sobre la configuración que describe su capacidad y requisitos en cuestión de recursos. Antes de que una función pueda ser usado debe ser configurado por el host [41].

A algunos ejemplos de funciones pueden ser los siguientes:

- Una interfaz humana (HID) como ratón, teclado, tablas digitalizadoras o controladores de juegos.
- Dispositivos de imágenes: cámaras, escáneres o impresoras
- Dispositivos de almacenamiento: CD-ROMs, DVD y disqueteras...

### ***La capa lógica***

El punto de vista lógico presenta capas y abstracciones que son relevantes para los distintos diseñadores e implementadores. La arquitectura lógica describe como unir el hardware del dispositivo USB a un driver del dispositivo en el host para que tenga el comportamiento que el usuario final desea.

La vista lógica de esta conexión es la mostrada en el esquema siguiente. En el podemos ver como el host proporciona conexión al dispositivo, donde esta conexión es a través de un simple enlace USB. La mayoría de los demás buses como PCI, ISA, etc proporcionan múltiples conexiones al dispositivos y los drivers lo manipulan mediante algunas combinaciones de estas conexiones (IO y direcciones de memoria, interrupciones y canales DMA).

Físicamente el USB tiene sólo un cable simple de bus que es compartido por todos los dispositivos del bus. Sin embargo, desde el punto de vista lógico cada dispositivo tiene su propia conexión punto a punto al host.

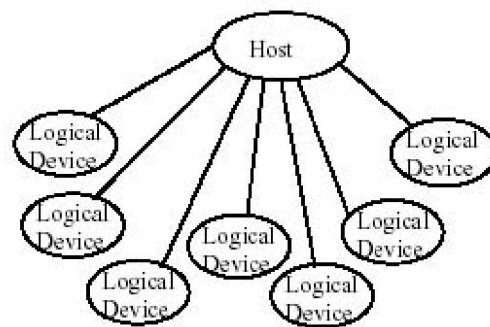


Figura 2.22 Lógica del HOST

Si lo miramos desde el punto de vista lógico, los hubs son también dispositivos pero no se muestran en la figura para la simplificación del esquema.

Aunque la mayoría de las actividades de los hosts o de los dispositivos lógicos usan esta perspectiva lógica, el host mantiene el conocimiento de la

topología física para dar soporte al proceso de desconexión de los hubs. Cuando se quita un hub, todos los dispositivos conectados a él son quitados también de la vista lógica de la topología

### ***La relación "Software del cliente-función"***

A pesar de que la topología física y lógica del USB refleja la naturaleza de compartición del bus, la manipulación del interfaz de una función USB por parte del software del cliente (CSw) se presenta con una vista distinta.

El software del cliente para las funciones USB debe usar el interfaz de programación software USB para manipular sus funciones en contraposición de las que son manipuladas directamente a través de la memoria o los accesos IO como pasa con otros buses (PCI, EISA, PCMCIA, ...). Durante esta operación, el software del cliente debería ser independiente a otros dispositivos que puedan conectarse al USB [39].

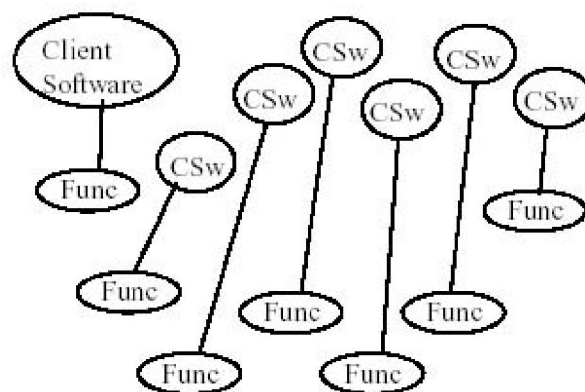


Figura 2.23 Diagrama de conexión cliente-función.

### ***b) El flujo de datos del bus USB***

Un dispositivo USB desde un punto de vista lógico hay que entenderlo como una serie de endpoints, a su vez los endpoints se agrupan en conjuntos que dan lugar a interfaces, las cuales permiten controlar la función del dispositivo.

Como ya se ha visto la comunicación entre el host y un dispositivo físico USB se puede dividir en tres niveles o capas. En el nivel más bajo el controlador de host USB se comunica con la interfaz del bus utilizando el cable USB, mientras que en un nivel superior el software USB del sistema se comunica con el dispositivo lógico utilizando la tubería de control por defecto (*"Default Control Pipe"*). En lo que al nivel de función se refiere, el software cliente establece la comunicación con las interfaces de la función a través de tuberías asociadas a endpoints.

### **Endpoints y direcciones de dispositivo**

Cada dispositivo USB está compuesto por una colección de endpoints independientes, y una dirección única asignada por el sistema en tiempo de conexión de forma dinámica. A su vez cada endpoint dispone de un identificador único dentro del dispositivo al que pertenece, a este identificador se le conoce como número de endpoint y viene asignado de fábrica. Cada endpoint tiene una determinada orientación de flujo de datos. La combinación de dirección, número de endpoint y orientación, permite referenciar cada endpoint de forma inequívoca.

Cada endpoint es por sí solo una conexión simple que soporta un flujo de datos en una única dirección, bien de entrada o bien de salida.

Cada endpoint se caracteriza por:

Frecuencia de acceso al bus requerida

Ancho de banda requerido

Número de endpoint

Tratamiento de errores requerido

Máximo tamaño de paquete que el endpoint puede enviar o recibir

Tipo de transferencia para el endpoint

La orientación en la que se transmiten los datos

Existen dos endpoints especiales que todos los dispositivos deben tener, los endpoints con número 0 de entrada y de salida, que deben de implementar un método de control por defecto al que se le asocia la tubería de control por defecto. Estos endpoints están siempre accesibles mientras que el resto no lo estarán hasta que no hayan sido configurados por el host [40].

## **Tuberías**

Una tubería USB es una asociación entre uno o dos endpoints en un dispositivo, y el software en el host. Las tuberías permiten mover datos entre software en el host, a través de un buffer, y un endpoint en un dispositivo. Hay dos tipos de tuberías:



**Stream:** Los datos se mueven a través de la tubería sin una estructura definida.

**Mensaje:** Los datos se mueven a través de la tubería utilizando una estructura USB.

Además una tubería se caracteriza por:

Demanda de acceso al bus y uso del ancho de banda

Un tipo de transferencia

Las características asociadas a los endpoints

Como ya se ha comentado, la tubería que está formada por dos endpoints con número cero se denomina tubería de control por defecto. Esta tubería está siempre disponible una vez se ha conectado el dispositivo y ha recibido un reseteo del bus. El resto de tuberías aparecen después que se configure el dispositivo. La tubería de control por defecto es utilizada por el software USB del sistema para obtener la identificación y requisitos de configuración del dispositivo, y para configurar al dispositivo.

El software cliente normalmente realiza peticiones para transmitir datos a una tubería vía RPs y entonces, o bien espera, o bien es notificado de que se ha completado la petición. El software cliente puede causar que una tubería devuelva todas las RPs pendientes. El cliente es notificado de que una RP se ha completado cuando todas las transacciones del bus que tiene asociadas se han completado correctamente, o bien porque se han producido errores.

Una RP puede necesitar de varias tandas para mover los datos del cliente al bus. La cantidad de datos en cada tanda será el tamaño máximo de un paquete excepto el último paquete de datos que contendrá los datos que faltan. De modo que un paquete recibido por el cliente que no consiga llenar el buffer de datos de la RP puede interpretarse de diferentes modos en función de las expectativas del cliente, si esperaba recibir una cantidad variable de datos considerará que se trata del último paquete de datos, sirviendo pues como delimitador de final de datos, mientras que si esperaba una cantidad específica de datos lo tomará como un error.

**Streams:** No necesita que los datos se transmitan con una cierta estructura. Las tuberías stream son siempre unidireccionales y los datos se transmiten de forma secuencial: *"first in, first out"*. Están pensadas para interactuar con un único cliente, por lo que no se mantiene ninguna política de sincronización entre múltiples clientes en caso de que así sea. Un stream siempre se asocia a un único endpoint en una determinada orientación.

**Mensajes:** A diferencia de lo que ocurre con los streams, en los mensajes la interacción de la tubería con el endpoint consta de tres fases. Primero se realiza una petición desde el host al dispositivo, después se transmiten los datos en la dirección apropiada, finalmente un tiempo después se pasa a la fase estado. Para poder llevar a cabo este paradigma es necesario que los datos se transmitan siguiendo una determinada estructura [39].

Las tuberías de mensajes permiten la comunicación en ambos sentidos, de hecho la tubería de control por defecto es una tubería de mensajes. El software

USB del sistema se encarga de que múltiples peticiones no se envíen a la tubería de mensajes concurrentemente. Un dispositivo ha de dar únicamente servicio a una petición de mensaje en cada instante por cada tubería de mensajes. Una tubería de mensajes se asocia a un par de endpoints de orientaciones opuestas con el mismo número de endpoint.

### **Frames y microframes**

USB establece una unidad de tiempo base equivalente a 1 milisegundo denominada frame y aplicable a buses de velocidad media o baja, en alta velocidad se trabaja con microframes, que equivalen a 125 microsegundos. Los (micro)frames no son más que un mecanismo del bus USB para controlar el acceso a este, en función del tipo de transferencia que se realice. En un (micro) frame se pueden realizar diversas transacciones de datos [40].

### **Tipos de transferencias**

La interpretación de los datos que se transmitan a través de las tuberías, independientemente de que se haga siguiendo o no una estructura USB definida, corre a cargo del dispositivo y del software cliente. No obstante, USB proporciona cuatro tipos de transferencia de datos sobre las tuberías para optimizar la utilización del bus en función del tipo de servicio que ofrece la función.

Estos cuatro tipos son:

Transferencias de control

Transferencias isócronas

Transferencias de interrupción

Transferencias de bulk ("*Bulk*")

### ***Transferencias de control***

Es el único tipo de transferencia que utiliza tuberías de mensajes, soporta por lo tanto comunicaciones de tipo configuración/comando/estado entre el software cliente y su función. Una transferencia de tipo control se compone de una transacción de setup del host a la función, cero o mas transacciones de datos en la dirección indicada en la fase de setup, y por último una transacción de estado de la función al host. La transacción de estado devolverá éxito cuando el endpoint haya completado satisfactoriamente la operación que se había solicitado.

Por lo tanto este tipo de transferencia esta pensado para configurar, obtener información, y en general manipular el estado de los dispositivos. El tamaño máximo de datos que se transmiten por el bus viene determinado por el endpoint. En dispositivos de velocidad media los posibles tamaños máximos son de 8, 16, 32 o 64 bytes, en velocidad baja el tamaño es de 8 bytes y en velocidad alta 64 bytes. El porcentaje de (micro)frame utilizado ronda el 30% en velocidad baja, 5% en velocidad media y el 2% en alta.

El endpoint puede estar ocupado durante la fase de envío de datos y la fase de estado, en esos casos el endpoint indica al host que se encuentra ocupado, invitando al host a intentarlo mas tarde. Si el endpoint recibe un mensaje de setup y se encontraba en mitad de una transferencia de control, aborta la transferencia actual y pasa a la nueva que acaba de recibir. Normalmente el host no inicia una nueva transferencia de control con un endpoint hasta que no ha acabado la actual,

si bien debido a problemas de transmisión el host puede considerar que se han producido errores y pasar a la siguiente. USB proporciona detección y recuperación, vía retransmisión, de errores en las transferencias de control.

### ***Transferencias isócronas***

Hacen uso de tuberías stream. Garantiza un acceso al bus USB con una latencia limitada, asegura una transmisión constante de los datos a través de la tubería siempre y cuando se suministren datos, además en caso de que la entrega falle debido a errores no se intenta reenviar los datos [42].

USB limita el máximo tamaño de datos para los endpoints con tipo de transferencia isócrona a 1023 bytes para los endpoints de velocidad media y 1024 bytes para velocidad alta. De hecho las transferencias isócronas solo se pueden usar en dispositivos de velocidad alta o media. En función de la cantidad de datos que se estén transmitiendo en un momento dado, en velocidad media el porcentaje de frame utilizado puede variar desde un 1% hasta un 69%, mientras que el porcentaje de micro frame utilizado en velocidad alta varía entre un 1% y un 41%.

### ***Transferencias de interrupción***

Utiliza tuberías stream. Este tipo de transferencia está diseñado para servicios que envían o reciben datos de forma infrecuente. Esta transferencia garantiza el máximo servicio para la tubería durante el periodo en el que envía. En caso de error al enviar los datos se reenvían en el próximo periodo de envío de datos.

El tamaño de paquete de datos máximo es de 1024 bytes para alta velocidad, 64 bytes para velocidad media y 8 bytes para baja velocidad. En ningún caso se precisa que los paquetes sean de tamaño máximo, es decir, no es necesario rellenar los paquetes que no alcancen el máximo. Cuando en una transferencia de interrupción se necesite transmitir más datos de los que permite el paquete máximo, todos los paquetes a excepción del último paquete deben de tener el tamaño máximo. De modo que la transmisión de un paquete se ha llevado a cabo cuando se ha recibido la cantidad exacta esperada o bien, se ha recibido un paquete que no alcanza el tamaño máximo. El porcentaje de (micro) frame utilizado ronda el 13% en velocidad baja y el 2.5% en velocidad media, mientras que en velocidad alta para cantidades similares utilizadas para obtener los anteriores porcentajes se obtienen resultados del 1%, pero para cantidades muy superiores se puede llegar a una utilización del 42% [42].

### ***Transferencias de bulk ("Bulk")***

Hace uso de tuberías stream. Está diseñado para dispositivos que necesitan transmitir grandes cantidades de datos en un momento determinado sin importar mucho el ancho de banda disponible en ese momento. Esta transferencia garantiza el acceso al USB con el ancho de banda disponible, además en caso de error se garantiza el reenvío de los datos. Por lo tanto este tipo de transferencia garantiza la entrega de los datos pero no un determinado ancho de banda o latencia.

El tamaño máximo de paquete de datos para velocidad media es de 8, 16, 32 o 64 bytes, mientras que en velocidad alta es de 512 bytes. Los dispositivos de velocidad baja no disponen de endpoints con este tipo de transferencia. No es necesario que los paquetes se rellenen para alcanzar el tamaño máximo. El porcentaje de frame utilizado en velocidad media en función del número de bytes enviados varía entre el 1% y el 5%, mientras que el porcentaje de microframe en velocidad alta varía entre un 1% y un 5%, eso sí, teniendo en cuenta mayor cantidad de datos [40].

### ***c) Capa de protocolo***

La forma en la que las secuencias de bits se transmiten en USB es la siguiente; primero se transmite el bit menos significativo, después el siguiente menos significativo y así hasta llegar al bit más significativo. Cuando se transmite una secuencia de bytes se realiza en formato *"little-endian"*, es decir del byte menos significativo al byte más significativo.

En la transmisión se envían y reciben paquetes de datos, cada paquete de datos viene precedido por un campo Sync y acaba con el delimitador EOP, todo esto se envía codificado además de los bits de relleno insertados. En este punto cuando se habla de datos se refiere a los paquetes sin el campo Sync ni el delimitador EOP, y sin codificación ni bits de relleno [42].

El primer campo de todo paquete de datos es el campo PID. El PID indica el tipo de paquete y por lo tanto, el formato del paquete y el tipo de detección de errores aplicado al paquete. En función de su PID podemos agrupar los diferentes tipos de paquetes en cuatro clases:

<b>Tipo PID</b>	<b>Nombre PID</b>	<b>PID</b>	<b>Descripción</b>
Token	OUT	0001B	Dirección + número de endpoint en una transacción host a función.
	IN	1001B	Dirección + número de endpoint en una transacción función a host.
	SOF	0101B	Indicador de inicio de frame (Start Of Frame) y número de frame.
	SETUP	1101B	Dirección + número de endpoint en una transacción host a función para realizar un Setup de una tubería de control.
Data	DATA0	0011B	PID de paquete de datos par.
	DATA1	1011B	PID de paquete de datos impar.
	DATA2	0111B	PID de paquete de datos de alta velocidad, elevado ancho de banda en una transferencia isócrona en un microframe.
	MDATA	1111B	PID de paquete de datos de alta velocidad para split y elevado ancho de banda en una transferencia isócrona.



Handshake	ACK	0010B	El receptor acepta el paquete de datos libre de errores.
	NAK	1010B	El dispositivo receptor no puede aceptar los datos o el dispositivo emisor no puede enviar los datos.
	STALL	1110B	Endpoint sin servicio o una petición de control sobre una tubería no está soportado.
	NYET	0110B	Aún no se ha recibido una respuesta del receptor.
Special	PRE	1100B	(Token) Habilita tráfico de bajada por el bus a dispositivos de velocidad baja.
	ERR	1100B	(Handshake) Error de transferencia Split.
	SPLIT	1000B	(Token) Transferencia de alta velocidad Split.
	PING	0100B	(Token) Control de flujo sobre endpoints de tipo control bulk.
	Reservado	0000B	PID reservado.

A continuación el formato de los campos y los paquetes, además de una vista general de cómo funciona el protocolo:

El formato de los campos

El formato de los paquetes

Transacciones

Split

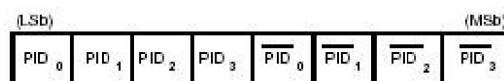
El protocolo y las transferencias

### ***Formato de los campos***

Los paquetes se dividen en campos, a continuación el formato de los diferentes campos.

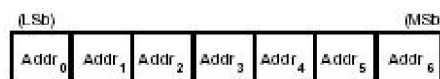
#### **Campo identificador de paquete (PID)**

Es el primer campo que aparece en todo paquete. El PID indica el tipo de paquete, y por tanto el formato del paquete y el tipo de detección de error aplicado a este. Se utilizan cuatro bits para la codificación del PID, sin embargo el campo PID son ocho bits, que son los cuatro del PID seguidos del complemento a 1 de esos cuatro bits. Estos últimos cuatro bits sirven de confirmación del PID. Si se recibe un paquete en el que los cuatro últimos bits no son el complemento a 1 del PID, o el PID es desconocido, se considera que el paquete está corrupto y es ignorado por el receptor.



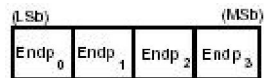
#### **Campo dirección**

Este campo indica la función, a través de la dirección, que envía o es receptora del paquete de datos. Se utilizan siete bits, de lo cual se deduce que hay un máximo de 128 direcciones.



### **Campo endpoint**

Se compone de cuatro bits e indica el número de "endpoint" al que se quiere acceder dentro de una función, como es lógico este campo siempre sigue al campo de dirección.

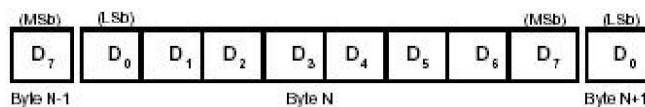


### **Campo número de frame**

Es un campo de 11 bits que es incrementado por el host cada (micro)frame en una unidad. El máximo valor que puede alcanzar es el 7FFH, si se vuelve a incrementar pasa a cero.

### **Campo de datos**

Los campos de datos pueden variar de 0 a 1024 bytes. En el dibujo se muestra el formato para múltiples bytes.



### ***"Cyclic Redundancy Checks (CRC)"***

El CRC se usa para proteger todos los campos no PID de los paquetes de tipo token y de datos. El CRC siempre es el último campo y se genera a partir del resto de campos del paquete, a excepción del campo PID. El receptor al recibir el paquete comprueba si se ha generado de acuerdo a los campos del paquete, si no

es así, se considera que alguno o mas de un campo están corruptos, en ese caso se ignora el paquete.

El CRC utilizado detecta todos los errores de un bit o de dos bits. El campo de CRC es de cinco bits para los paquetes de tipo N, SETUP, OUT, PING y SPLIT. El polinomio generador es el siguiente:

$$G(X) = X^5 + X^2 + 1$$

En el caso de los paquetes de datos se utiliza un campo CRC de 16 bits, el polinomio generador es el siguiente:

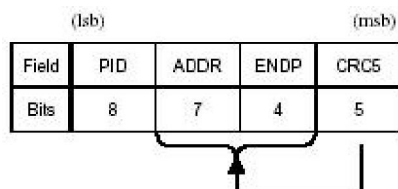
$$G(X) = X^{16} + X^{15} + X^2 + 1$$

### ***Formato de los paquetes***

A continuación los diferentes paquetes y sus formatos en función de los campos que los forman.

### **Paquetes de tipo token**

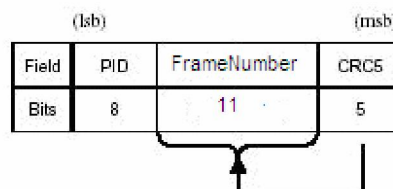
Un token está compuesto por un PID que indica si es de tipo N, OUT o SETUP. El paquete especial de tipo PING también tiene la misma estructura que token. Después del campo PID viene seguido de un campo dirección y un campo endpoint, por último hay un campo CRC de 5 bits.



En los paquetes OUT y SETUP esos campos identifican al endpoint que va a recibir el paquete de datos que va a continuación. En los paquetes IN indican el endpoint que debe transmitir un paquete de datos. En el caso de los paquetes PNG hacen referencia al endpoint que debe responder con un paquete "handshake".

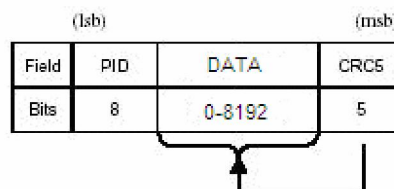
### Paquete inicio de frame (SOF)

Estos paquetes son generados por el host cada un milisegundo en buses de velocidad media y cada 125 microsegundos para velocidad alta. Este paquete está compuesto por un campo número de frame y un campo de CRC de 5 bits.



### Paquete de datos

Este paquete está compuesto por cero o más bytes de datos seguido de un campo de CRC de 16 bits. Existen cuatro tipos de paquetes de datos: DATA0, DATA1, DATA2 y MDATA. El número máximo de bytes de datos en velocidad baja es de ocho bytes, en media de 1023 bytes y en alta de 1024 bytes. El número de bytes de datos ha de ser entero.



## Paquetes "*Handshake*"

Los paquetes "*handshake*", en castellano apretón de manos, se utilizan para saber el estado de una transferencia de datos, indicar la correcta recepción de datos, aceptar o rechazar comandos, control de flujo, y condiciones de parada. El único campo que contiene un paquete de este tipo es el campo PID.

	(lsb)	(msb)
Field	PID	
Bits	8	

Existen cuatro paquetes de tipo "*handshake*" además de uno especial:

**ACK:** Indica que el paquete de datos ha sido recibido y decodificado correctamente. ACK sólo es devuelto por el host en las transferencias IN y por una función en las transferencias OUT, SETUP o PNG.

**NAK:** Indica que una función no puede aceptar datos del host (OUT) o que no puede transmitir datos al host (IN). También puede enviar una función durante algunas fases de transferencias IN, OUT o PNG. Por último se puede utilizar en el control de flujo indicando disponibilidad. EL host nunca puede enviar este paquete.

**STALL:** Puede ser devuelto por una función en transacciones que intervienen paquetes de tipo IN, OUT o PNG. Indica que una función es incapaz de transmitir o enviar datos, o que una petición a una tubería control no está soportada. El host no puede enviar bajo ninguna condición paquetes STALL.

**NYET:** Sólo disponible en alta velocidad es devuelto como respuesta bajo dos circunstancias. Como parte del protocolo PNG, o como respuesta de un hub a una transacción SPLIT indicando que la transacción de velocidad media o baja

aún no ha terminado, o que el hub no está aún habilitado para realizar la transacción.

**ERR:** De nuevo sólo disponible en alta velocidad y de nuevo formando parte del protocolo PING, permite a un hub de alta velocidad indicar que se ha producido un error en un bus de media o baja velocidad.

### ***Transacciones***

Los paquetes de tipo token tiene como objetivo indicar el inicio de una transacción de datos de una determinada forma.

#### **Transacción IN**

La transacción empieza con el envío de un paquete de tipo IN por parte del host a un determinado endpoint en una función. Un endpoint cuando recibe un paquete de tipo IN debe comportarse como muestra la siguiente tabla.

**Tabla 2.1 Transacción de paquetes tipo IN**

<b>Token recibido corrupto</b>	<b>Estado de endpoint</b>	<b>La función puede transmitir los datos</b>	<b>Acción</b>
Si	--	--	No responde
No	Deshabilitado	--	EnvíaSTALL
No	Habilitado	No	EnvíaNAK
No	Habilitado	Si	Envía el paquete de datos

La respuesta que del host al recibir un paquete de datos en la transacción se puede observar en la siguiente tabla:

**Tabla 2.2 Respuesta del Host**

<b>Paquete de datos corrupto</b>	<b>El host puede aceptar los datos</b>	<b>Respuesta del host</b>
Si	--	Descarta el paquete, no responde
No	No	Descarta el paquete, no responde
No	Si	Envía ACK

### **Sincronización mediante conmutación de bits**

USB ofrece un mecanismo que garantiza la sincronización entre el emisor y el receptor de datos a lo largo de múltiples transacciones. La idea es garantizar que los dos sean conscientes de que los handshakes han sido recibidos correctamente. Para ello se utilizan los dos PID DATA0 y DATA1 que sólo varían en un bit. De manera que inicialmente tanto el emisor y el receptor están sincronizados en la secuencia de bits. De modo que, el que recibe datos conmuta su bit cuando puede aceptar datos y ha recibido un paquete de datos libre de errores. Por su parte el que envía conmuta su bit cuando recibe un ACK, tal que si el último paquete de datos tenía PID DATA0 el siguiente tendrá DATA1, y viceversa, además si todo ha salido bien, el PID del paquete coincidirá con la secuencia de bits del receptor.

### **Transacción OUT**

El host envía un paquete OUT a un determinado endpoint y seguidamente envía el paquete de datos. Suponiendo que el endpoint ha decodificado correctamente el token, en caso contrario se ignora el token y los datos, espera a



recibir un paquete de datos. El comportamiento del endpoint cuando recibe el paquete de datos es el siguiente.

**Tabla 2.3 Comportamiento del endpoint**

Paquete de datos corrupto	Estado del receptor	Coincidencia de la secuencia de bits	La función puede aceptar los datos	Acción
Si	--	--	--	No responde
No	Deshabilitado	--	--	Envía STALL
No	Habilitado	No	--	Envía ACK
No	Habilitado	Si	Si	Envía ACK
No	Habilitado	Si	No	Envía NAK

### **Transacción SETUP**

La transacción SETUP es una transacción especial que tiene las mismas fases que una transacción OUT, que sólo se puede utilizar con endpoints de tipo control y cuya finalidad es la de indicar el inicio de la fase setup.

### ***Split***

El token especial SPLIT es utilizado para poder soportar transacciones en varias partes entre un hub que trabaja a velocidad alta con dispositivos de velocidad media o baja. Se definen nuevas transacciones que utilizan el token SPLIT, en concreto dos transacciones con sus respectivos paquetes: *"start-split transaction (SSPLIT)"* y *"complete-split transaction (CSPLIT)"*.

### **Transacciones Split**

La transacción split es utilizada solamente por el host y un hub cuando se quiere comunicar con un dispositivo de velocidad media o baja conectado con el hub. El host puede iniciar una transacción por partes a través del paquete SSPLIT y puede completarla, recibiendo las respuestas de la función de velocidad baja o media, a través del paquete CSPLIT. Esto permite al host realizar otras transacciones a velocidad alta sin tener que esperar a que acabe la transacción.

A continuación un ejemplo de una transacción por partes para establecer una transacción de tipo IN.

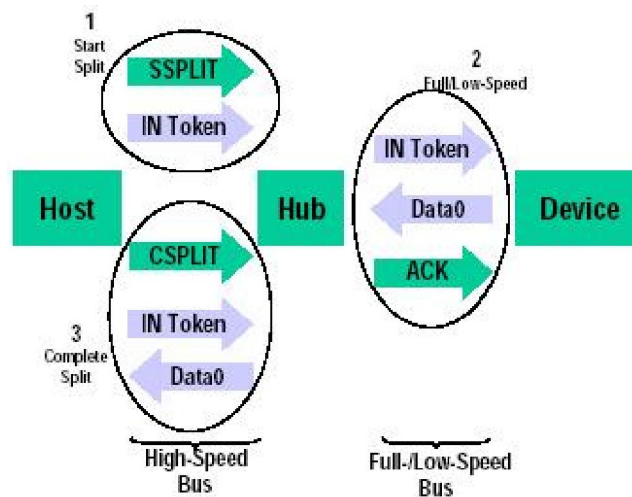


Figura 2.24 Transacción SPLIT

### Formato del paquete SPLIT

Campos del paquete SSPLIT y CSPLIT:

	(lsb)							(msb)
Field	SPLIT PID	Hub Addr	SC	Port	S	E	ET	CRC5
Bits	8	7	1	7	1	1	2	5

El diagrama muestra una flecha que indica que los campos SPLIT PID, Hub Addr, SC, Port, S, E y ET están agrupados y forman parte de un campo más grande que incluye CRC5.

El campo "*Hub Addr*" es un campo de dirección que indica la dirección del hub. El bit SC indica el tipo de paquete, si SC=0 se trata de un SSPLIT, y si SC=1 de un CSPLIT. El campo "puerto del hub destino" hace referencia al puerto del hub al que está destinado esta transacción por partes, el formato de este campo es idéntico al de un campo de dirección. El significado de los bits S y E dependen del tipo de transferencia de datos que se esté utilizando, y puede hacer referencia a como se relaciona los tamaños de los paquetes de datos de alta y media velocidad, o hacer referencia a la velocidad de la transacción. El bit E en los paquetes CSPLIT nunca se utiliza, de hecho se llama bit U. Por último el campo de dos bits ET indica el tipo de transferencia de datos, es decir, el tipo de endpoint. En la siguiente tabla se puede observar el funcionamiento de ET [42].

**Tabla 2.5 Tipos de endpoint**

<b>ET</b>	<b>Tipo de endpoint</b>
00	Control
01	Isócrono
10	<i>"Bulk"</i>
11	Interrupción

### ***El protocolo y las transferencias***

En este apartado se muestra de forma general las transacciones que se realizan, y las particularidades de cada tipo de transferencia. De forma general toda función en principio está esperando a que el host le envíe un paquete, si este paquete es de tipo token entonces cambia el estado de la

función iniciándose una transacción. También se debe tener en cuenta, que cuando o bien el host, o bien una función se encuentran en un estado que no es el de reposo, aparecen los timeouts como otra causa de error[11].

### Transferencias de bulto ("*Bulk*")

En las transferencias de bulto se puede hablar en parte de transacciones de bulto, pues la idea es enviar datos de paquete en paquete sin que tenga que haber un flujo de datos continuo. La diferencia reside en que si hablamos de transferencia, no se puede considerar que haya terminado hasta que el host haya conseguido enviar los datos, o bien que después de varios intentos fallidos de un error.

Cada transacción de bulto se puede dividir en tres fases: token, datos y "*handshake*", si bien gracias al token PING pueden haber transacciones de dos fases: token y "*handshake*". A continuación un esquema de una transacción de bultos.

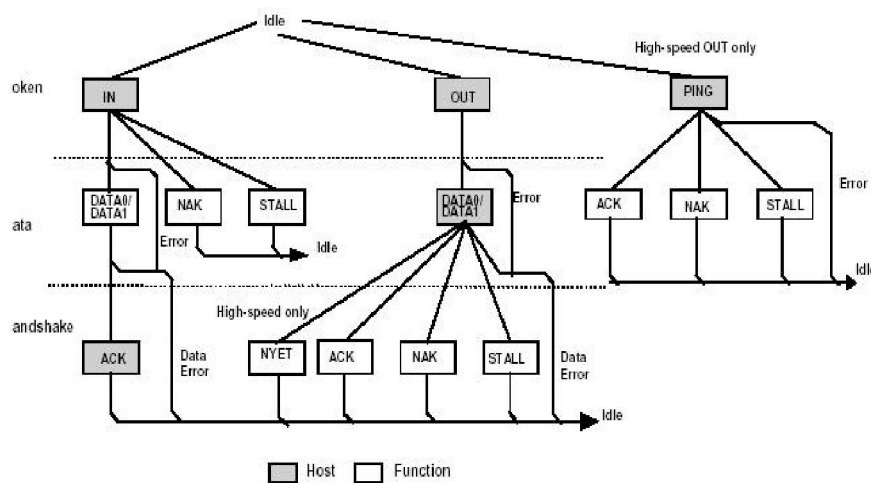


Figura 2.25 Transferencia en Bulk.

## Transferencias de control

Estas transferencias constan de tres fases: transacción setup, fase de datos y transacción de estado. La transacción siempre la inicia el host, y sirve para enviar información de control para indicar al endpoint que se quiere realizar. El siguiente esquema representa una transacción setup [43].

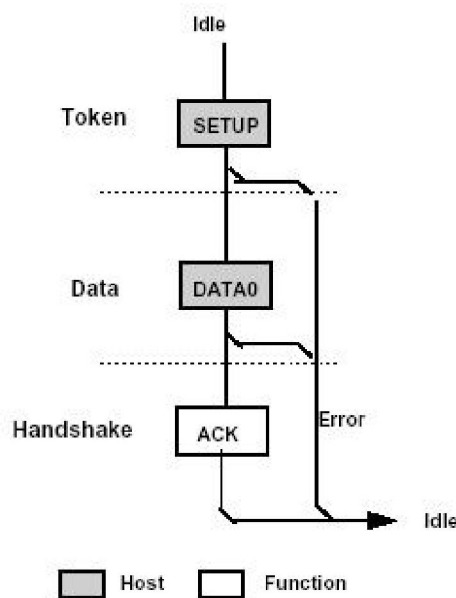


Figura 2.26 Transferencia de control.

A continuación se inicia la fase de transferencia de datos, que consiste en una secuencia de transacciones de datos siempre en la misma dirección alternando DATA0 y DATA1. Esta fase no es obligatoria, la dirección se establece en la fase setup. Finalmente tiene lugar una transacción estado, esta transacción es idéntica a una transacción de bultos. La dirección de esta transacción es siempre la contraria a la de la fase de transferencia de datos, y si esta no existiese, la dirección es del endpoint al host [11].

## Transferencias de interrupción

Las transferencias de interrupción son solamente transacciones de tipo IN y OUT. Desde el punto de vista de las transacciones es muy similar a una transferencia de búlbos. A continuación el esquema de una transacción de interrupción[43].

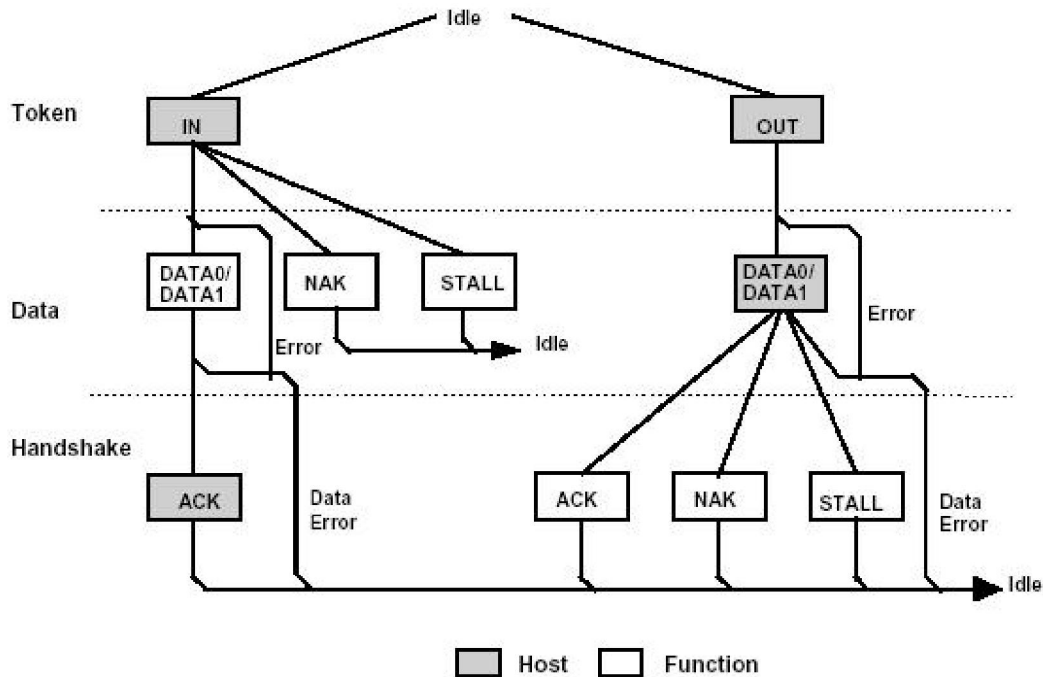


Figura 2.27 Transferencia de interrupciones.

## Transferencias isócronas

Una transferencia isócrona se plantea como una secuencia de transacciones muy sencillas para enviar o recibir datos. Estas transacciones no utilizan *"handshakes"* y por lo tanto no se reenvían paquetes, ya que el objetivo de la transferencia es simular un flujo constante de datos. A continuación un esquema de una transacción [11].

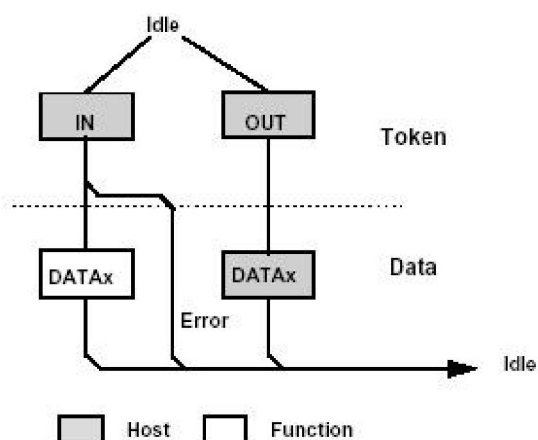


Figura 2.28 Transferencia isócrona.

#### *d) La eléctrica*

##### **Identificación de la velocidad del dispositivo**

Para poder iniciar cualquier tipo de transacción cuando se conecta el dispositivo al host, es necesario que este conozca la velocidad a la que trabaja. Con esa finalidad existe un mecanismo a nivel eléctrico. La diferencia entre los dispositivos de velocidad media y los de velocidad baja, es que en velocidad media tiene una resistencia conectada al D+, en velocidad baja la misma resistencia se encuentra en D- y no en D+ como se puede observar en los dibujos.

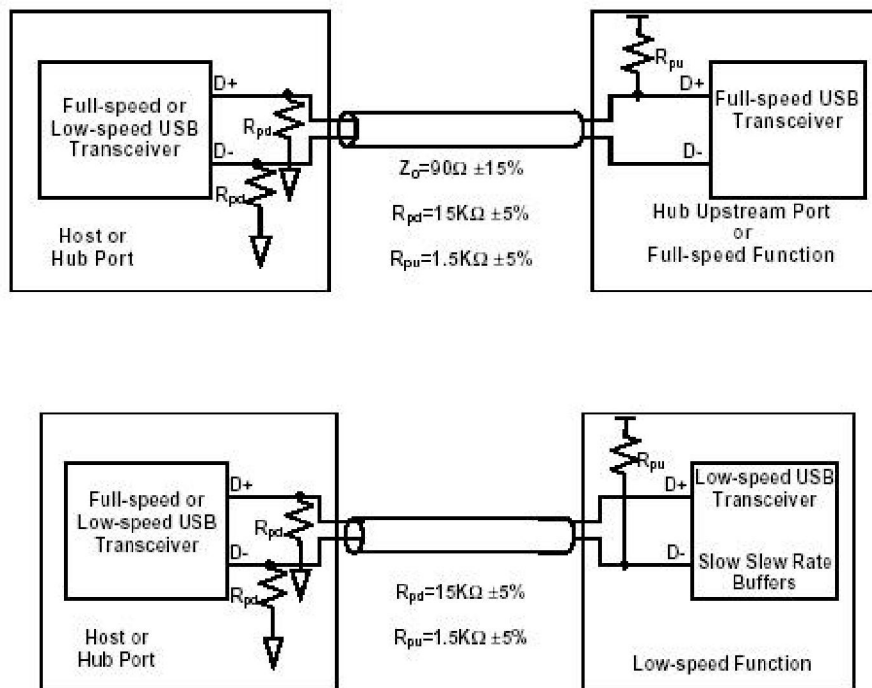


Figura 2.29 Esquema eléctrico de conexión por velocidad de transmisión.

De forma que después del reset el estado de reposo de la línea es diferente si se trata de baja o media velocidad. En el caso de dispositivos de alta velocidad lo que se hace es que en un principio se conecta como un dispositivo de velocidad media y más tarde a través de un protocolo se pasa a velocidad alta.

### Codificación de datos

El USB utiliza la codificación NRZI para la transmisión de paquetes. En esta codificación los "0" se representan con un cambio en el nivel, y por el contrario los "1" se representan con un no cambio en el nivel. De modo que las cadenas de cero producen transiciones consecutivas en la señal, mientras que



cadenas de unos produce largos periodos sin cambios en la señal. A continuación un ejemplo:

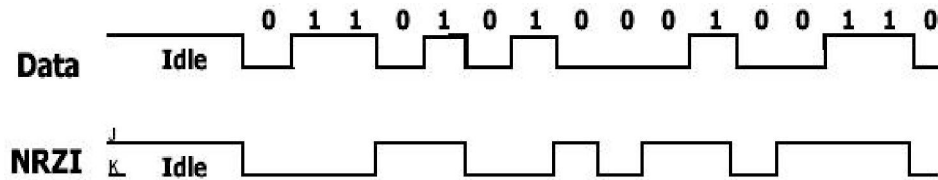


Figura 2.30 Codificación cero no invertido

### Relleno de bits

Debido a que cadenas de unos pueden producir largos periodos en los que la señal no cambia dando lugar a problemas de sincronización, se introducen los bits de relleno. Cada 6 bits consecutivos a "1" se inserta un bit a "0" para forzar un cambio, de esta forma el receptor puede volverse a sincronizar. El relleno bits empieza con el patrón de señal Sync. El "1" que finaliza el patrón de señal Sync es el primer uno en la posible primera secuencia de seis unos.

En las señales a velocidad media o baja, el relleno de bits se utiliza a lo largo de todo el paquete sin excepción. De modo que un paquete con siete unos consecutivos será considerado un error y por lo tanto ignorado.

En el caso de la velocidad alta se aplica el relleno de bits a lo largo del paquete, con la excepción de los bits intencionados de error usados en EOP a velocidad alta.

## Sync

Teniendo en cuenta que K y J representan respectivamente nivel bajo y nivel alto, el patrón de señal Sync en itido, con los datos codificados, es de 3 pares KJ seguidos de 2 K para el caso de velocidad media y baja. Para velocidad alta es una secuencia de 15 pares KJ seguidos de 2 K. A continuación el caso de velocidad media y baja:

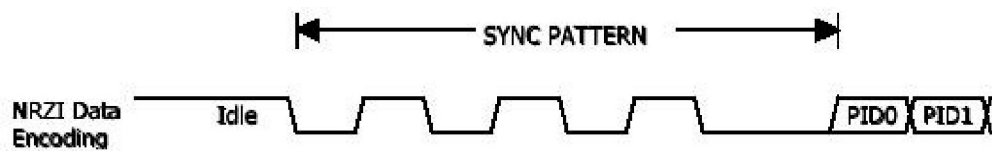


Figura 2.31 Sincronismo NRZI.

El patrón de señal Sync siempre precede al envío de cualquier paquete, teniendo como objetivo que el emisor y el receptor se sincronicen y se preparen para emitir y recibir datos respectivamente.

Si partimos de que el estado de reposo de la señal es J, podemos interpretar Sync como una secuencia impar de "0's" y un "1" que se inserta antes de los datos.

## EOP ( "End Of Packet")

A todo paquete le sigue EOP, cuya finalidad es indicar el final del paquete

En el caso de velocidad media y baja el EOP consiste en que, después del último bit de datos en el cual la señal estará o bien en estado J, o bien en estado K, se pasa al estado SE0 durante el periodo que se corresponde con el ocupado por

dos bits, finalmente se transita al estado J que se mantiene durante 1 bit. Esta última transición indica el final del paquete.

En el caso de la velocidad alta se utilizan bits de relleno erróneos, que no están en el lugar correcto, para indicar el EOP. Concretamente, el EOP sin aplicar codificación consistiría en añadir al final de los datos la secuencia 0111 1111.

#### ***e) La mecánica***

Como ya se ha visto la topología física USB consiste en la conexión de el puerto de bajada de un hub o host, con el puerto de subida de algún otro dispositivo o hub. Para facilitar la conexión de dispositivos de cara al usuario, USB utiliza dos tipos de conectores totalmente diferentes, los conectores de Serie A y los conectores de serie B. Los conectores de serie A permiten la conexión directa de dispositivos USB con el host o con el puerto de bajada de un host, y es obligatorio que estén presentes en todos los dispositivos y hubs USB. Los conectores de Serie B no son obligatorios y sirven para conectar un cable USB con el puerto de subida de un dispositivo, permitiendo por parte de los fabricantes de dispositivos la utilización de cables estándar USB.

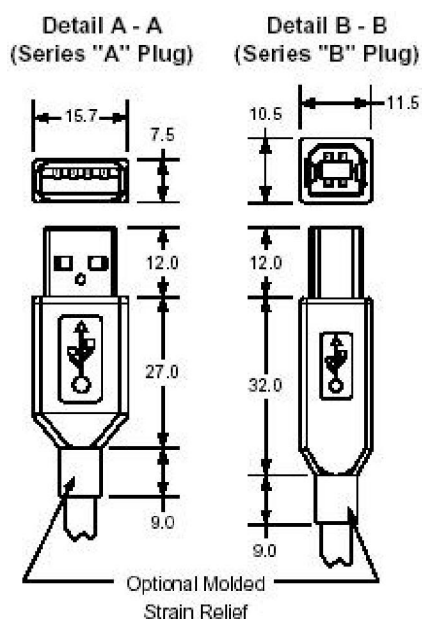


Figura 2.32 Detalle de conectores tipo A y B.

El cable USB consiste de cuatro conductores, dos conductores de potencia y dos de señal, D+ y D-. Los cables de media y alta velocidad están compuestos por un par trenzado de señal, además de GND (Tierra) y Vbus.

Existen tres tipos de cables USB: cable estándar de quita y pon, cable fijo de media y alta velocidad, y cable fijo de baja velocidad.

### **Cable estándar de quita y pon**

Se trata de un cable de velocidad alta y media, con un conector macho de Serie A en un extremo y un conector macho de Serie B en el otro extremo. Esto permite a los fabricantes de dispositivos fabricarlos sin cable y al usuario le

facilita la sustitución del cable en caso de que se estropee. Es un requisito que los dispositivos que utilicen este cable sean de velocidad alta o media, el correcto funcionamiento del cable con dispositivos de velocidad baja no está garantizado porque podría superar la longitud máxima del cable de velocidad baja.

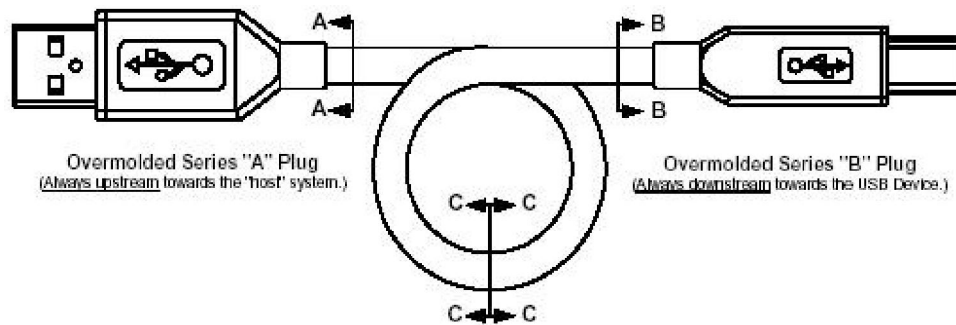


Figura 2.33 Cable USB A-B

### Cable fijo de velocidad alta y media

Con la denominación de fijo nos referimos a los cables que son proporcionados por el fabricante del dispositivo fijos a este, o bien sin ser fijos, con un conector específico del fabricante. Es obligatorio que en un extremo tenga un conector macho de Serie A. Dado que lo suministra el fabricante, puede ser utilizado por dispositivos tanto de velocidad alta y media, como de velocidad baja. En el caso de que se utilice para un dispositivo de velocidad baja, además de poder ser utilizado con dispositivos de velocidad media y alta, deberá cumplir con todos los requisitos propios de la velocidad baja.

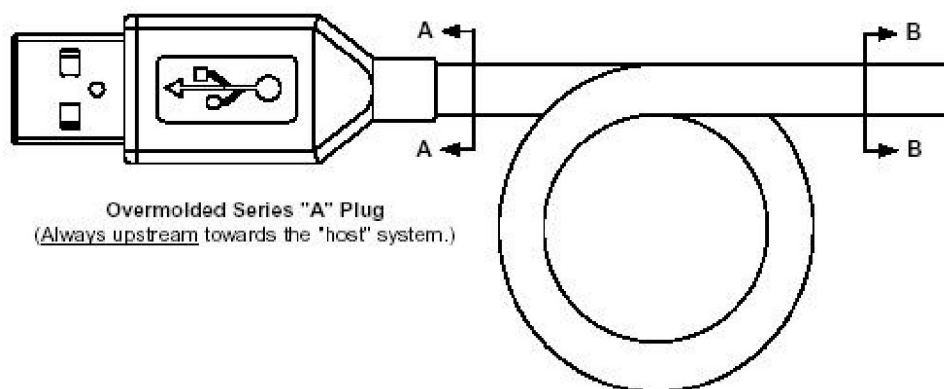


Figura 2.34 Cable USB fijo tipo A .

### **Cable fijo de velocidad baja**

A l igual que el cable fijo de alta y media velocidad tiene un conector macho de Serie A en un extremo ,m ientras que el otro depende del fabricante. La diferencia es que este tipo de cables só b funciona con dispositivos de velocidad baja.

**Tabla 2.6 Codificación del cable USB**

<b>Contacto</b>	<b>Señal</b>	<b>Cable</b>
1	VBUS	Rojo
2	Dats-	B lanco
3	Dats+	Verde
4	GND	Negro

### **2.4.2 Protocolo de Comunicaciones USART.**

RS232 es el nombre de la interfaz de comunicación seriem ás utilizado del mundo.

La norma serie está disponible en prácticamente el 99% de los ordenadores. Entre ellos el IBM PC compatible que habitualmente está equipado con dos puertos serie RS232, uno de ellos utilizado para conectar el ratón. El Apple Macintosh es una notable excepción, utilizando otra norma serie, la RS422[10].

La norma RS232 fue originalmente diseñada para conectar terminales de datos con dispositivos de comunicación (como módems y AITs). Desde un principio, fue también utilizada para conectar casi cualquier dispositivo imaginable. Los usos de la RS232 en el entorno doméstico son muchos y ampliamente conocidos. Desde la conexión del ratón, el módem/fax, agendas electrónicas de bolsillo, impresoras serie, grabadores de memoria (tipo EPROM, EEPROM), digitalizadores de vídeo, radios de AM/FM, etc. La lista sólo está limitada por la imaginación de los diseñadores[7].

En el entorno industrial el peso de la RS232 es también muy importante. Si bien existen soluciones de comunicación serie más robustas y versátiles, como la RS422 o la RS475, la RS232 sigue siendo por su sencillez, su diseño económico y, sobre todo, por su gran difusión, la norma más frecuente. Así, es fácil ver cómo robots industriales, manipuladores, controles de todo tipo, utilizan la RS232. Existen hasta cafeterías industriales (de las utilizadas en bares y restaurantes) que disponen de una RS232 para ser conectadas a un PC e informar de cuántos cafés han hecho en el transcurso del día, permitiendo al gerente de la empresa un control de caja, estadísticas de uso, etc.

La subnorma eléctrica de la RS232 es la V28. La norma fija una transmisión en modo común (cada circuito tienen una referencia a tierra y esta es común para todos los circuitos). Los circuitos son punto a punto, es decir, un *driver* con un sólo receptor de la señal.

La señal es bipolar con lógica invertida, utilizando los siguientes valores:

1 lógico = -3 a -15 voltios

0 lógico = +3 a +15 voltios

La ausencia de señal (0 voltios) queda diferenciado de 0 y 1 lógicos.

La RS232 es cortocircuitable. Esto quiere decir que, al menos teóricamente, los *drivers* de salida de las puertas disponen de un mecanismo de auto-protección contra Sobrecalentamientos. La tensión máxima de operación es +/-25 voltios y la carga máxima es de 3 Kohm. a 7 Kohm., con una corriente máxima de 500mA.

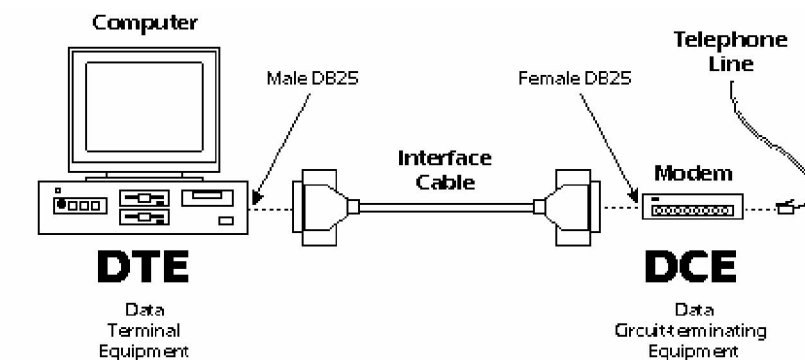
**Funcional:** (sobre norma asíncrona)

La norma asíncrona la forman nueve líneas.

La línea GND conecta la masa de ambos equipos y no merece mayor comentario. Las restantes ocho líneas pueden ser agrupados en tres bloques funcionales que se explican fácilmente si recordamos que la norma fue diseñada para conectar un PC (DTE típico) con un módem (DCE típico).



**Primer bloque:** Lo denominaremos "de establecimiento de conexión".



Está formado por las líneas:

Figura 2.35 Esquema de conexión RS232.

**DTR (Data Terminal Ready).** Terminal de datos preparado. (El PC y su RS232 están listos).

**DSR (Data Set Ready).** Equipo de comunicación preparado. (El módem está listo).

**RI (Ring Indicator).** Indicador de llamada. (El módem indica a su PC que ha recibido una llamada).

El objetivo es que ambos PCs sepan que se ha establecido un canal de comunicación (normalmente a través de la línea telefónica).

Las líneas DTR y DSR del equipo local y del remoto deben estar activas (set) durante todo el proceso. (De hecho cuando un PC desea dar por terminada una conexión basta con que, momentáneamente, desactive (reset) su DTR).

La conexión se inicia manualmente (el usuario llama con el teléfono al modem remoto) o automáticamente (el modem tiene capacidad de marcar un número de teléfono – dialling ) y se gestiona en los modems (que negocian, de forma automática, los parámetros de transferencia como la velocidad, compresión, etc). Se asume que el usuario del PC que llama activará el proceso que va a utilizar la conexión (un programa de transmisión de ficheros, por ejemplo). En el PC llamado se asume que el proceso homólogo está ya activo (porque, p.e., lo está permanentemente) o se puede activar automáticamente al recibir de su modem la señal de RI. Sea como fuera, la conexión queda establecida. A partir de este momento los PCs pueden intercambiar información.

### **Segundo bloque: "Control de flujo".**

Estas líneas tienen sentido en el caso de que el canal de comunicación establecido tenga una gestión half-duplex (ver a continuación “gestión simplex, half-duplex y full-duplex...”). Si el canal está establecido, el protocolo software de nivel de enlace de datos que se esté utilizando (Xmodem, Ymodem, HDLC, ...) fijará cuál de los dos DTEs debe comenzar a hablar/transmitir.

Las líneas en este bloque son usadas de la siguiente manera:

RTS (Request To Send). Petición de transmisión. El PC indica a su módem que quiere transmitir a la máquina remota.

CTS (Clear To Send). Canal libre para la transmisión. El módem indica a su PC que puede transmitir. Previamente habrá transmitido una señal portadora por el canal de comunicación para avisar al otro módem que ocupa el canal.

DCD (Data Carrier Detected). Detectada portadora. El módem indica a su PC que el canal de comunicación está ocupado por el equipo remoto.

El PC que quiere transmitir activa RTS, entonces su módem manda una señal portadora (sin modular, sin datos) para avisar al módem remoto que se reserva el canal. Una vez reservado el canal comunica a su DCE que ya puede transmitir activando la línea CTS.

Cuando un PC haya terminado de transmitir, desactivará RTS, el módem quitará la portadora y desactivará CTS. Entonces el otro módem podrá reservar el canal si su PC desea transmitir.

En caso de que la gestión del canal sea *full-duplex* todo es más sencillo. Cuando un PC quiere transmitir activa su RTS. Automáticamente su módem le da paso activando CTS.

**Tercer Bloque** : “Transmisión/recepción de datos”.

El funcionamiento de las líneas de este bloque es obvio. Cuando un PC puede transmitir, lo hace por la línea TxD. Transmisión de datos.

...y si está recibiendo datos lo hace por RxD. Recepción de datos.

La transmisión serial de los datos, tal y como se ha explicado, con el bit de START, de STOP, etcétera, se produce en estas líneas.

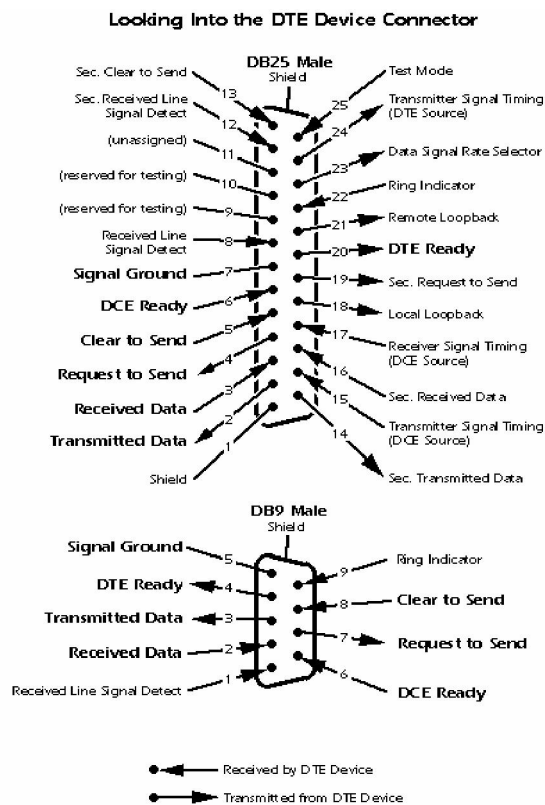


Figura 2.36 Conector DB 25 y 9 para DTE.

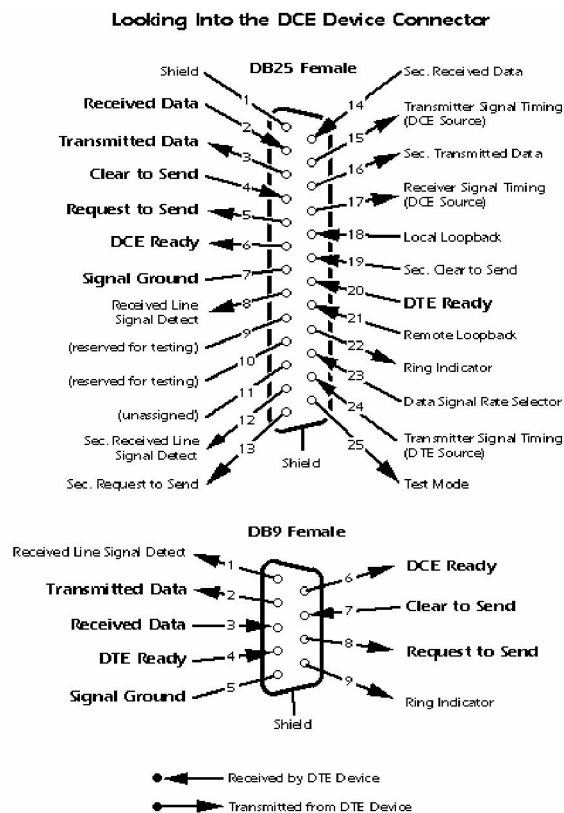


Figura 2.37 Conector DB 25 y 9 para DCE .

### 2.4.3 Protocolo de Comunicaciones I2C.

El bus I2C es una interfase serial de dos alambres desarrollada por la Corporación Philips® . La especificación original, o modo de estándar, fue para transferencia de datos hasta 100 Kbps. La especificación mejorada (modo rápido) está también implementada en los pic. La comunicación entre dispositivos se puede realizar en el modo estándar y en el modo rápido si los

dispositivos están unidos al mismo bus. El clock determinará la velocidad de los datos. La interfase I2C emplea un protocolo amplio para asegurar una transmisión y recepción de datos fiable [7].

### **Introducción de las especificaciones I2C**

Está orientado a las aplicaciones de 8-bit controladas por un microprocesador y éstas son básicamente los criterios que se deben establecer:

Un sistema consiste en al menos un microcontrolador y varios sistemas periféricos como memorias o circuitos diversos

El costo de conexión entre los varios dispositivos dentro del sistema debe de ser el mínimo.

El sistema que utiliza este Bus no requiere una alta tasa de transferencia de datos

La total eficacia del sistema depende de la correcta selección de la naturaleza de los dispositivos y de la interconexión de la estructura del bus.

### **El concepto del Bus I2C**

El bus I2C soporta cualquier tipo de componente (NMOS, CMOS, bipolar, etc.). Dos hilos físicos uno de datos (SDA) y otro de reloj (SCL) transportan la información entre los diversos dispositivos conectados al bus.

Cada dispositivo es reconocido por una única dirección (si es un microcontrolador, LCD, memoria o teclado) y puede operar cualquiera como

transmisor o receptor de datos, dependiendo de la función del dispositivo. Un display es solo un receptor de datos mientras que una memoria recibe y transmite datos.

En función de que envíe o reciba datos se debe considerar los dispositivos como Maestros (Master) o esclavos (Slaves).

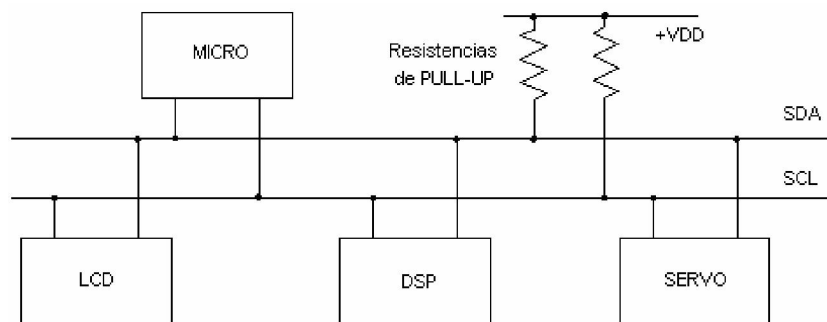


Figura 2.38 Esquema I2C

**Tabla 2.7 Terminología del Bus I2C**

TERMINOLOGÍA BÁSICA DEL BUS I2C	
Términos	Descripción
Transmisor	El dispositivo que envía datos al Bus
Receptor	El dispositivo que recibe datos desde el Bus
Master (Maestro)	El dispositivo que inicia una transferencia, genera las señales de reloj y termina un envío de datos
Slave (Esclavo)	El dispositivo direccionado por un master

M u l t i M a s t e r	M a s d e u n m a s t e r p u e d e c o n t r o l a r e l b u s a l m i s m o t i e m p o s i n c o r r u p c i ó n d e l o s m e n s a j e s
A r b i t r a j e	P r o c e d i m i e n t o q u e a s e g u r a q u e s i u n o o m a s m a s t e r s i m u l t á n e a m e n t e d e c i d e n c o n t r o l a r e l B u s s o l o u n o e s p e r m i t i d o a c o n t r o l a r l o y e l m e n s a j e s a l i e n t e n o e s d e t e r i o r a d o
S i n c r o n i z a c i ó n	P r o c e d i m i e n t o p a r a s i n c r o n i z a r l a s s e ñ a l e s d e l r e l o j d e d o s o m a s d i s p o s i t i v o s

### Generalidades

Los master son generalmente microcontroladores, por lo que un microcontrolador puede ser unas veces Master y otras esclavo.

Para imaginar la imagen del Bus son dos cables a los que se conectan diversos circuitos o chips en cantidad variable según las necesidades, controlado el conjunto por uno o más microcontroladores que dan instrucciones para el buen funcionamiento del conjunto.

La posibilidad de conectar más de un microcontrolador al Bus significa que uno o más microcontroladores pueden iniciar el envío de datos al mismo tiempo. Para prevenir el caos que esto ocasionaría se ha desarrollado un sistema de arbitraje.

Si uno o mas master intentan poner información en el bus es la señal del reloj si esta a "1" o a "0" lo que determina los derechos de arbitraje.

La generación de señales de reloj (SCL) es siempre responsabilidad de los dispositivos Master, cada Master genera su propia señal de reloj cuando



envía datos al bus, las señales de reloj de un master solo pueden ser alteradas cuando la línea de reloj sufre una caída por un dispositivo esclavo o por el dominio del control del Bus por el arbitraje de otro microcontrolador.

Los dispositivos conectados al bus deben ser de colector abierto o drenaje abierto ("en paralelo"), así los estados de salida de las líneas de reloj (SCL) y dato (SDA) desempeñan la función de "cable en AND" del bus.

Durante el tiempo en que no hay transferencia de datos (tiempo inactivo), tanto la línea del reloj (SCL) como la línea de datos (SDA) son "tiradas" arriba a través de resistencias externas pull-up.

La única limitación en la conexión de dispositivos al bus depende de la capacidad máxima que no puede superar los 400 pF. Los tipos de transferencia de datos en el bus son:

Modo Estándar aproximadamente a 100 kBits/Sg.

Modo Rápido aproximadamente a 400kBits/Sg.

Modo Alta velocidad mas de 3,4Mbits/Sg.

## TRANSFERENCIA DEL BIT

Debido a la variedad de diferentes tecnologías usadas en los dispositivos conectados al Bus I2C los niveles lógicos de "0" (Bajo) y "1" (Alto) no están

fijos y dependen de la tensión de alimentación del circuito. Un pulso de reloj se genera por cada bit de datos transferidos.

Los bits de datos transferidos en la línea SDA deben ser estables cuando la línea SCL está a nivel "1". El estado de la línea SDA en "1" o "0" solo puede cambiar cuando en la línea SCL la señal es "0". Ver Fig 2.39.

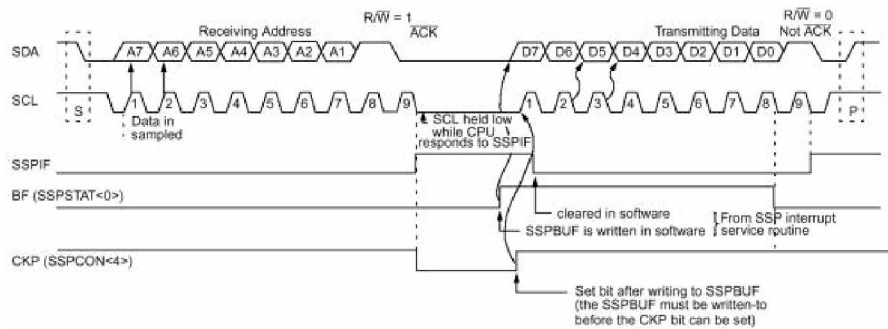


Figura 2.39 Transferencia I2C.

## Instrucciones en el I2C Bus

Para operar un esclavo sobre el Bus I2C solo son necesarios seis simples códigos, suficientes para enviar o recibir información.

Un bit de Inicio

7-bit o 10-bit de direccionamiento

Un R/W bit que define si el esclavo es transmisor o receptor

Un bit de reconocimiento

Mensaje dividido en bytes

Un bit de Stop

### Condiciones de Inicio (Start) y Stop

Dentro del proceso de transferencia de datos en el Bus I2C hay dos situaciones básicas que son el Inicio y el Stop de toda transferencia de datos. Estas son:

**INICIO (START)** - Una transición de "1" a "0" (caída) en la línea de datos (SDA) mientras la línea de reloj (SCL) está a "1".

**PARADA (STOP)** - Una transición de "0" a "1" (ascenso) en la línea de datos (SDA) mientras la línea de reloj (SCL) está a "1".

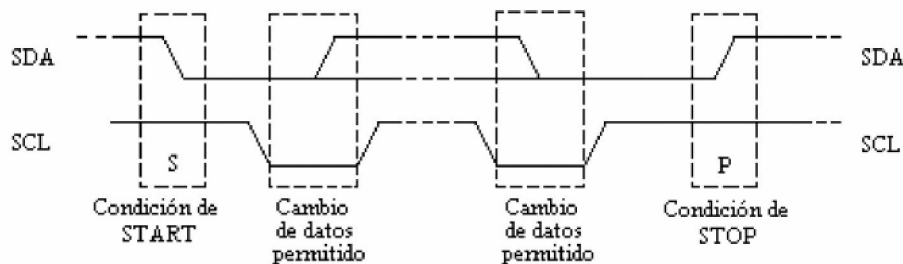


Figura 2.40 Condiciones Start, Stop.

Las condiciones de Inicio y Stop son siempre generadas por el Master. El Bus I2C se considera ocupado después de la condición de Inicio. El Bus se considera libre de nuevo después de un cierto tiempo tras la condición de Stop.

Es decir al pulso "1" de la línea SCL le puede corresponder un pulso "0" o "1" de la línea SDA en función de la información del byte que se envíe, recordemos que a cada bit de SDA le corresponde un bit de SCL, pero nunca salvo en la condición de Inicio a un bit de SCL le corresponde una situación de "1" a "0" o sea pasa por dos estados la línea SDA al revés ocurre en la condición de Stop que el Master envía un bit a la línea SCL mientras cambia en la SDA de "0" a "1" durante el tiempo que esta enviando la señal de "1" a SCL [35].

## TRANSFIRIENDO DATOS

El número de bytes que se envíen a la línea SDA no tiene restricción. Cada byte debe ir seguido por un bit de reconocimiento, el byte de datos se transfiere empezando por el bit de mas peso (7) precedido por el bit de reconocimiento (ACK).

**Tabla 2.8 Estado de transferencia de datos**

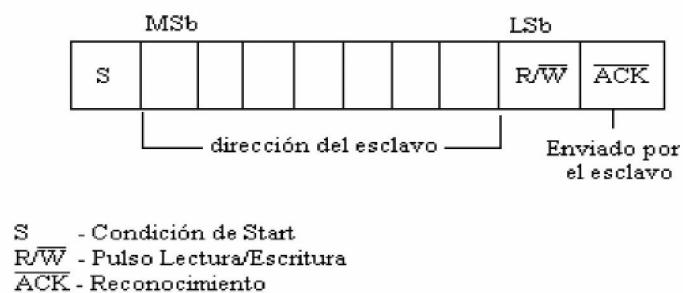
Estado De los bits en la transferencia		SSPSR a SSPBUF	Genera impulso ACK	Sube a uno el bit SSPIF (CCP genera interrupción si está habilitada)
BF	SSPOV			
0	0	SI	SI	SI
1	0	NO	NO	SI
1	1	NO	NO	SI
0	1	SI	NO	SI

Si un dispositivo esclavo no puede recibir o transmitir un byte de datos completo hasta que haya acabado alguno de los trabajos que realiza, puede mantener la línea SCL a "0" lo que fuerza al Master a permanecer en un estado de espera. Los datos continúan transfiriéndose cuando el dispositivo esclavo está listo para otro byte de datos y desbloquea la línea de reloj SCL.

## RECONOCIMIENTO

El bit de reconocimiento es obligatorio en la transferencia de datos. El pulso de reloj correspondiente al bit de reconocimiento (ACK) es generado por el Master. El Transmisor desbloquea la línea SDA ("1") durante el pulso de reconocimiento.

El receptor debe poner a "0" la línea SDA durante el pulso ACK de modo que siga siendo "0" durante el tiempo que el master genera el pulso "1" de ACK.



Normalmente un receptor cuando ha sido direccionado está obligado a generar un ACK después de que cada byte ha sido recibido.

Cuando un dispositivo esclavo no genera el bit ACK (porque esta haciendo otra cosa y no puede atender el Bus) debem mantener el esclavo la línea SDA a nivel "1" durante el bit ACK. El Master entonces puede generar una condición de STOP abortando la transferencia de datos o repetir la condición de Inicio enviando una nueva transferencia de datos.

Si un Esclavo-receptor que esta direccionado no desea recibir más bytes, el master debe detectar la situación y no enviar más bytes. Esto se indica porque el esclavo no genera el bit ACK en el primer byte que sigue. El esclavo pone la línea SDA a "1" lo que es detectado por el Master el cual genera la condición de Stop o repite la condición de Inicio.

Si un Master-receptor esta recibiendo datos de un Esclavo-transmisor debe generar un bit ACK tras cada byte recibido de transmisor, para finalizar la transferencia de datos no debe generar el ACK tras el último byte enviado por el esclavo. El esclavo-transmisor debe permitir desbloquear la línea SDA generando el master la condición de Stop o de Inicio [35].

## **Arbitraje y generación de señales de Reloj**

### **Sincronización**

Todos los Master generan su propia señal de reloj sobre la línea SCL al transferir datos sobre el Bus I2C. Los bit de datos son solo validos durante los periodos "1" del reloj. Un control es necesario para mantener un orden en los diversos bit que se generan.

La sincronización del reloj se realiza mediante una conexión AND de todos los dispositivos del Bus a la línea SCL. Esto significa que una transición de un Master de "1" a "0" en la línea SCL hace que la línea pase a "0", esto mantiene la línea SCL en ese estado. Sin embargo la transición de "0" a "1" no cambia el estado de la línea SCL si otro reloj está todavía en su periodo de "0". Por lo tanto la línea SCL permanecerá a "0" tanto como el periodo mas largo de cualquier dispositivo cuyo nivel sea "0". Los dispositivos que tienen un periodo mas corto de reloj "0" entran en un periodo de espera.

Cuando todos los dispositivos conectados al Bus han terminado con su periodo "0", la línea del reloj se desbloquea y pasa a nivel "1". Por lo que hay que diferenciar entre los estados de reloj de los dispositivos y los estados de la línea SCL, y todos los dispositivos empiezan a nivel "1". El primer dispositivo que completa su nivel "1" pone nuevamente la línea SCL a "0".

Resumiendo, la sincronización de la línea SCL se genera a través de la señal a "0" por el dispositivo con el mas largo periodo de nivel a "0" y la señal a "1" por el dispositivo con el mas corto periodo de nivel a "1".

### **Arbitraje**

Un master puede iniciar una transmisión solo si el bus está libre. Dos o más masters pueden generar una condición de Inicio en el bus lo que da como resultado una condición de Inicio general. Cada Master debe comprobar si el bit de datos que transmite junto a su pulso de reloj, coincide con el nivel lógico en

la línea de datos SDA. El sistema de arbitraje actúa sobre la línea de datos SDA, mientras la línea SCL está a nivel "1", de una manera tal que el master que transmite un nivel "1", pierde el arbitraje sobre otro master que envía un nivel "0" a la línea de datos SDA. Esta situación continua hasta que se detecte la condición de Stop generada por el master que se hizo cargo del Bus.

El arbitraje puede continuar varios bits hasta que se de la circunstancia de control del Bus por uno de los Masters.

Tras el arbitraje los Masters perdedores se deben poner inmediatamente en modo Master-receptor y esclavo pues los datos que envíe el Master dominante pueden ser para uno de ellos.

Un master que pierde el arbitraje puede generar pulsos de reloj hasta el fin de byte en el cual el pierde el arbitraje.

En el momento que un master toma el control solo este master toma las decisiones y genera los códigos de dirección, no existen master centrales, ni existen ordenes prioritarias en el Bus.

Especial atención debe ponerse si durante una transferencia de datos el procedimiento de arbitraje está todavía en proceso justo en el momento en el que se envía al Bus una condición de Stop. Es posible que esta situación pueda ocurrir en este caso el master afectado deberá mandar códigos de Inicio o Stop.



## ESPECIFICACIONES ELÉCTRICAS Y DE TIEMPOS

Este tema por su complejidad debe ser ampliado en el manual de Philips, en forma muy resumida y para circuitos sencillos vale lo que sigue.

Dada la gran cantidad de diferentes dispositivos que se pueden conectar en el Bus I2C las tensiones dependen por un lado de las necesarias para cada uno de los componentes y de una cierta normativa bastante elástica para las líneas SDA y SCL.

Se debe pretender que la alimentación de las líneas SCL y SDA debe ser a 5V lt. manteniéndose las siguientes tolerancias:

Máxima tensión permitida a nivel bajo ("0") --> 1,5 V lt.

Mínima tensión permitida a nivel alto ("1") --> 3 V lts.

### **2.5 Diseño de la placa.**

Para el diseño del hardware se buscarán los procesos necesarios para que el microcontrolador funcione de acuerdo a las especificaciones necesarias y le permite interactuar con el mundo exterior.

A continuación definiremos las señales de Entrada/Salida y alimentación, seleccionaremos los módulos del microcontrolador que vamos a utilizar y sus respectivos pines, para finalizar diseñaremos la placa del circuito impreso.

#### **2.5.1 Placa USB.**

Se considero pertinente utilizar un microcontrolador Prolific® para la transformación de USB a RS232, la razón de esta selección fue su disponibilidad en el mercado.

El microcontrolador seleccionado fue el PL-2330 SSOP de 28 pines, el mismo que cuenta dentro de su estructura una memoria eeprom bajo protocolo I2C una entrada Full Duplex para UART y otra para USB, lo que favorece su labor y sencillez en la programación de la aplicación final deseada[25].

A continuación se muestra el esquema de bloques del microcontrolador seleccionado.

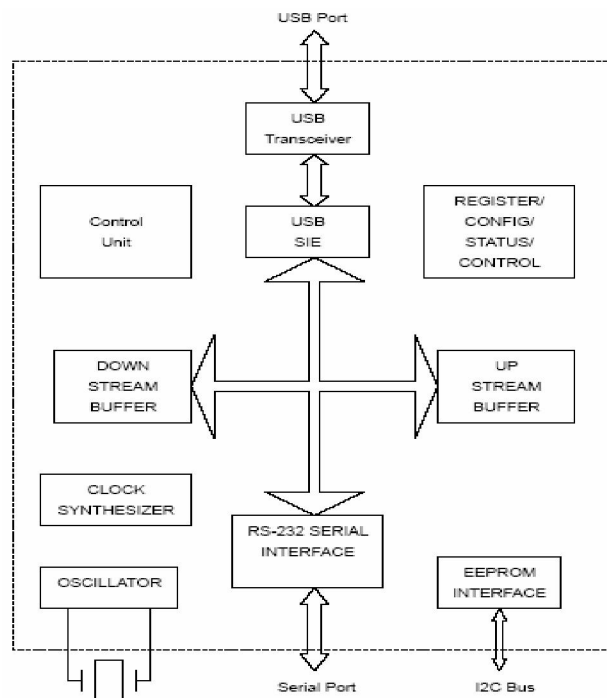


Figura 2.42 Diagrama de bloques del PL2330.

Este microcontrolador permite transferir a una razón de 1228800 baudios con la dirección 0012C000h

La asignación de pines así como las conexiones necesarias para la transformación de protocolo USB a RS232 Full Duplex se explican en el siguiente diagrama, mostrando además, los voltajes necesarios para el correcto funcionamiento.



## 2.5.2 Placa Sensores

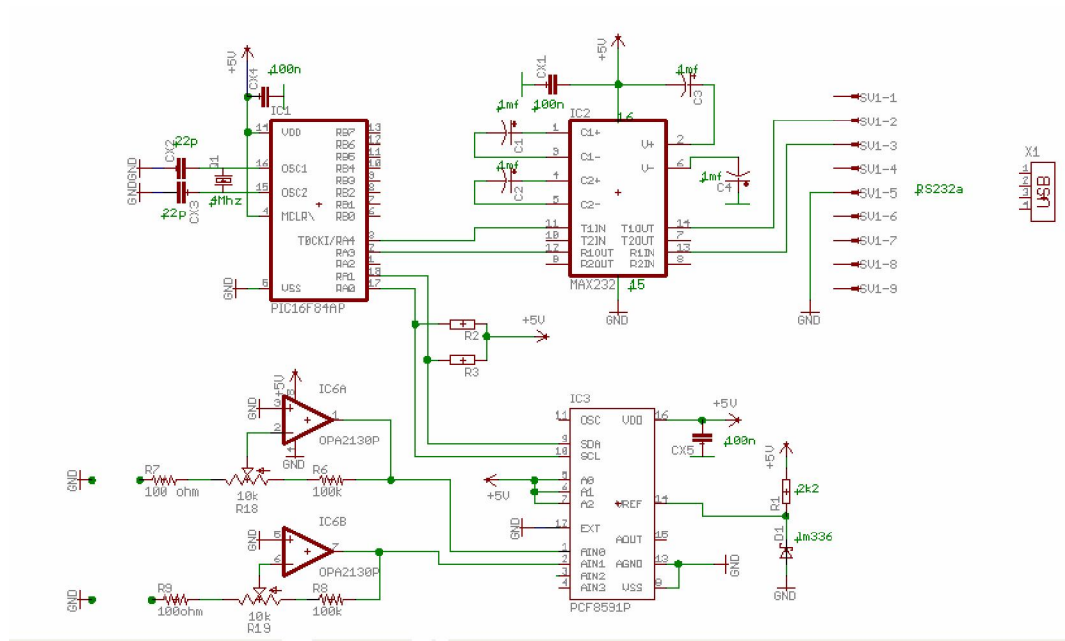


Figura 2.44 Esquema de conexión del PIC con termopares.

En este diseño se empleó el DS2760 como amplificador de señal y un voltímetro integrado por un PCF8591P conversor análogo digital de 10 B it con protocolo I2C, se pensó inicialmente en otro sistema con un conversor de temperatura de MAX667X, pero no fue posible encontrarlo por ello se utilizó el diagrama expuesto anteriormente y permitir que por medio de rutinas de software se transforme el voltaje a medidas de temperatura.

Otra posibilidad de diseño fue utilizar una configuración comercial con base a un microcontrolador de microchip el 16C57 como se muestra en la siguiente figura:

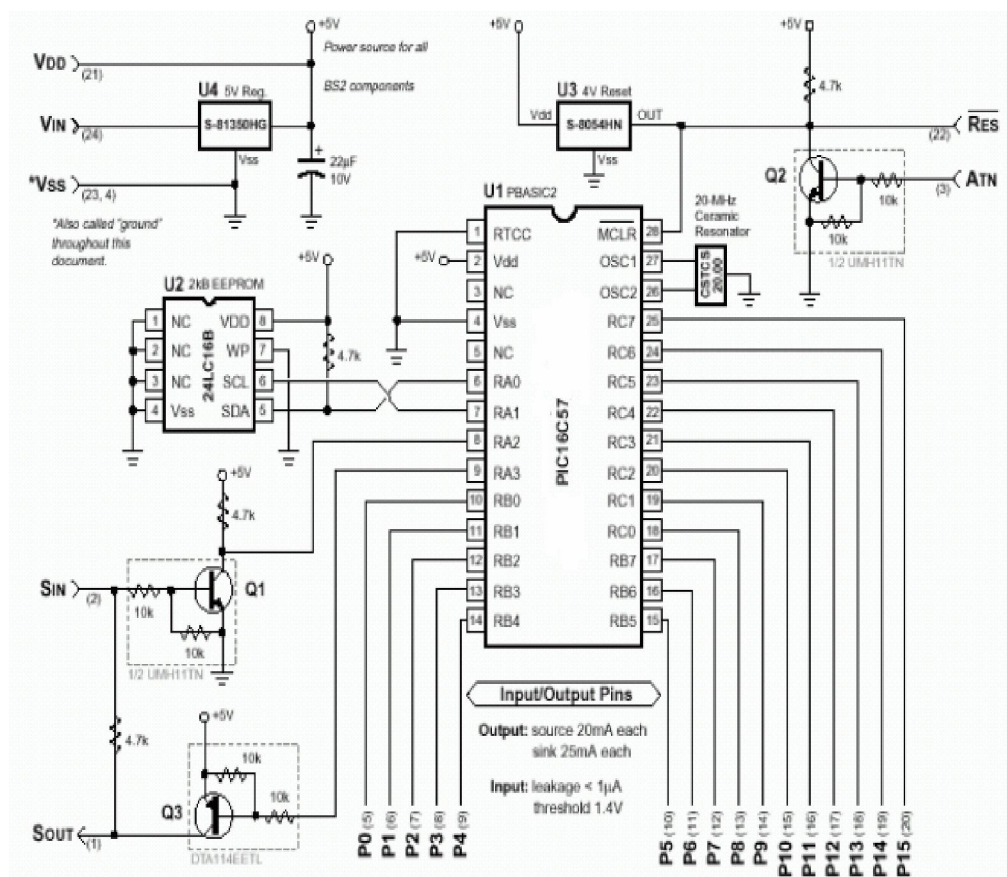


Figura 2.45 Esquema de conexión del PIC comercial

Si bien el diseño es robusto, la velocidad de transmisión de datos así como la cantidad de información es mayor al de la figura 2.44, pero aun no es lo suficientemente rápida para evitar la saturación del bus de conversión USB-UART, ya que la máxima capacidad es 9600 baudios, por ello se busca otra configuración para alcanzar los 19200 baudios como la configuración que muestra

la figura 2.46, además de aumentar su capacidad de memoria evita la saturación del bus pudiendo incorporar más componentes al diagrama original.

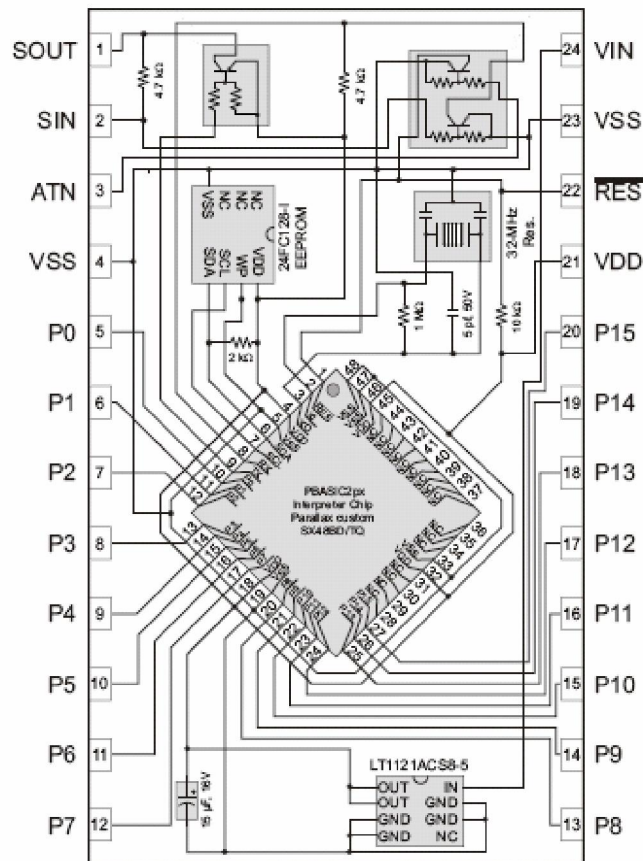


Figura 2.46 Esquema de conexión del PIC SX48BD/TQ

### 2.5.3 Termopares

Los termopares se utilizan extensamente, ya que ofrecen una gama de temperaturas mucho más amplia y una construcción más robusta que otros tipos. Además, no precisan alimentación de ningún tipo y su reducido precio los convierte en una opción muy atractiva para grandes sistemas de adquisición de datos. Sin embargo, para superar algunos de los inconvenientes inherentes a los

termopares y obtener resultados de calidad, es importante entender la naturaleza de estos dispositivos.

Estudios realizados sobre el comportamiento de termopares han permitido establecer tres leyes fundamentales:

1. Ley del circuito homogéneo. En un conductor metálico homogéneo no puede sostenerse la circulación de una corriente eléctrica por la aplicación exclusiva de calor.

2. Ley de metales intermedios. Si en un circuito de varios conductores la temperatura es uniforme desde un punto de soldadura A a otro punto B, la suma algebraica de todas las fuerzas electromotrices es totalmente independiente de los conductores metálicos intermedios y es la misma que si se pusieran en contacto directo A y B.

3. Ley de las temperaturas sucesivas. La fem. generada por un termopar con sus uniones a las temperaturas  $T_1$   $T_3$  es la suma algebraica de la fem. del termopar con sus uniones a  $T_1$   $T_2$  de la fem. del mismo termopar con sus uniones a las temperaturas  $T_2$   $T_3$ .

### **¿Cómo funcionan los Termopares?**

El comportamiento de un termopar se basa en la teoría del gradiente, según la cual los propios hilos constituyen el sensor. La figura 2.47a ilustra este concepto. Cuando se calienta uno de los extremos de un hilo, le produce una tensión que es una función de (A) el gradiente de temperatura desde uno de los extremos del hilo al otro, y (B) el coeficiente de Seebeck, una constante de proporcionalidad que varía de un metal a otro.



Un termopar se compone sencillamente de dos hilos de diferentes metales unidos en un extremo y abiertos en el otro (Figura 2.47b). La tensión que pasa por el extremo abierto es una función tanto de la temperatura de la unión como de los metales utilizados en los dos hilos. Todos los pares de metales distintos presentan esta tensión, denominada tensión de Seebeck en honor a su descubridor, Thomas Seebeck..

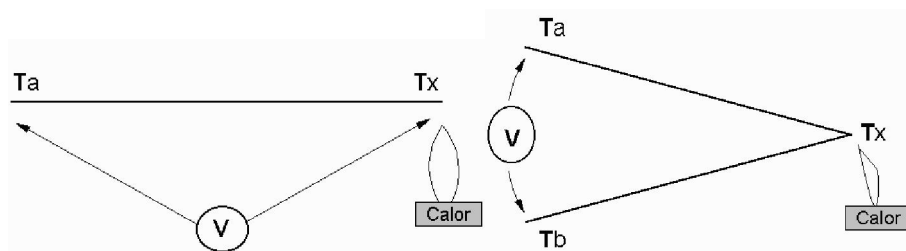


Figura 2.47 a) Termopar lineal

Figura 2.47 b) Termopar de dos hilos

En pequeñas gamas de temperaturas, los coeficientes de Seebeck de los dos hilos son constantes y la tensión de Seebeck es, por consiguiente, proporcional, pero en gamas más grandes, el propio coeficiente de Seebeck es una función de la temperatura, convirtiendo la tensión de Seebeck en no lineal. Como consecuencia, las tensiones del termopar también tienden a ser no lineales.

### Temperatura relativa frente a temperatura absoluta

Los RTD, termistores y sensores de IC miden todos ellos temperaturas absolutas, pero el termopar mide solamente temperaturas relativas, y el motivo resulta obvio cuando pensamos en la conexión de un termopar a un voltímetro o a

un sistema de adquisición de datos. Supongamos que estamos utilizando un termopar Tipo J, que es el más normal y consiste en un hilo de hierro y otro de constantan (una aleación con un 45% de níquel y un 55% de cobre). ¿Qué ocurrirá cuando conectemos los dos hilos conductores de prueba, que probablemente sean de cobre? Que crearemos otros dos termopares (Figura 2.48), cada uno de los cuales aportará una tensión al circuito, con lo que tendremos tres termopares y tres temperaturas desconocidas.

La solución clásica a este dilema consiste en añadir un termopar opuesto y una unión de referencia a una temperatura conocida (Figura 2.49). En este ejemplo, el termopar opuesto es otra unión de cobre y hierro equivalente a la unión de cobre y hierro que hemos creado al añadir un hilo conductor de cobre al hilo conductor de hierro del termopar "real". Estas dos uniones, si están aisladas en un bloque isotérmico (temperatura constante), se anularán mutuamente.

Ahora tenemos sólo dos uniones, la unión original del termopar ( $T_x$ ) y la de referencia ( $T_{ref}$ ) que acabamos de añadir. Si conocemos la temperatura de la unión de referencia, podremos calcular  $T_x$ . (Muchos sistemas de adquisición de datos y muchos voltímetros que efectúan medidas con un termopar realizan este cálculo de forma automática.)

Lamentablemente, la naturaleza de la temperatura dificulta un poco las cosas en este caso, ya que hay muy pocos puntos de referencia prácticos y

económicos para la temperatura. Los puntos de congelación y ebullición del agua, a 0 y a 100 °C respectivamente, son prácticamente los únicos aseguibles que nos ofrecen la Madre Naturaleza. Una forma habitual de determinar la temperatura de Tref es introducir físicamente la unión en un baño de hielo, forzando la temperatura a 0 °C. De hecho, todas las tablas de termopares utilizan un baño de hielo como referencia.

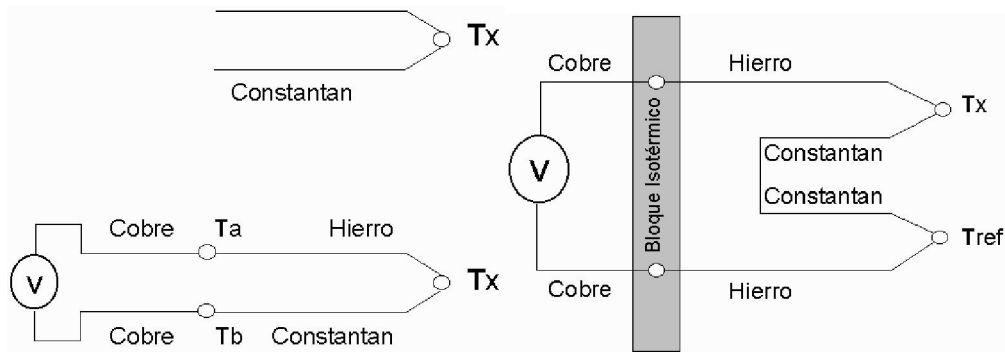


Figura 2.48 termopar normal

Figura 2.49 Termopar con bloque isotérmico

### Ahora, para simplificar el panorama

El enfoque del baño de hielo ofrece lecturas exactas, pero no es precisamente el accesorio más indicado para un sistema de adquisición de datos y, además, seguimos teniendo que conectar dos termopares. El primer paso hacia la simplificación es eliminar el baño de hielo. Si medimos Tref con un dispositivo de medida de temperaturas absolutas (como por ejemplo un RTD) y compensamos el resultado matemáticamente, no tenemos necesidad de forzarlo a 0 °C.

El siguiente paso es eliminar el segundo termopar (Figura 2.50). Ampliando el bloque isotérmico para incluir Tref, ajustamos la temperatura del bloque isotérmico a Tref (puesto que los otros dos termopares del bloque siguen anulándose mutuamente).

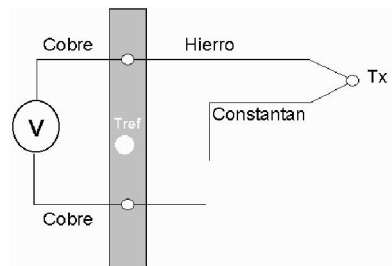


Figura 2.50 termopar tipo K

La determinación de Tref es cuestión de medir la temperatura del bloque isotérmico con el RTD o con otro cualquier dispositivo de medida de temperaturas absolutas.

### ¿Cómo llegar a la respuesta?

Tref es una de las dos cantidades que necesitamos conocer para calcular Tx. La otra es V, que medimos con el sistema de adquisición de datos (o voltímetro). Aplicando la fórmula:

$$V = (T_x - T_{ref})$$

Podemos calcular tensiones equivalentes para los dos valores de temperatura y a continuación restar para determinar el valor de Tx.

En realidad, no queremos hacer el cálculo nosotros mismos, ya que el coeficiente no lineal de Seebeck convierte esta tarea en un trabajo rutinario. Como hemos dicho anteriormente, los voltímetros y los sistemas de adquisición de datos que efectúan medidas con termopares se encargan a menudo de realizar el cálculo.

A propósito, los coeficientes de Seebeck y las tensiones de salida resultantes son números pequeños (ver la Tabla 2.9), por lo que resulta difícil medir con exactitud tanto los niveles absolutos como los cambios relativos. En este punto el ruido eléctrico puede alterar la precisión de las medidas de temperatura. El acoplamiento magnético y electrostático se reduce utilizando cable de par trenzado, reduciendo al mínimo la longitud de los hilos conductores y permaneciendo alejado de campos magnéticos y eléctricos intensos. Por último, pero no por ello menos importante, se necesita instrumentación capaz de realizar medidas de bajo nivel limpias.

**TABLA 2.9 COEFICIENTE DE SEEBECK**

Tipo de Termopar	a 0 °C	a 100 °C	Tensión de salida a 100 °C
B	-0,25 V/C	0,90 V/C	0,033 mV
E	58,7 V/C	67,5 V/C	6,32 mV
J	50,4 V/C	54,4 V/C	5,27 mV
K	39,5 V/C	41,4 V/C	4,10 mV
S	5,40 V/C	7,34 V/C	0,65 mV

Coeficientes de Seebeck y tensiones de salida para los termopares utilizados habitualmente. Las dos cifras que representan los coeficientes para cada uno de los tipos muestran la no linealidad a través de una amplia gama de temperaturas.

Un asunto adicional muy importante en el uso de termopares en la industria tiene que ver con la variación de la temperatura ambiente en las uniones frías. Esta es la situación: si supiéramos de antemano la temperatura de las uniones frías, entonces en lugar de relacionar la lectura del voltímetro con la diferencia de temperatura, se podría relacionarla con la temperatura de la unión caliente misma. Esto sería posible pues podríamos construir las tablas de temperatura contra voltaje para que reflejaran el hecho de que las uniones frías están a una cierta temperatura de referencia (como se le denomina) conocida.

Ejemplo:

Considere un termopar tipo J. A una diferencia de temperatura de 400 °F, el voltaje de la malla del termopar es de 12 mV. Si se supiera que la unión fría siempre estará, digamos a 75 °F, entonces podríamos concluir que un voltaje de la malla de 12 mV representaría una temperatura de la unión caliente de 475 °F

$$(475\text{ °F} - 75\text{ °F} = 400\text{ °F}).$$

Mientras la unión fría se mantuviera constantemente a la temperatura de referencia de 75°F se podría ir directamente a la tabla del termopar y sumar 75 °F a

cada lectura de diferencia de temperatura. El valor de temperatura resultante entonces representaría la temperatura de la unión caliente.

De hecho, esto es exactamente lo que se hace en las tablas de termopares industriales. La cifra de 75 °F se ha escogido porque representa una estimación bastante razonable de la temperatura ambiental promedio en una instalación industrial. (En las tablas de termopares para uso de laboratorio, se considera normalmente que la temperatura de referencia es de 32 °F, el punto de congelación del agua)

Para que el enfoque anterior funcione adecuadamente, la unión fría debe mantenerse constantemente a la temperatura de referencia de 75 °F. Esto por lo general es impráctico, a menos que el dispositivo de medición de temperatura esté colocado en un cuarto con aire acondicionado. Con toda seguridad, el equipo de medición estará ubicado junto con el equipo industrial y la maquinaria. La temperatura ambiente podrá variar con facilidad de unos 50°F en el invierno a unos 100 °F en el verano. Son comunes los cambios de estación aún mayores en la temperatura ambiente. Debido a esta variación en la temperatura de la unión fría, las mallas de termopares industriales deben ser compensadas.

**Tabla 2.10 Datos Técnicos de Referencia de las Termocuplas**

Tipo Termocupla	Materiales	Rango de Aplicación (°F)	mV
B	Platinum30% Rhodium (+)	100 – 3270	0.007-13.499
	Platinum 6% Rhodium (-)		

C	W5Re Tungsten 5% Rhenium (+)	3000-4200	-
	W26Re Tungsten 26% Rhenium (-)		
E	Chromel (+)	32 – 1800	0 – 75.12
	Constantan (-)		
J	Iron (+)	-300 – 1600	-7.52 – 50.05
	Constantan (-)		
K	Chromel (+)	-300 – 2300	-5.51 – 51.05
	Alumel (-)		
N	Nicrosil (+)	1200-2300	-
	Nisil (-)		
R	Platinum 13% Rhodium (+)	32 – 2900	0 – 18.636
	Platinum (-)		
S	Platinum 10% Rhodium (+)	32 – 2800	0 – 15.979
	Platinum (-)		
T	Copper (+)	-300 – 750	-5.28 – 20.80
	Constantan (-)		

### Códigos de color de los Termocuplas

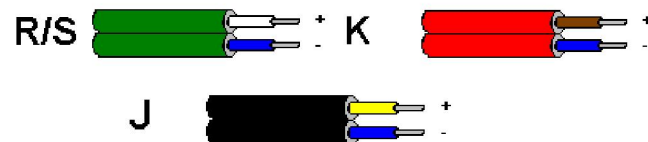
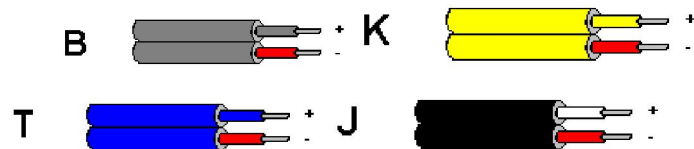
El alambrado de las termocuplas esta codificado dependiendo del tipo.

Diferentes países utilizan códigos diferentes para los colores. Los códigos más comunes son:

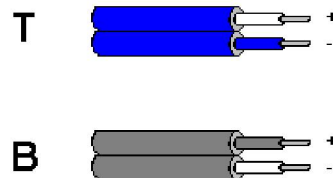




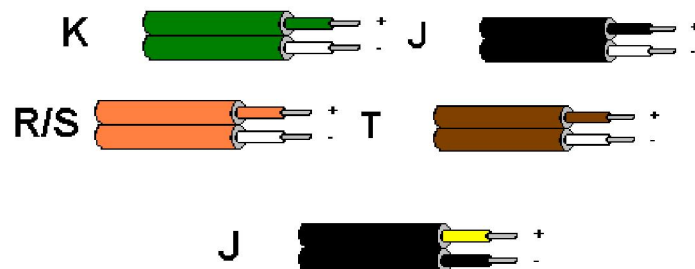
United States ASTM:



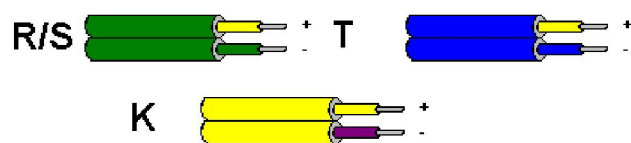
British BS1843: 1952:



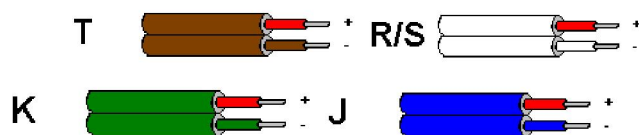
British BS4937: Part 30: 1993:



French NFE:



German DIN:



Se trabajo con termopar tipo K el cual presenta una ecuación de transformación segmentada la cual se explica a continuación:

Termopar tipo K para medir temperaturas desde -270°C hasta 0°C

$$E = \sum_{i=0}^{10} a_i t_{68}^i$$

donde:

$$\begin{aligned} a_0 &= 0 \\ a_1 &= 3.9475433139 \times 10^{-1} \\ a_2 &= 2.7465251138 \times 10^{-2} \\ a_3 &= -1.6565406716 \times 10^{-4} \\ a_4 &= -1.5190912392 \times 10^{-6} \\ a_5 &= -2.4581670924 \times 10^{-8} \\ a_6 &= -2.4757917816 \times 10^{-10} \\ a_7 &= -1.5585276173 \times 10^{-12} \\ a_8 &= -5.9729921255 \times 10^{-15} \\ a_9 &= -1.2688801216 \times 10^{-17} \\ a_{10} &= -1.1382797374 \times 10^{-20} \end{aligned}$$

Termopar tipo K para medir temperaturas desde 0°C hasta 1372°C

$$E = \sum_{i=0}^8 b_i t_{68}^i + 125 \exp \left[ -1/2 \left( \frac{t_{68}^i}{65} - 127 \right)^2 \right]$$

donde:

$$\begin{aligned} b_0 &= -1.8533063273 \times 10^1 \\ b_1 &= 3.8918344612 \times 10^1 \\ b_2 &= 1.6645154356 \times 10^{-2} \\ b_3 &= -7.8702374448 \times 10^{-5} \\ b_4 &= 2.2835785557 \times 10^{-7} \\ b_5 &= -3.5700231258 \times 10^{-10} \\ b_6 &= 2.9932909136 \times 10^{-13} \\ b_7 &= -1.2849848798 \times 10^{-16} \\ b_8 &= 2.2239974336 \times 10^{-20} \end{aligned}$$

Nota: En las ecuaciones la 't' significa la temperatura y la 'E' el voltaje de salida en milivoltios.

Para la lectura de temperatura se modificara la configuración de DS2760 de Dallas/Maxim, el cual es un monitor de baterías de litio de alta precisión, al interior de este posee un sensor de temperatura el cual hará las veces de unión fría (cero eléctrico), que permite hacer la corrección por efecto seebeck en rango de 0 a 127°C

para unión fría y de 0 a 1023°C para lectura, con resolución de 0.125, la configuración queda como la figura siguiente.

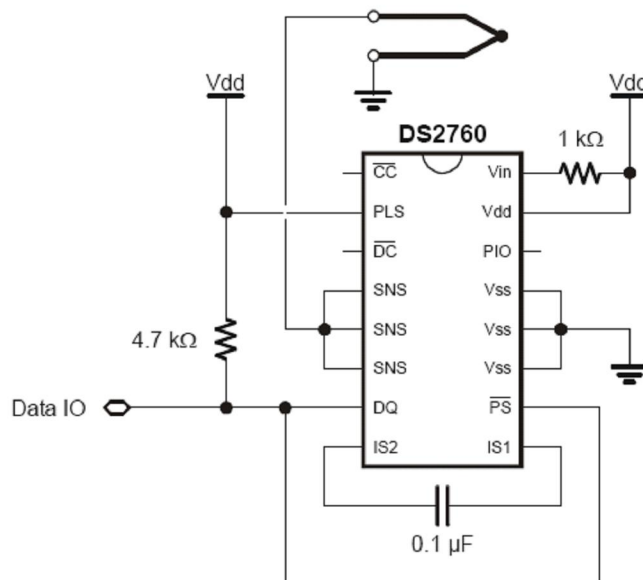


Figura 2.51 termopar digital tipo K

#### **2.5.4 Voltímetro.**

Para la adquisición de voltaje de fuente calorica se utiliza el ADC0831/2 el cual es un conversor de 8 bit serial de 0 a 5 volt en corriente continua, como el circuito traba de 0 a 10 volt se previó un circuito atenuador clasico con un potenciómetro de 100K par la lectura de 0 a 10 V transformarla de 0 a 5 Vpor 2, dejando la conversión de escalado via software; además es necesario la lectura de la intensidad del circuito, esto se puede leer al incorporar un transductor lineal de corriente como el LTS15-NP el cual por efecto Hall transforma la lectura a un voltímetro de 0 a 5 V .

#### **2.5.5 Diagrama final**

Al unir todo los diagramas anteriores el pic SX48DB/TQ, con los sensores de temperatura DS2760 y los voltímetros el diagrama final queda como la figura 2.52, la cual entregaria los datos bajo protocolo UART que serán transformados a USB por la figura 2.43

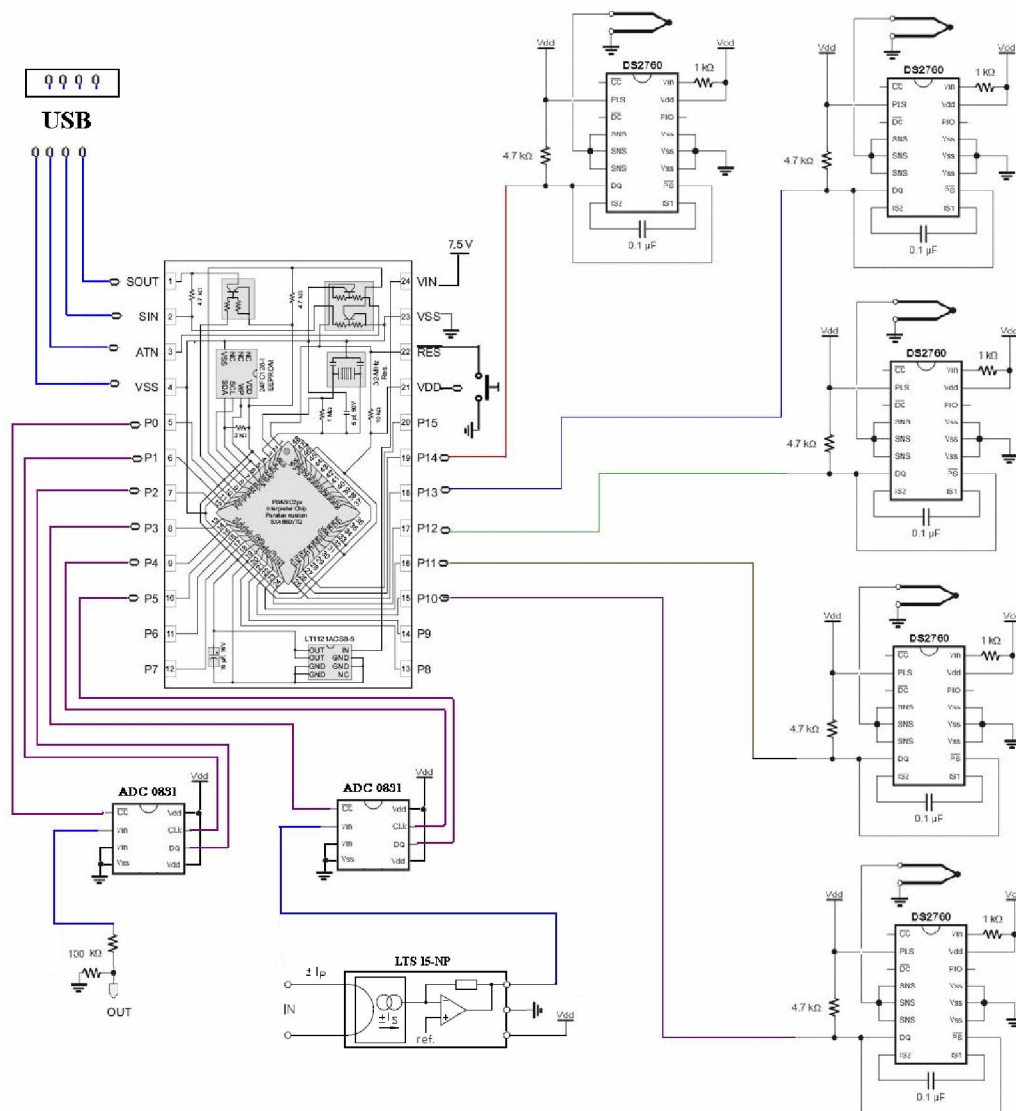


Figura 2.52 Diagrama final placa sensores



## ***Capítulo 3***

### **El Software de control (Frimware y Software)**

#### **3.1 Microcontrolador y sensores.**

La etapa de adquisición de datos de los sensores al pic básicamente es vía hardware, solo la conversión del protocolo y la comparación de la junta fría se realiza por medio de frimware el cual se muestra en el anexo 1.

#### **3.2 Software de control e interface.**

Para la aplicación final se tomo como referencia el Lenguaje Unificado de Modelamiento (UML) [14], el cual dicta las pautas para realizar una aplicación computacional, las mismas que se reflejan en el siguiente análisis.

##### **3.2.1 Requerimientos del sistema.**

Todo sistema tiene requerimientos. Dentro de los requerimientos existen los Iguientes

- ***No Funcionales***
- ***Funcionales***

Los primeros se refieren a tipo de hardware necesario para la aplicación, aplicaciones extras que la aplicación necesita, otro tipo de sistemas, sistema operativo, tiempos de respuesta etc.

Los segundos se refieren a los requerimientos que la aplicación tiene. Pueden ser evidentes, ocultos, etc. Los ocultos son las instrucciones que la aplicación realiza pero que el usuario no tiene idea de lo que ocurre. Los evidentes son eso, evidentes al usuario.

### 3.2.1.1 REQUERIMIENTOS NO FUNCIONALES

Los requerimientos mínimos para que la aplicación pueda ejecutarse correctamente son:

- Procesador Pentium III de 800Mhz o superior
- 128Mb RAM
- 128Mb de Memoria en Vídeo
- Sistema de Gestión de Base de Datos ( Opcional )
- Puerto USB disponible.
- Sistema Operativo Windows XP

### 3.2.1.2 REQUERIMIENTOS FUNCIONALES

<b>Código</b>	<b>Descripción</b>	<b>Tipo</b>
R1	Inicio de Sesión	Evidente
R2	Informe Gráfico	Evidente
R3	Análisis Gráfico	Evidente
R4	Configuración Proceso	Evidente
R5	Administración de Usuarios	Evidente
R6	Gráfico Termograma	Evidente
R7	Informe Resumen	Evidente
R8	Informe Termograma	Evidente
R9	Informe Progreso Adquisición de Datos	Evidente
R10	Administración Materias Primas	Evidente
R11	Conexión a Instrumento	Oculto
R12	Verificación de Instrumento	Oculto
R13	Adquisición de Datos	Oculto
R14	Verificación de Usuario	Oculto
R15	Análisis Matemático	Oculto
R16	Análisis Termograma	Oculto
R17	Conexión a Sistema Persistente	Oculto
R18	Configuración Aplicación	Oculto
R19	Cálculo Constantes Térmicas	Oculto

Todo lo que se ve en la tabla y es oculto, es lo que la aplicación realiza sin que el usuario se de cuenta de la realización de dichas operaciones.



Los requisitos que están marcados como evidentes es lo que el usuario percibe, es lo que el sistema realiza pero en este caso el usuario se da cuenta de lo que se está realizando.

Todo lo que es análisis es oculto, ya que los resultados se muestran de forma gráfica.

### 3.2.2 Casos de Uso

Los casos de uso son la secuencia de acciones que realiza el usuario de la aplicación y las respuestas que debe tener ésta en cada una de las acciones. Se definen además los flujos alternativos que puede tomar la aplicación.

Para las fichas de caso de uso se usa el formato de dos columnas.

#### 3.2.2.1 CASOS DE USO USUARIO – APLICACIÓN

Se toma como *actor principal*, para estas fichas, al usuario de la aplicación y como sistema la aplicación que realiza las operaciones solicitadas por el actor.

Se muestra a continuación una tabla con los casos de uso que se desarrollarán en las fichas siguientes.

Casos de Uso Usuario - Aplicación	
Ficha	Descripción
1	Adquirir Datos
2	Agregar Materia Prima
3	Agregar Usuario
4	Análisis Filtro ( Regresión polinomial )
5	Análisis Lineal
6	Análisis Termograma
7	Análisis Termopar
8	Calibración Instrumento
9	Cerrar Sesión
10	Eliminar Materia Prima
11	Eliminar Usuario
12	Informe log file
13	Informe Materias Primas
14	Informe Resumen Datos

15	Informe Termograma
16	Informe Termopares
17	Informe Usuarios
18	Iniciar Sesión
19	Modificar Materia Prima
20	Modificar Usuario

### **Ficha 1.** Adquirir Datos

**Actor Principal** Operador del sistema

**Interesados** Operador del sistema

**Precondiciones** No tiene

**Condiciones de éxito** Operación se realiza exitosamente en el tiempo especificado.

### **Escenario de éxito**

<b>Acción o Intensión del Actor</b>	<b>Respuesta del Sistema</b>
<b>1.</b> Solicita adquisición de datos.	<b>2.</b> Despliega ventana de configuración.
<b>3.</b> Configura la adquisición de datos.	<b>3.</b> Verifica conexión del instrumento.
	<b>4.</b> Verifica estado del instrumento.
	<b>5.</b> Despliega ventana de progreso.
	<b>6.</b> Adquiere los datos del instrumento.
	<b>7.</b> Despliega ventana de finalización.

### **Flujos**

#### **Alternativos**

- A1 Ocurrió un error en la configuración de la operación. Despliega mensaje y solicita configurar la operación nuevamente.
- A2 Error al verificar la conexión del instrumento. No se puede completar la operación.
- A3 Error en el estado del instrumento. Se despliega mensaje de error con el posible error. No se puede completar la operación.
- A4 Error al adquirir los datos. Se despliega mensaje de error. No se puede completar la operación.

- La verificación del estado del instrumento la solicita la aplicación al instrumento, luego de que la aplicación verifica la conexión con el instrumento.

### **Ficha 2.** Agregar Materia Prima

**Actor Principal** Operador del Sistema.

**Interesados** Operador del Sistema.

**Precondiciones** No tiene

**Condiciones de éxito** La materia prima no existe en los registros y se agregó correctamente.

### Escenario de éxito

Acción o Intención del Actor	Respuesta del Sistema
1. Solicita agregar materia prima.	2. Despliega ventana para agregar materia prima.
	3. Solicita ingresar los datos necesarios.
4. Ingresa los datos solicitados por el sistema.	5. Verifica que la materia prima no exista en los registros.
	6. Solicita confirmación de usuario.
7. Ingresa información de usuario.	8. Verifica al usuario.
	9. Agrega la nueva materia prima a los registros.
	10. Despliega mensaje de éxito.

### Flujos Alternativos

- A1 Ocurrió un error al verificar la existencia de la materia prima. Se despliega un mensaje con el error ocurrido.
- A2 Ocurrió un error al agregar la nueva materia prima. Se despliega mensaje de error con el error ocurrido. Se cancela la operación.
- A3 Información de usuario incorrecta. Despliega mensaje de error y solicita confirmación nuevamente.

### Ficha 3. Agregar Usuario

<b>Actor Principal</b>	Administrador del sistema
<b>Interesados</b>	Administrador del sistema
<b>Precondiciones</b>	No tiene
<b>Condiciones de éxito</b>	Se agrega el nuevo usuario.

### Escenario de éxito

Acción o Intención del Actor	Respuesta del Sistema
1. Solicita agregar un nuevo usuario.	2. Despliega ventana para ingreso de usuario.
	3. Solicita ingresar los datos necesarios para el nuevo usuario.
4. Ingresa los datos necesarios para el nuevo usuario.	5. Verifica que el nuevo usuario no exista en los registros.

7. Ingresar la información de usuario.
6. Solicita confirmación de usuario.
8. Verifica que sea usuario.
9. Agrega el nuevo usuario y la nueva configuración al registro.
10. Despliega mensaje de éxito.

#### Flujos Alternativos

- A1 Ocurrió un error al verificar la existencia del nuevo usuario. Se despliega un mensaje con el error ocurrido.
- A2 Ocurrió un error al agregar al nuevo usuario. Se despliega mensaje de error con el error ocurrido. Se cancela la operación.
- A3 Información de usuario incorrecta. Despliega mensaje de error y solicita confirmación nuevamente.

#### Ficha 4. Análisis Filtro (Regresión Polinomial)

<b>Actor Principal</b>	Operador del Sistema.
<b>Interesados</b>	Operador del Sistema.
<b>Precondiciones</b>	Datos deben existir para hacer el filtrado.
<b>Condiciones de éxito</b>	No tiene.

#### Escenario de éxito

Acción o Intención del Actor	Respuesta del Sistema
1. Solicita filtrar datos.	2. Despliega ventana para configuración del filtro.
3. Configura el filtro.	4. Filtra los datos y despliega una ventana con los resultados del filtro.

#### Flujos Alternativos

- A1 Error al filtrar los datos. Se despliega mensaje de error y se solicita comenzar con la operación nuevamente.
- A2 Error en la configuración. Despliega mensaje de error y solicita configurar nuevamente la operación.

#### Ficha 5. Análisis Lineal

<b>Actor Principal</b>	Operador del Sistema.
<b>Interesados</b>	Operador del Sistema.
<b>Precondiciones</b>	Deben existir datos para hacer el filtrado.
<b>Condiciones de éxito</b>	No tiene.
<b>Escenario de éxito</b>	

<b>Acción o Intensión del Actor</b>	<b>Respuesta del Sistema</b>
<b>1.</b> Solicita linealización datos.	<b>2.</b> Despliega resultado de la linealización de los datos.

**Flujos  
Alternativos**

- A1 Error al linealizar los datos. Despliega mensaje de error y cancela la operación.

**Ficha 6.** Análisis Termograma

<b>Actor Principal</b>	Operador del Sistema.
<b>Interesados</b>	Operador del Sistema.
<b>Precondiciones</b>	Datos deben existir para realizar el análisis.
<b>Condiciones de éxito</b>	No tiene.

**Escenario de éxito**

<b>Acción o Intensión del Actor</b>	<b>Respuesta del Sistema</b>
<b>1.</b> Solicita termograma.	<b>2.</b> Despliega termograma a partir de un set de datos.

**Flujos  
Alternativos**

- A1 Error al producir el termograma. Despliega mensaje de error. Se cancela la operación

**Ficha 7.** Análisis Termopar.

<b>Actor Principal</b>	Operador del Sistema
<b>Interesados</b>	Operador del Sistema
<b>Precondiciones</b>	No tiene.
<b>Condiciones de éxito</b>	No tiene.

**Escenario de éxito**

<b>Acción o Intensión del Actor</b>	<b>Respuesta del Sistema</b>
<b>1.</b> Solicita análisis termopares	<b>2.</b> Despliega análisis del termopar.

**Flujos  
Alternativos**

- A1 Error al producir el termograma. Despliega mensaje de error. Se cancela la operación

**Ficha 8.** Calibración instrumento.

**Actor Principal** Operador del Sistema.  
**Interesados** Operador del Sistema.  
**Precondiciones** No tiene.  
**Condiciones de éxito** No tiene.

#### Escenario de éxito

Acción o Intensión del Actor	Respuesta del Sistema
1. Solicita calibrar instrumento.	2. Verifica la conexión del instrumento. 3. Verifica el estado del instrumento. 4. Despliega instrucciones para la calibración.
5. Inicia la calibración.	6. Calibra el instrumento. 7. Despliega mensaje de éxito.

#### Flujos

##### Alternativos

- A1 Error al conectar al instrumento. Despliega mensaje de error y solicita verificar la conexión.
- A2 Error en el estado del instrumento. Despliega mensaje de error. Cancela la operación.

#### Ficha 9. Cerrar sesión.

**Actor Principal** Administrador del Sistema, Operador del Sistema.  
**Interesados** Administrador del Sistema, Operador del Sistema.  
**Precondiciones** No tiene.  
**Condiciones de éxito** No tiene.

#### Escenario de éxito

Acción o Intensión del Actor	Respuesta del Sistema
1. Solicita cerrar la sesión de experimentos.	2. Almacena la configuración del usuario en los registros. 3. Almacena el cierre de sesión en el registro de acciones. 4. Cierra todas las ventanas de sesión. 5. Despliega ventana de inicio de sesión.

#### Flujos

##### Alternativos

- A1 Error al almacenar configuración en los registros. Despliega mensaje de error. Continúa el cierre de sesión.

- A2 Error al almacenar en el registro de acciones. Despliega mensaje de error. Se almacena el error en el registro de errores. Continúa el cierre de sesión.

**Ficha 10.** Eliminar Materia Prima

**Actor Principal** Operador del Sistema.  
**Interesados** Operador del Sistema.  
**Precondiciones** No tiene.  
**Condiciones de éxito** No tiene.

**Escenario de éxito**

<b>Acción o Intensión del Actor</b>	<b>Respuesta del Sistema</b>
<b>1.</b> Solicita eliminar materia prima	<b>2.</b> Solicita el código de la materia prima a eliminar.
<b>3.</b> Selecciona la materia prima.	<b>4.</b> Verifica el registro por la materia prima.
	<b>5.</b> Solicita verificar usuario.
<b>6.</b> Ingresar información de verificación.	<b>7.</b> Verifica usuario.
	<b>8.</b> Elimina la materia prima.
	<b>9.</b> Despliega mensaje de éxito.

**Flujos**

**Alternativos**

- A1 Error al verificar materia prima. Despliega mensaje de error y cancela la operación.  
A2 Error al verificar usuario. Despliega mensaje de error y solicitar verificar usuario nuevamente.  
A3 Error al eliminar del registro. Despliega mensaje de error y cancela la operación.

**Ficha 11.** Eliminar Usuario

**Actor Principal** Administrador del Sistema.  
**Interesados** Administrador del Sistema.  
**Precondiciones** No tiene.  
**Condiciones de éxito** No tiene.

**Escenario de éxito**

<b>Acción o Intensión del Actor</b>	<b>Respuesta del Sistema</b>
<b>1.</b> Solicita eliminar usuario del registro	<b>2.</b> Solicita información del usuario a eliminar
<b>3.</b> Ingresar la información solicitada.	

- 4. Verifica la existencia del usuario en el registro.
- 5. Solicita verificación del administrador para eliminar el usuario.
- 6. Confirma la eliminación del usuario.
- 7. Elimina al usuario del registro.
- 8. Despliega mensaje de éxito.

### Flujos Alternativos

- A1 Error al verificar el usuario en el registro. Despliega mensaje de error y cancela la operación.
- A2 Error al confirmar la eliminación del usuario. Despliega mensaje de error y cancela la operación.
- A3 Error al eliminar del registro. Despliega mensaje de error y cancela la operación.

### Ficha 12. Informe Log File

**Actor Principal** Administrador del Sistema.  
**Interesados** Administrador del Sistema.  
**Precondiciones** No tiene.  
**Condiciones de éxito** No tiene.

### Escenario de éxito

Acción o Intensión del Actor	Respuesta del Sistema
1. Solicita informe del registro de sesiones.	2. Solicita confirmación del administrador.
3. Confirma la solicitud.	4. Despliega una previsualización del informe a imprimir.
5. Solicita imprimirlo	6. Imprime el informe y despliega mensaje de éxito.

### Flujos Alternativos

- A1 Error al verificar la confirmación del usuario. Despliega mensaje de error y cancela la operación.
- A2 Error al imprimir el informe. Despliega mensaje de error y cancela la operación.

### Ficha 13. Informe Materias Primas

**Actor Principal** Operador del Sistema  
**Interesados** Operador del Sistema



**Precondiciones** Existen materias primas en el registro.  
**Condiciones de éxito** No tiene.

**Escenario de éxito**

<b>Acción o Intensión del Actor</b>	<b>Respuesta del Sistema</b>
<b>1.</b> Solicita informe de materias primas.	<b>2.</b> Despliega previsualización del informe.
<b>3.</b> Solicita imprimir el informe.	<b>4.</b> Envía el informe a la impresora. <b>5.</b> Despliega mensaje de éxito.

**Flujos Alternativos**

- A1 Error al desplegar el informe. Despliega mensaje de error y cancela la operación.
- A2 Error al imprimir el informe. Despliega mensaje de error y cancela la operación.

**Ficha 14.** Informe Resumen de Datos

**Actor Principal** Operador del Sistema  
**Interesados** Operador del Sistema  
**Precondiciones** No tiene.  
**Condiciones de éxito** No tiene.

**Escenario de éxito**

<b>Acción o Intensión del Actor</b>	<b>Respuesta del Sistema</b>
<b>1.</b> Solicita informe resumido de los datos.	<b>2.</b> Despliega informe con el resumen de los datos.
<b>3.</b> Solicita imprimir el informe.	<b>4.</b> Solicita verificación del usuario.
<b>5.</b> Verifica ser el usuario.	<b>6.</b> Envía informe a la impresora. <b>7.</b> Despliega mensaje de éxito.

**Flujos Alternativos**

- A1 Error al desplegar el informe. Despliega mensaje de error y cancela la operación.
- A2 Error al verificar usuario. Despliega mensaje de error y solicita nueva verificación.
- A3 Error al imprimir el informe. Despliega mensaje de error y cancela la operación.

**Ficha 15.** Informe Termograma

**Actor Principal** Operador del Sistema  
**Interesados** Operador del Sistema  
**Precondiciones** No tiene.  
**Condiciones de éxito** No tiene.

**Escenario de éxito**

<b>Acción o Intensión del Actor</b>	<b>Respuesta del Sistema</b>
<b>1.</b> Solicita informe termograma	<b>2.</b> Despliega termograma.
<b>3.</b> Solicita imprimir.	<b>4.</b> Despliega previsualización de la impresión.
<b>5.</b> Confirma impresión.	<b>6.</b> Solicita verificación de usuario.
<b>7.</b> Verifica ser el usuario.	<b>8.</b> Envía informe a la impresora.
	<b>9.</b> Despliega mensaje de éxito.

**Flujos**

**Alternativos**

- A1 Error al desplegar el informe. Despliega mensaje de error y cancela la operación.
- A2 Error al verificar usuario. Despliega mensaje de error y solicita nueva verificación.
- A3 Error al imprimir el informe. Despliega mensaje de error y cancela la operación.

**Ficha 16.** Informe Termopares

**Actor Principal** Operador del Sistema  
**Interesados** Operador del Sistema  
**Precondiciones** No tiene.  
**Condiciones de éxito** No tiene.

**Escenario de éxito**

<b>Acción o Intensión del Actor</b>	<b>Respuesta del Sistema</b>
<b>1.</b> Solicita informe termopares	<b>2.</b> Despliega información sobre los termopares.
<b>3.</b> Solicita impresión.	<b>4.</b> Solicita verificación de usuario.
<b>5.</b> Verifica ser el usuario.	<b>6.</b> Despliega previsualización del informe.
<b>7.</b> Confirma impresión.	<b>8.</b> Envía informe a la impresora.

**9.** Despliega mensaje de éxito.

**Flujos  
Alternativos**

- A1 Error al desplegar el informe. Despliega mensaje de error y cancela la operación.
- A2 Error al verificar usuario. Despliega mensaje de error y solicita nueva verificación.
- A3 Error al imprimir el informe. Despliega mensaje de error y cancela la operación.

**Ficha 17.** Informe Usuarios

**Actor Principal** Administrador del Sistema  
**Interesados** Administrador del Sistema  
**Precondiciones** No tiene.  
**Condiciones de éxito** No tiene.

**Escenario de éxito**

<b>Acción o Intención del Actor</b>	<b>Respuesta del Sistema</b>
<b>1.</b> Solicita informe de usuarios.	<b>2.</b> Despliega información sobre los usuarios.
<b>3.</b> Solicita impresión.	<b>4.</b> Solicita verificación de usuario.
<b>5.</b> Verifica ser el usuario.	<b>6.</b> Despliega previsualización del informe.
<b>7.</b> Confirma impresión.	<b>8.</b> Envía informe a la impresora.
	<b>9.</b> Despliega mensaje de éxito.

**Flujos  
Alternativos**

- A1 Error al desplegar el informe. Despliega mensaje de error y cancela la operación.
- A2 Error al verificar usuario. Despliega mensaje de error y solicita nueva verificación.
- A3 Error al imprimir el informe. Despliega mensaje de error y cancela la operación.

**Ficha 18.** Iniciar sesión.

**Actor Principal** Administrador del Sistema, Operador del Sistema.  
**Interesados** Administrador del Sistema, Operador del Sistema.  
**Precondiciones** No tiene.

**Condiciones de éxito** No tiene.

**Escenario de éxito**

<b>Acción o Intensión del Actor</b>	<b>Respuesta del Sistema</b>
<b>1.</b> Solicita iniciar sesión.	<b>2.</b> Solicita nombre de usuario y contraseña
<b>3.</b> Ingresa su nombre de usuario y contraseña.	<b>4.</b> Verifica al usuario y la contraseña en los registros.
	<b>5.</b> Carga la configuración del usuario.
	<b>6.</b> Registra el inicio de sesión en el registro de sesiones.
	<b>7.</b> Despliega menú de operaciones de la aplicación.

**Flujos**

**Alternativos**

- A1 Error al verificar usuario. Despliega mensaje de error y solicita iniciar sesión nuevamente.
- A2 Error al cargar configuración del usuario. Despliega mensaje de error. Crea una nueva configuración y solicita iniciar nuevamente la sesión.
- A3 Error al registrar el inicio de sesión. Despliega mensaje de error y se cancela la operación.

**Ficha 19.** Modificar Materia Prima

**Actor Principal** Operador del Sistema.

**Interesados** Operador del Sistema.

**Precondiciones** No tiene.

**Condiciones de éxito** No tiene.

**Escenario de éxito**

<b>Acción o Intensión del Actor</b>	<b>Respuesta del Sistema</b>
<b>1.</b> Solicita modificación materia prima.	<b>2.</b> Solicita información sobre la materia prima.
<b>3.</b> Ingresa la información necesaria para identificar la materia prima.	<b>4.</b> Verifica que exista en el registro.
	<b>5.</b> Despliega la información de la materia prima.
<b>6.</b> Modifica la información.	<b>7.</b> Solicita verificar usuario.
<b>8.</b> Ingresa la información para verificar	

usuario.

- 9.** Verifica usuario.
- 10.** Modifica la información en el registro.
- 11.** Despliega mensaje de éxito.

### Flujos Alternativos

- A1 Error al verificar la materia prima en el registro. Despliega mensaje de error y solicita nueva información.
- A2 Error al verificar usuario. Despliega mensaje de error y se solicita nueva verificación.
- A3 Error al modificar la información. Despliega mensaje de error y cancela la operación.

### Ficha 20. Modificar Usuario

**Actor Principal** Administrador del Sistema.  
**Interesados** Administrador del Sistema.  
**Precondiciones** No tiene.  
**Condiciones de éxito** No tiene.

### Escenario de éxito

Acción o Intención del Actor	Respuesta del Sistema
<b>1.</b> Solicita modificación de usuario.	<b>2.</b> Solicita información sobre el usuario.
<b>3.</b> Ingresa la información necesaria para identificar al usuario.	<b>4.</b> Verifica que exista en el registro. <b>5.</b> Despliega la información del usuario.
<b>6.</b> Modifica la información.	<b>7.</b> Solicita verificar usuario.
<b>8.</b> Ingresa la información para verificar usuario.	<b>9.</b> Verifica usuario. <b>10.</b> Modifica la información en el registro. <b>11.</b> Despliega mensaje de éxito.

### Flujos Alternativos

- A1 Error al verificar usuario en el registro. Despliega mensaje de error y solicita nueva información.
- A2 Error al verificar usuario. Despliega mensaje de error y se solicita nueva verificación.

- A3 Error al modificar la información. Despliega mensaje de error y cancela la operación.

### 3.2.2.2 CASOS DE USO APLICACIÓN- INSTRUMENTO

En este caso el *actor principal* no es una persona como el administrador o el operador del sistema. En los casos de uso anteriores el *actor principal* era uno de estos dos ya que era el que interactuaba con el sistema, pero en este caso, la aplicación se vuelve el actor puesto que procesa la solicitud del *operador* o *administrador* y este hace la solicitud al instrumento (para la experimentación no tiene interacción con el instrumento).

Casos de Uso Aplicación - Instrumento	
Ficha	Descripción
A1	Cerrar Conexión a Instrumento
A2	Transmisión de Datos
A3	Verificar Conexión a Instrumento
A4	Verificar Estado Instrumento

**Ficha A1.** Cerrar conexión a Instrumento

**Actor Principal** Aplicación  
**Interesados** Operador del Sistema  
**Precondiciones** No tiene.  
**Condiciones de éxito** No tiene.

#### Escenario de éxito

Acción o Intención del Actor	Respuesta del Sistema
1. Solicita verificar el estado del equipo.	2. Verifica el estado del equipo.
3. Solicita cierre de conexión.	4. Cierra la conexión.
5. Cierra la conexión.	

#### Flujos

##### Alternativos

- A1 Error en el estado del instrumento. Despliega mensaje de error y cancela la operación de desconexión.  
A3 Error al terminar la conexión. Despliega mensaje de error y cancela la operación.

**Ficha A2.** Transmisión de Datos.

**Actor Principal** Aplicación  
**Interesados** Operador del Sistema  
**Precondiciones** No tiene.  
**Condiciones de éxito** No tiene.

**Escenario de éxito**

<b>Acción o Intensión del Actor</b>		<b>Respuesta del Sistema</b>
<b>1.</b>	Verifica conexión con el instrumento.	
<b>2.</b>	Solicita comenzar transmisión.	
		<b>3.</b> Envía la información del instrumento.
		<b>4.</b> Termina el envío de información.
<b>5.</b>	Cierra la conexión.	
		<b>6.</b> Cierra la conexión.

**Flujos Alternativos**

- A1 Error al verificar la conexión. Despliega mensaje de error y cancela la operación.
- A3 Error en el envío de información. Despliega mensaje de error y cancela la operación.
- A4 Error al cerrar la conexión. Despliega error. Salva los datos adquiridos del instrumento. Termina la sesión.

**Ficha A3.** Verificar conexión del instrumento.

**Actor Principal** Aplicación  
**Interesados** Operador del Sistema  
**Precondiciones** No tiene.  
**Condiciones de éxito** No tiene.

**Escenario de éxito**

<b>Acción o Intensión del Actor</b>		<b>Respuesta del Sistema</b>
<b>1.</b>	Solicita verificar el estado del instrumento.	
		<b>2.</b> Verifica el estado del equipo.
<b>3.</b>	Cierra la conexión.	
		<b>4.</b> Cierra la conexión.

**Flujos Alternativos**

- A1 Error en el estado del equipo. Despliega mensaje de error y cancela la operación de desconexión.
- A2 Error al cerrar la conexión. Despliega error. Salva los datos adquiridos del instrumento. Termina la sesión.

**Ficha A4.** Verificar Estado del instrumento.

**Actor Principal** Aplicación  
**Interesados** Operador del Sistema  
**Precondiciones** No tiene.  
**Condiciones de éxito** No tiene.

**Escenario de éxito**

<b>Acción o Intención del Actor</b>	<b>Respuesta del Sistema</b>
<b>1.</b> Envía dato para verificación de conexión.	<b>2.</b> Recibe el dato y envía dato verificando la conexión.
<b>3.</b> Recibe dato de verificación	
<b>4.</b> Envía dato solicitando verificar estado del instrumento.	<b>5.</b> Recibe dato y verifica el estado del equipo.
	<b>6.</b> Envía dato con verificación de estado.
<b>7.</b> Recibe dato de verificación y Cierra la conexión.	<b>8.</b> Cierra la conexión.

**Flujos**

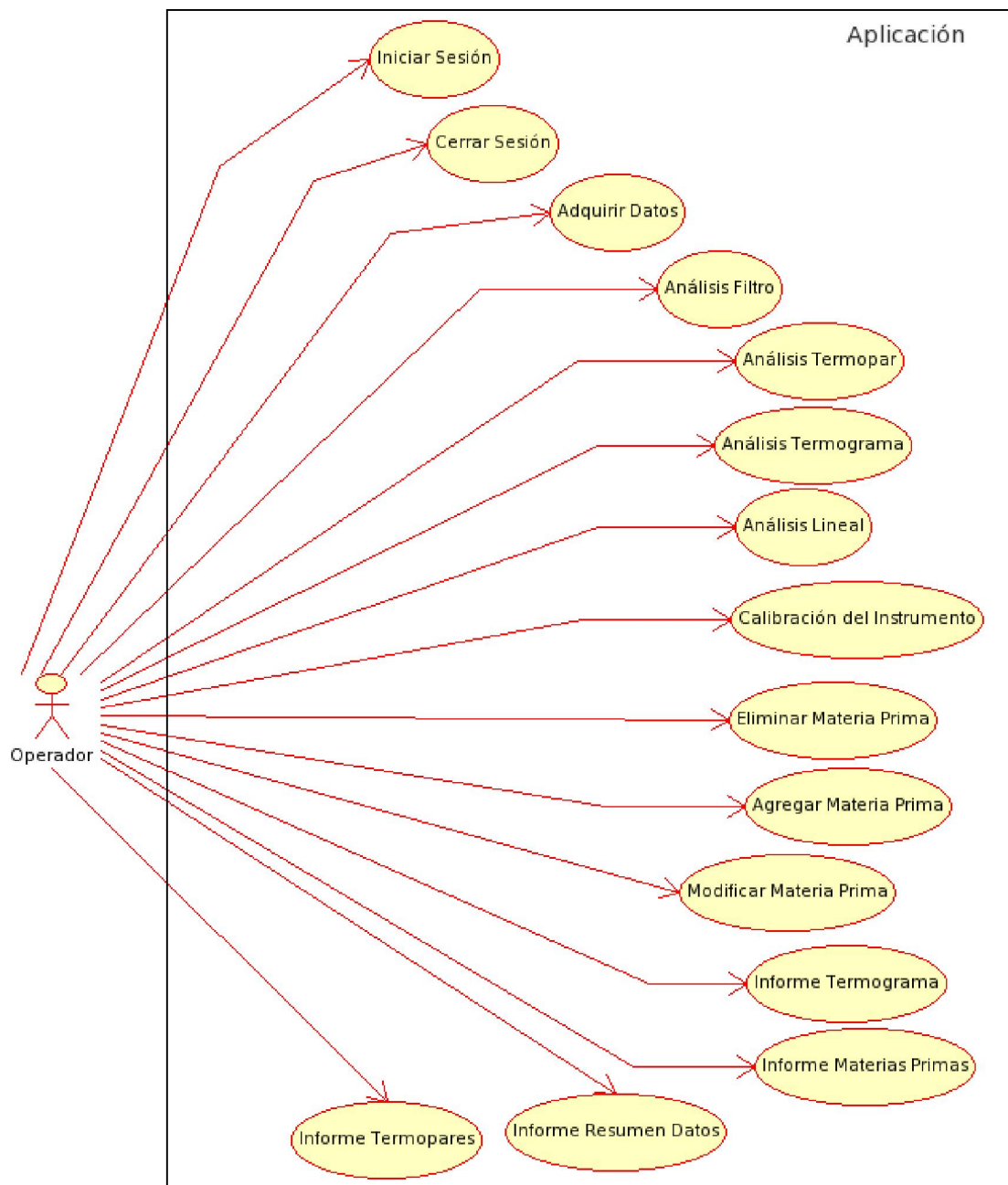
**Alternativos**

- A1 Error en el envío para verificar la conexión. Despliega mensaje de error y cancela la operación.
- A2 Error en el estado del instrumento. Despliega mensaje de error y cancela la operación.

**3.2.2.3 Diagrama Casos de Uso Operador – Aplicación**

Los diagramas representan la interacción entre los casos de uso descritos en las fichas. Quién depende de quién, quien generaliza a quien, como se **relacionan** los actores e interesados, etc.





\*El recuadro es el límite del sistema, dentro del recuadro está lo que hace el sistema

Figura 3.1 Diagrama de casos de uso operador-aplicación

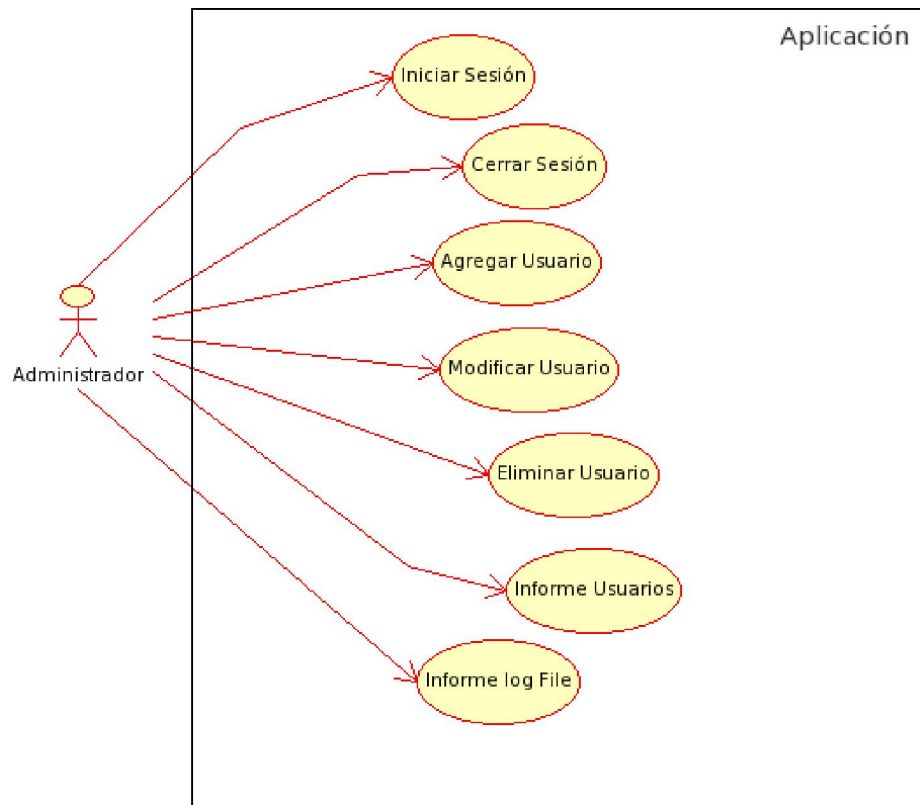


Figura 3.2 Diagrama Casos de Uso Administrador – Aplicación

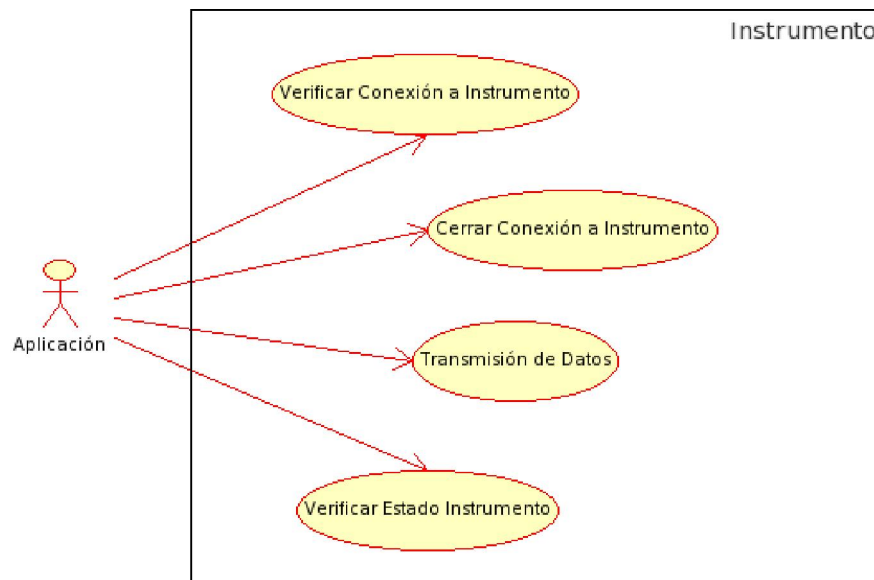


Figura 3.3 Diagrama Casos de Uso Aplicación-Instrumento

### 3.3 Diagramas de Secuencia (de mensajes)

Muestra los mensajes que hay entre las instancias de clases ( objetos ), actores, etc. Los con cuadros son sincronos y los otros son asincronos.

De estos diagramas se separan los métodos de las clases pero no como deben hacerlo, para eso están los diagramas de colaboración los que dan una idea de como debe ser la implementación o como debe funcionar cada mensaje.

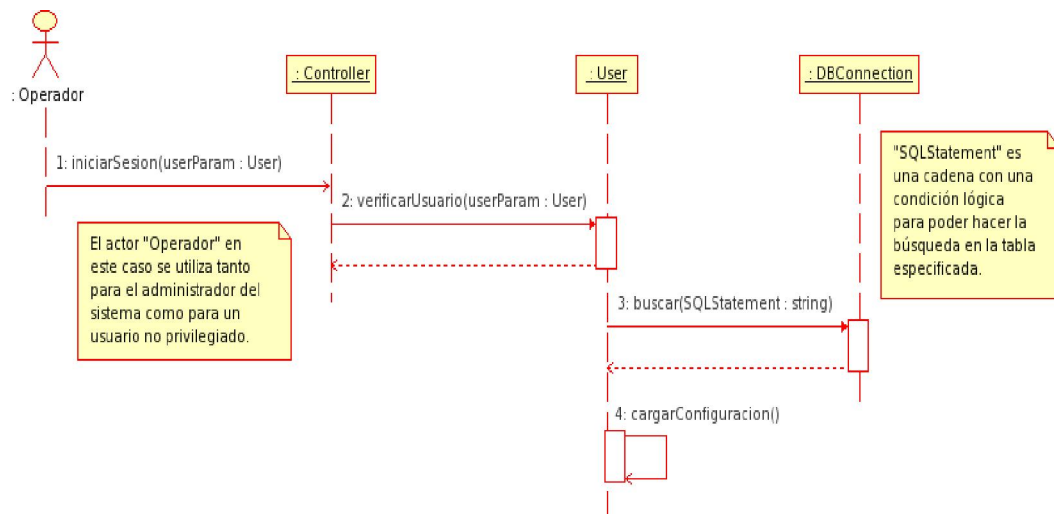


Figura 3.4 Diagrama de secuencia para el inicio de sesión

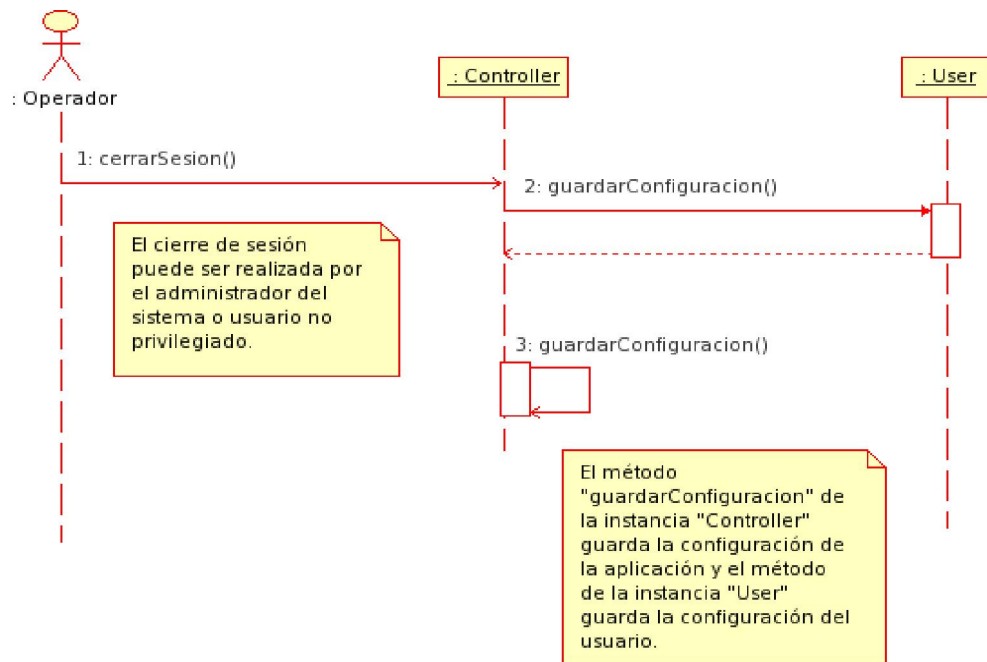


Figura 3.5 diagrama de secuencia para el cierre de sesión.

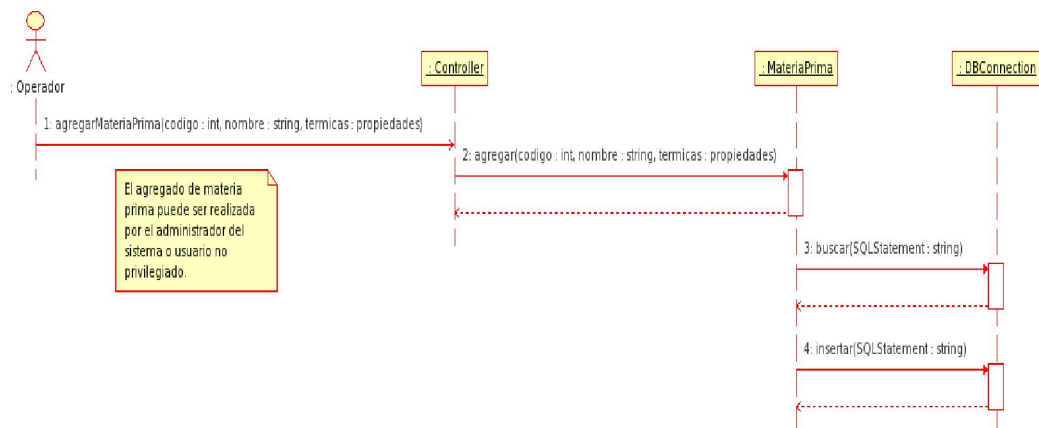


Figura 3.6 Diagrama de secuencia para agregar materias primas

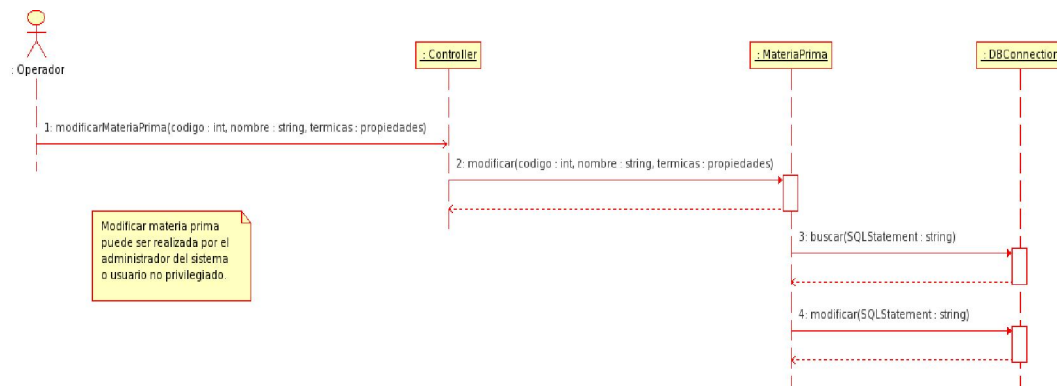


Figura 3.7 Diagrama de secuencia para eliminar materias primas

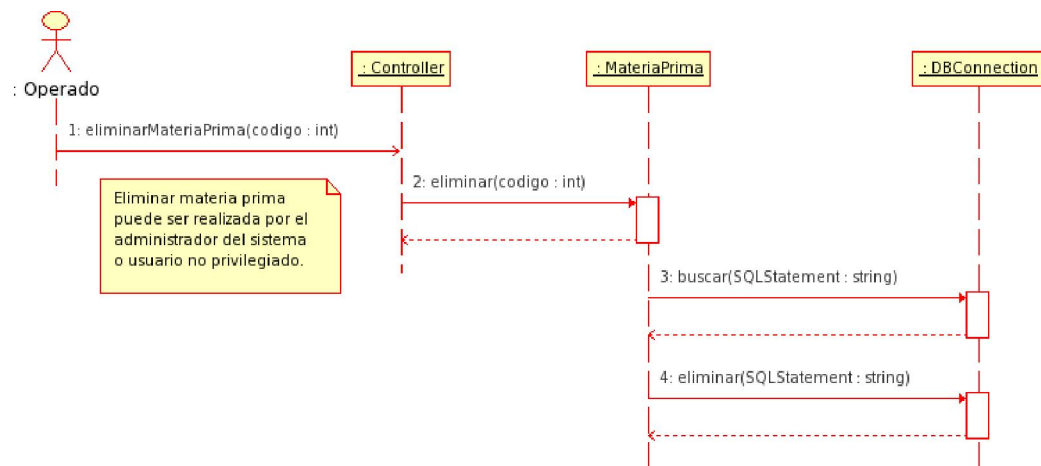


Figura 3.8 Diagrama de secuencia para modificar materias primas

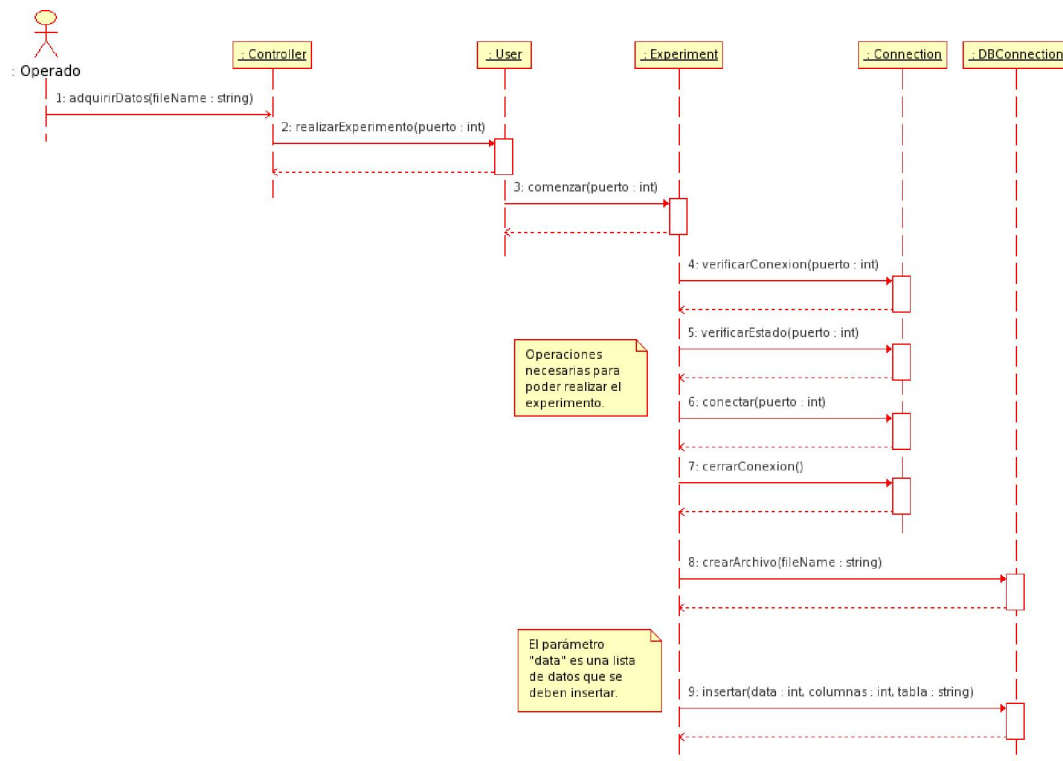


Figura 3.9 Diagrama de secuencia para Adquirir datos

La letra “**X**” al final de la instancia de *iConnection* significa que después de terminar la operación de adquisición de datos y cerrar la conexión la instancia de la clase *iConnection* se termina y se libera la memoria. *iConnection* es un TDA ( Tipo de dato abstracto ) temporal.

*La clase experiment contiene los datos del experimento, además de las operaciones para la linealización y la regresión polinomial. En resumen, tiene todos los métodos para hacer las operaciones sobre los datos. Experiment no siempre va adquirir los datos, también puede obtenerlos a partir de un archivo ya creado y si lo hace de esa manera entonces no se pueden adquirir datos a partir de esa instancia de la clase por lo que queda bloqueado.*

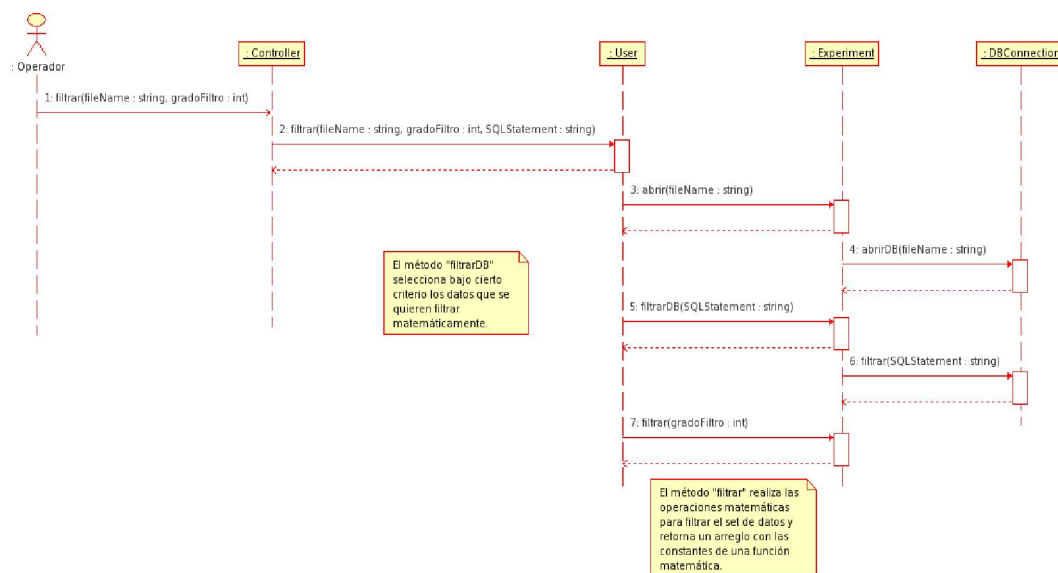


Figura 3.10 Diagrama de secuencia para Análisis de Filtro

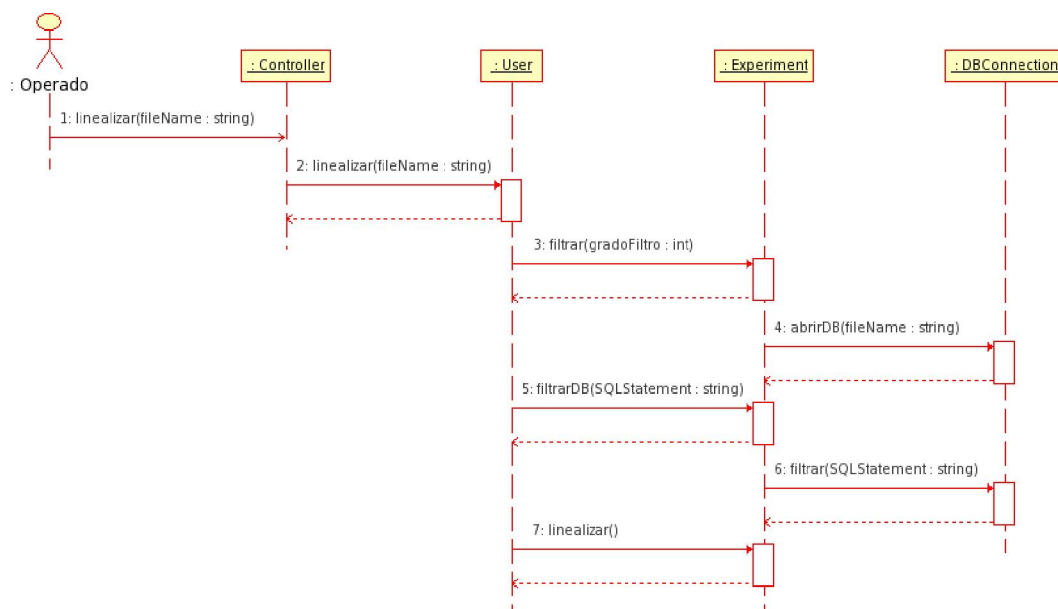


Figura 3.11 Diagrama de secuencia para Análisis Lineal

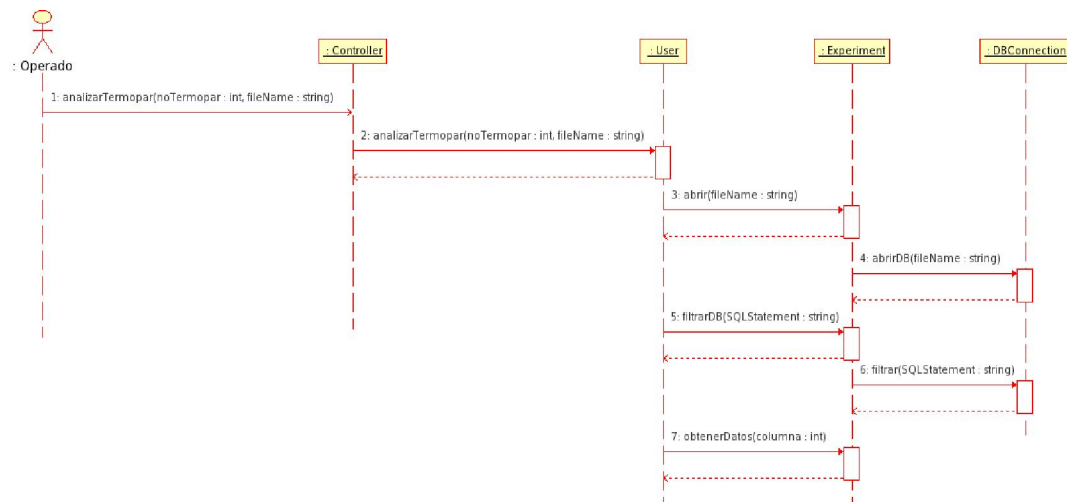


Figura 3.12 Diagrama de secuencia para Análisis de Termopar

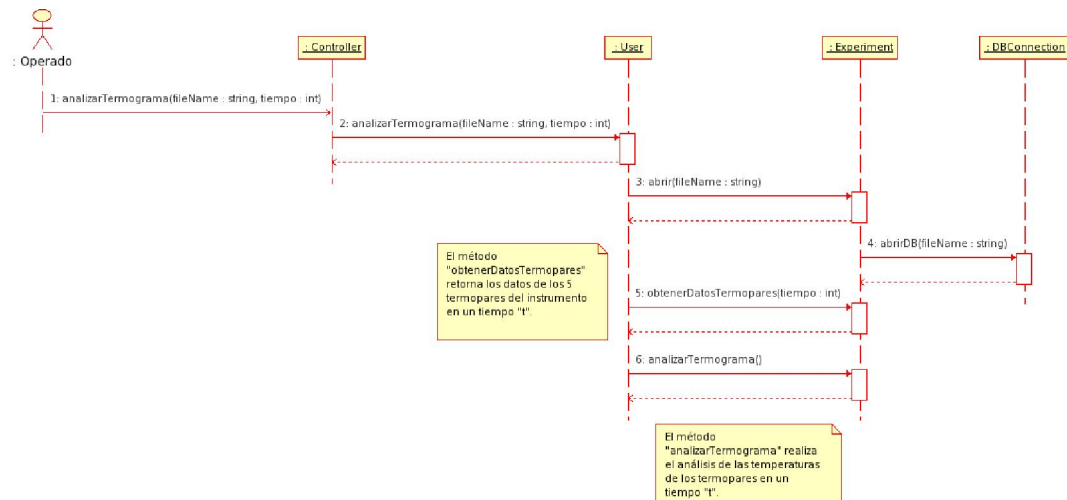


Figura 3.13 Diagrama de secuencia para Análisis de Termograma



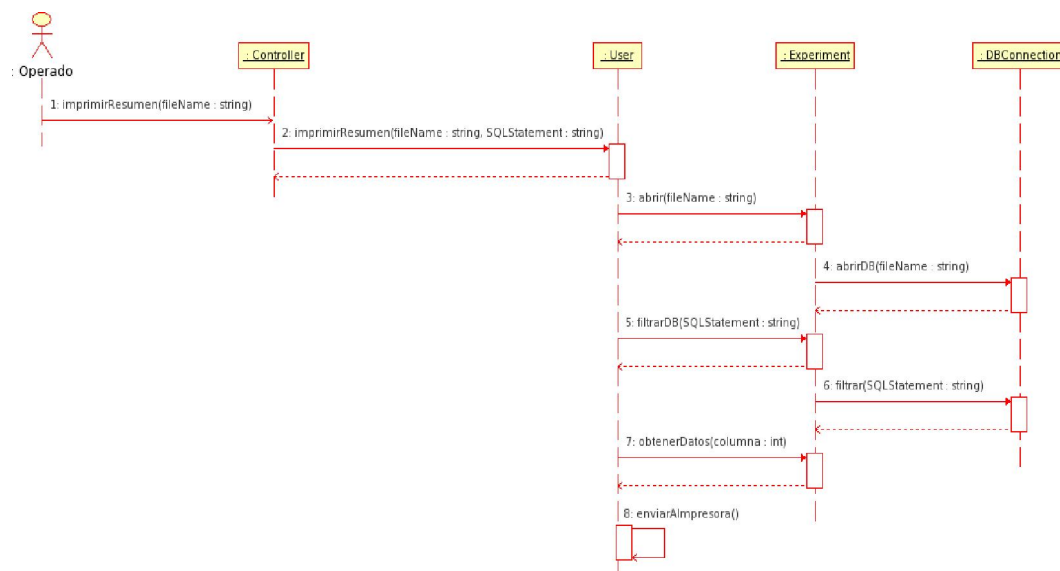


Figura 3.14 Diagrama de secuencia para Imprimir Resumen

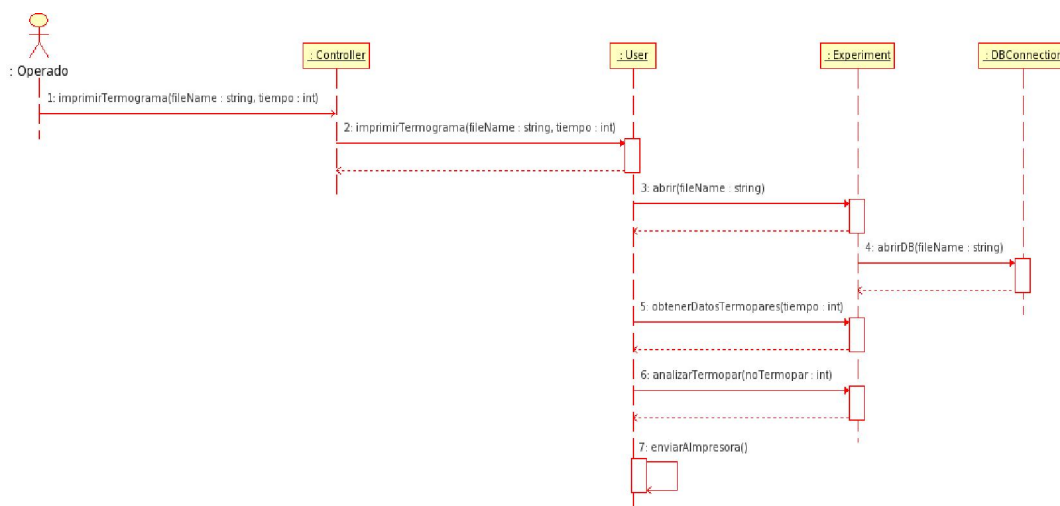


Figura 3.15 Diagrama de secuencia para Imprimir Termograma (gráfico)

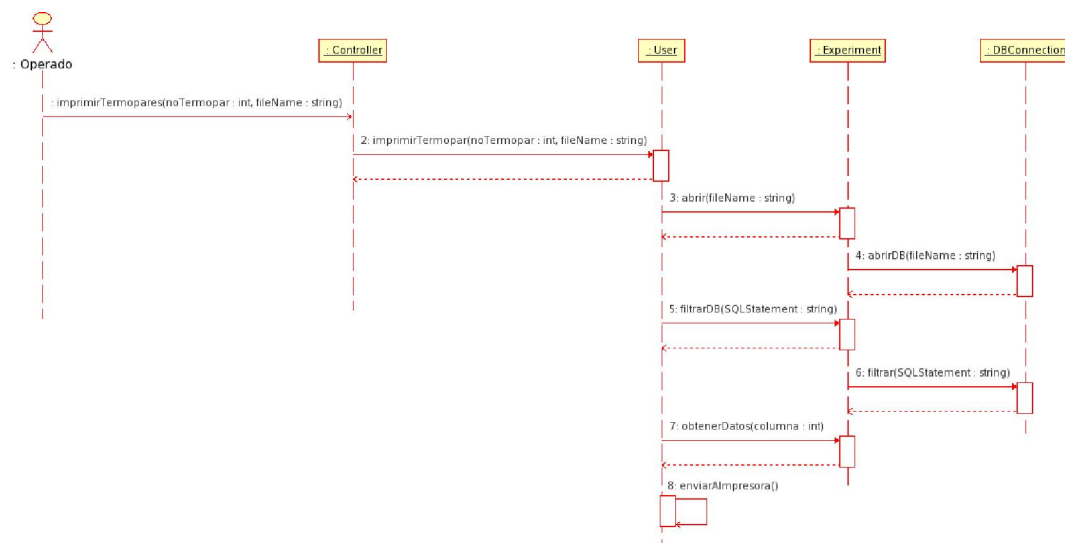


Figura 3.16 Diagrama de secuencia para Imprimir Termopar (graficos)

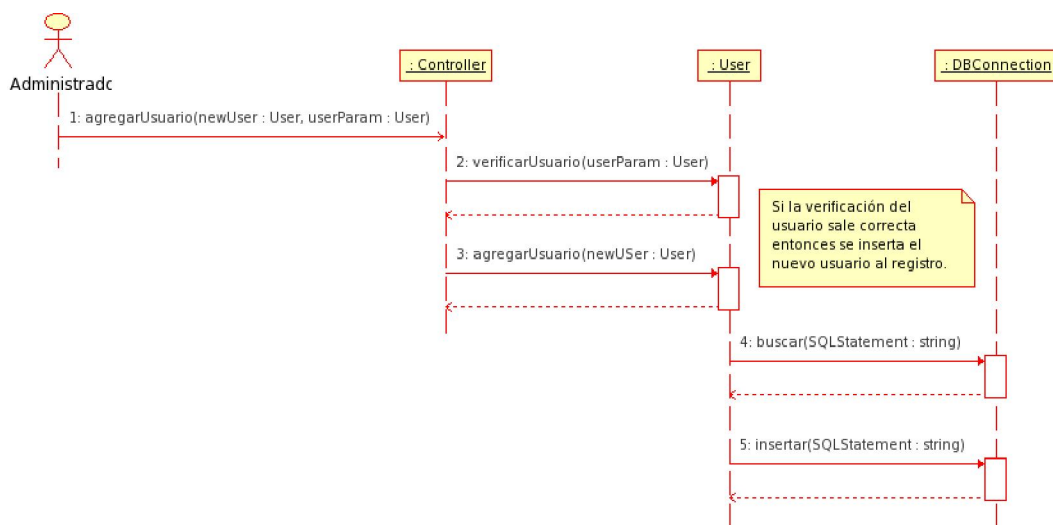


Figura 3.17 Diagrama de secuencia para agregar un usuario (Administrador)

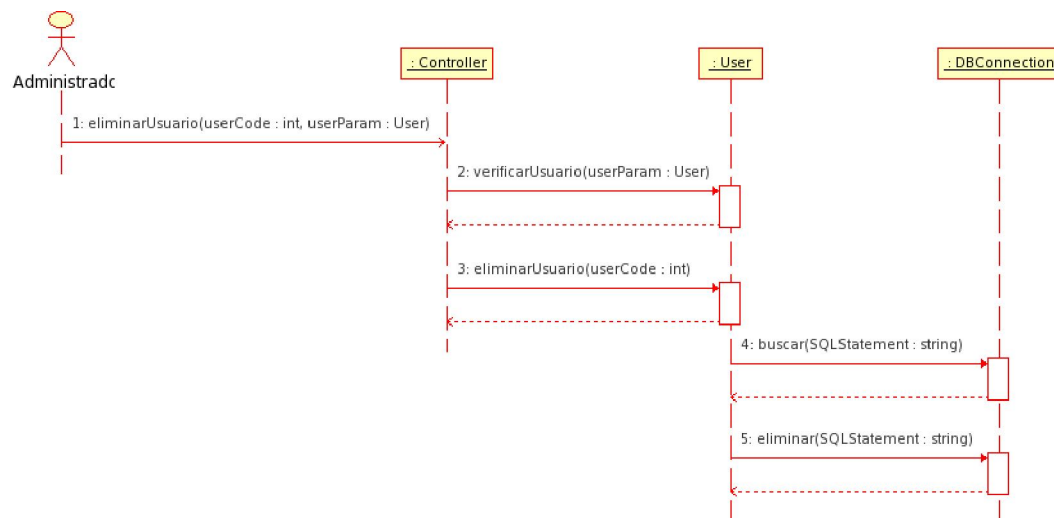


Figura 3.18 Diagrama de secuencia para eliminar un usuario (Administrador)

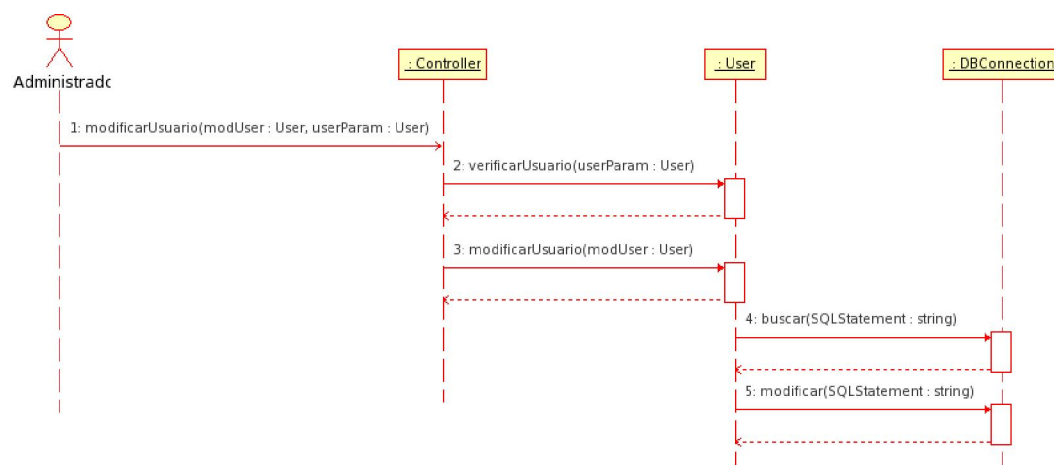


Figura 3.19 Diagrama de secuencia para modificar un usuario (Administrador)

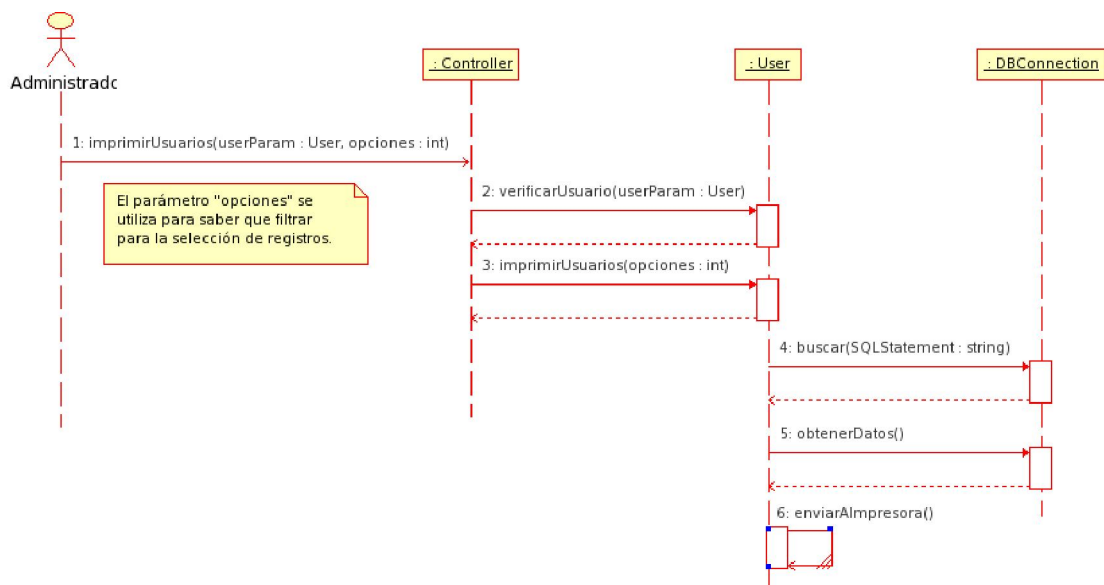


Figura 3.20 Diagrama de secuencia para informe de usuarios (Administrador)

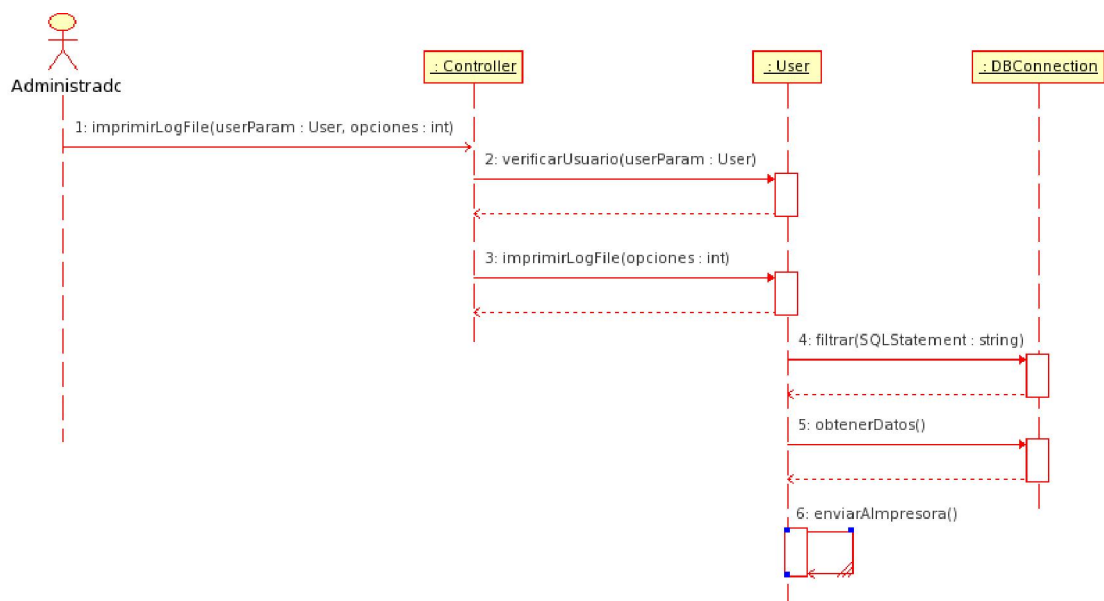


Figura 3.21 Diagrama de secuencia para informe de log file (Administrador)

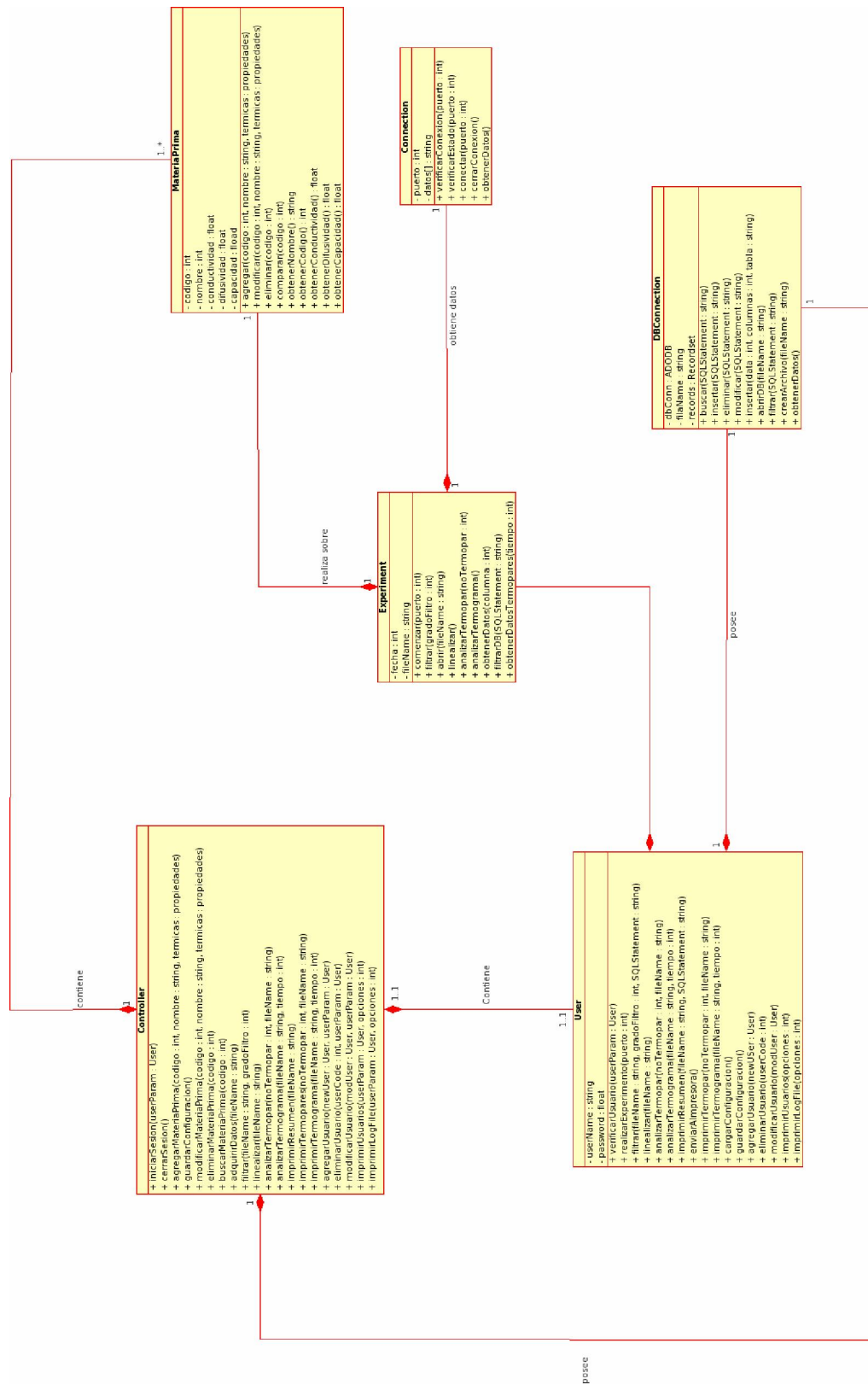


Figura 3.22 Diagrama de Clases

### **3.4 IMPLEMENTACIÓN**

Para la implementación de la aplicación final se utilizó Visual Basic 6.0® en plataforma de 32 bit, la aplicación cuenta con dos partes fundamentales: la primera adquisición en tiempo real y la segunda análisis de los datos obtenidos, (mayor detalle anexo 2).

**La aplicación grafica y los detalles de funcionamiento se explican en detalle en el capítulo siguiente de ensayos.**

## ***Capítulo 4***

### **Pruebas y ensayos**

Para la realización de las pruebas fueron seleccionadas algunas materias primas de carácter semisolido como pulpas de frutas y hortalizas, harinas, derivados y pures; los criterios de inclusión fueron:

- Composición semi solido.
- Propiedades físicas conocidas o fácilmente obtenibles.
- Propiedades Térmicas Conocidas.

Con estos datos conocidos se pudo comparar los valores logrados en el sistema que es el objetivo de este trabajo.

A continuación se describen los pasos a seguir para la obtención de resultados a traves del prototipo:

#### 1.- Ingreso de datos.

The screenshot shows a software window titled "Configuración para Adquisición de Datos". It contains several sections for data entry:

- Información del Operador:** Fields for "Operador" (containing "Administrador"), "Contraseña" (containing "ADMINISTRADOR"), and "Tipo Materia Prima" (containing "Pure de Manzanas").
- Información de Trabajo:** Fields for "Humedad de la Muestra (Schölin)" (containing "78"), "Voltaje de trabajo (V)" (containing "5" with a slider), and "Tiempo de Operación (seg)" (containing "120").
- Almacenamiento de Datos:** A field for "Archivo" (containing "Pure de Manzanas") and an "Examinar" button.
- Propiedades Buscadas:** Three checkboxes: "Conductividad" (checked), "Difusividad" (checked), and "Calor Especifico" (checked).

Figura 4.1 Ingresos de datos.

2.- Una vez ingresados los datos el usuario tiene dos opciones, calibrar las termocupas con agua a 0°C y a ebullición (aproximadamente 100°C) como límites en la escala y luego se realiza una medición a temperatura ambiente, logrando una calibración en los 5 termopares; si ya estan calibrados los termopares solo debe dar inicio a la captura de datos los cuales se ven así:

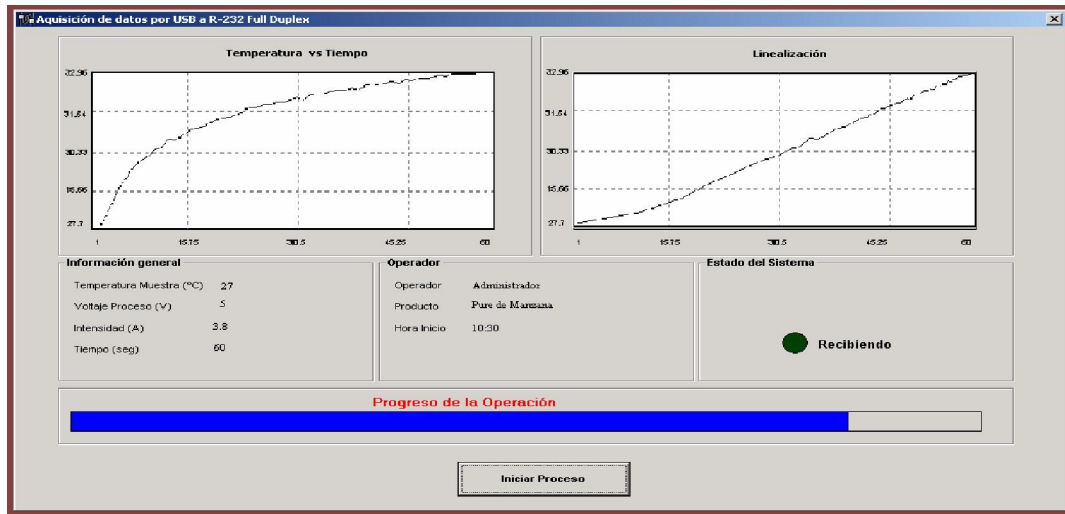


Figura 4.2 Captura de datos.

3.- Analisis de datos: En este punto se puede analizar cualquier set de datos obtenidos con el sistema, dependiendo de la selección de propiedades (conductividad, difusividad, calor específico), será el informe presentado; así como el análisis térmico que solo puede verse si se ha seleccionado todas las propiedades.

En el análisis el usuario tiene la posibilidad de escoger el tipo de filtro de la señal por medio del grado del polinomio, dejando así abierta la posibilidad de generar modelos para futuras pruebas o ensayos, la interfase de trabajo por termopar se muestra a continuación en la figura 4.3.



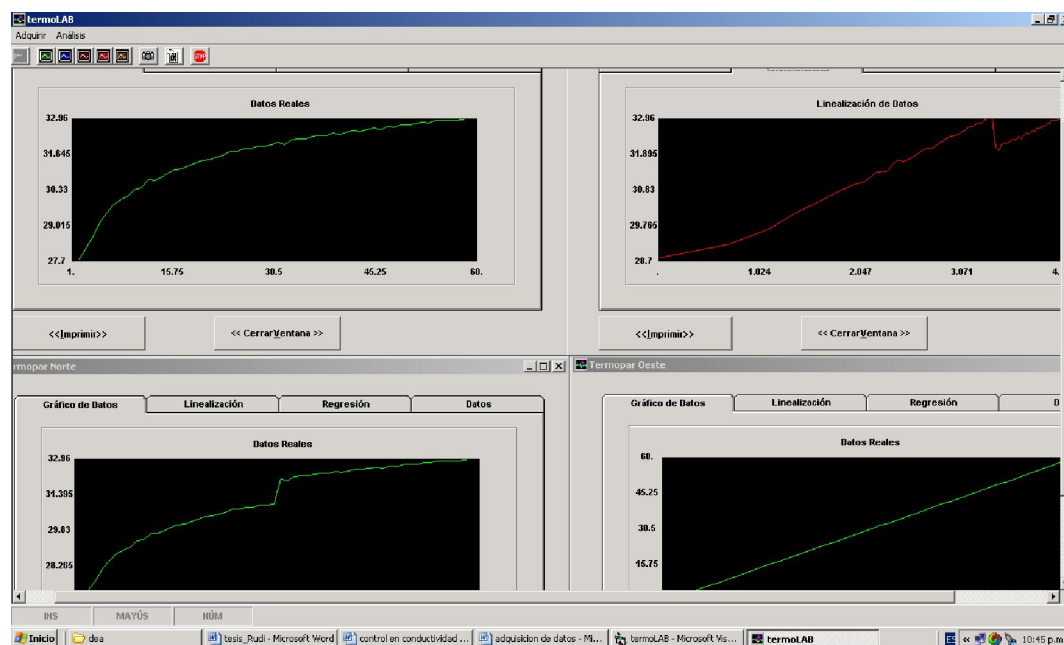


Figura 4.3 Interfase de trabajo.

4.- Después de analizar los datos, el sistema presenta la opción de imprimir los reportes por termopares como se muestra en la figura 4.4a y 4.4b o si lo prefiere el usuario puede ver un solo resumen de temperaturas de varios folios como en la figura 4.5; los reportes impresos son obcionales es decir se pueden ver y/o imprimir deacuerdo a la necesidad.

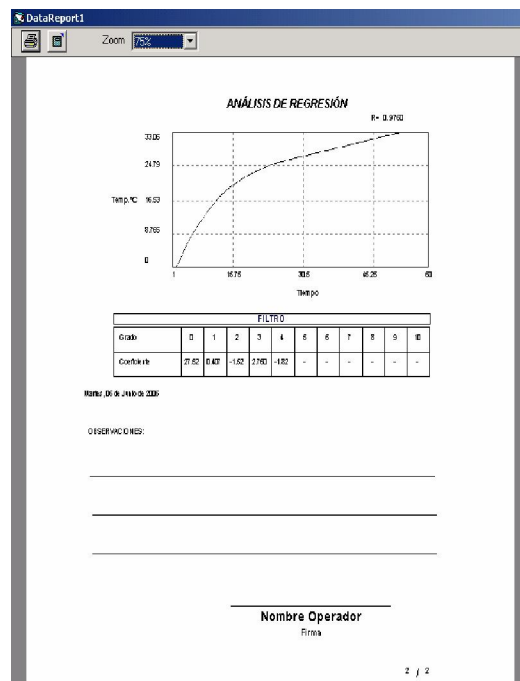
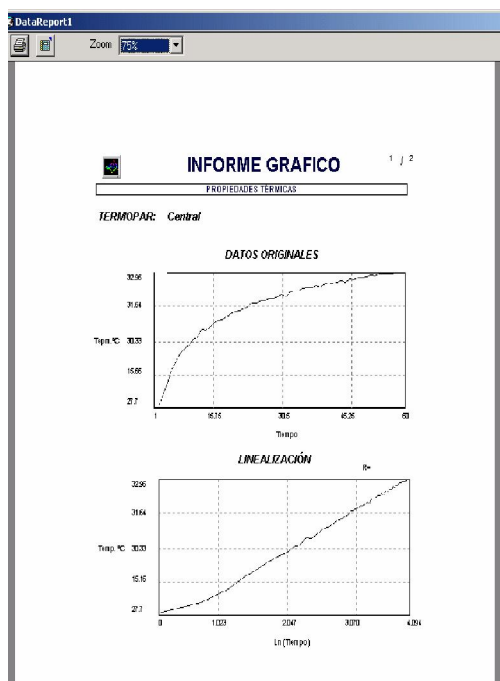


Figura 4.4 A Informe gráfico

Figura 4.4 B Analisis de Regresión

**Resumen Termopares**

Zoom: 100%

**RESUMEN TERMOPARES**

PROPIEDADES TÉRMICAS

Conductividad Media	0.427	Tempo Operación (s)	120
Difusividad Media	$1.30 \times 10^{-7}$	Humedad (Hrs)	88
Calor Especifico Medio	3.724	Temperatura (°C)	26

SET DE DATOS

TERMOPARES (°C)

Time (seg)	Central	Norte	Sur	Este	Oeste
1	27.7	26.7	26.7	0	1
2	28.1	27.1	28.1	0.683147180	2
3	28.56	27.56	29.55	1.036612268	3
4	29.09	28.09	30.09	1.386294361	4
5	29.38	28.38	30.38	1.609437912	5
6	29.59	28.59	30.59	1.791759469	6
7	29.89	28.89	30.89	1.945810149	7
8	30	29	31	2.079441541	8
9	30.27	29.27	31.27	2.197224577	9
10	30.31	29.31	31.31	2.302586062	10
11	30.63	29.63	31.63	2.397695272	11
12	30.81	29.81	31.81	2.484006649	12
13	30.73	29.73	31.73	2.564946367	13
14	30.86	29.86	31.86	2.639057329	14
15	31	30	32	2.708050201	15
16	31.02	30.02	32.02	2.772586722	16
17	31.15	30.15	32.15	2.833213344	17
18	31.26	30.26	32.26	2.890371757	18
19	31.35	30.35	32.35	2.944436979	19

Figura 4.5 Resumen de Propiedades Térmicas

5.-Si los datos almacenados poseen los 5 termopares el usuario tiene la posibilidad de generar un termograma en cualquier tiempo del proceso; ésta es una reconstrucción por medio de una simulación de distribución térmica bajo el principio de conducción multidimensional.

Al seleccionar un tiempo, el programa conoce las cinco temperaturas y con esa información genera 8211 volúmenes finitos los cuales responden de manera aproximada la distribución de temperaturas, notando una diferencia entre los resultados obtenidos bajo éstos principios, con los modelos teóricos de conducción radial como se ve en los termogramas a continuación:

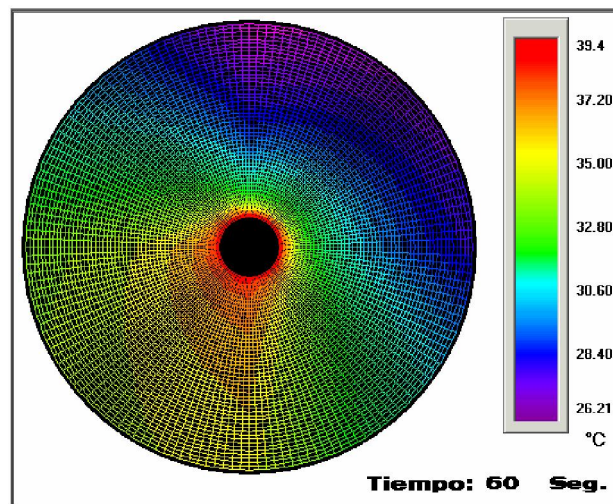


Figura 4.6 Termograma de pure de manzana a 60 segundos.

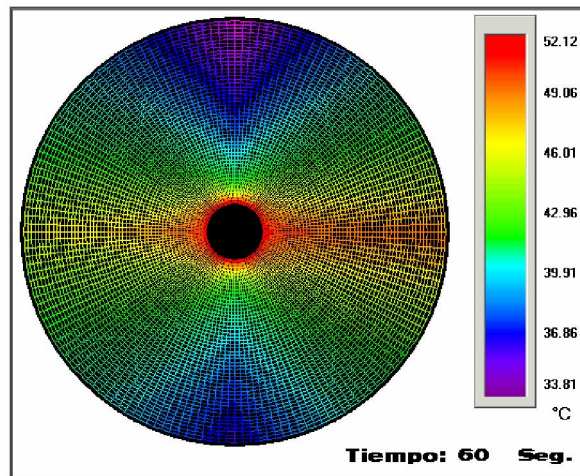


Figura 4.7 Termograma Pure de Pera a 60 segundos.

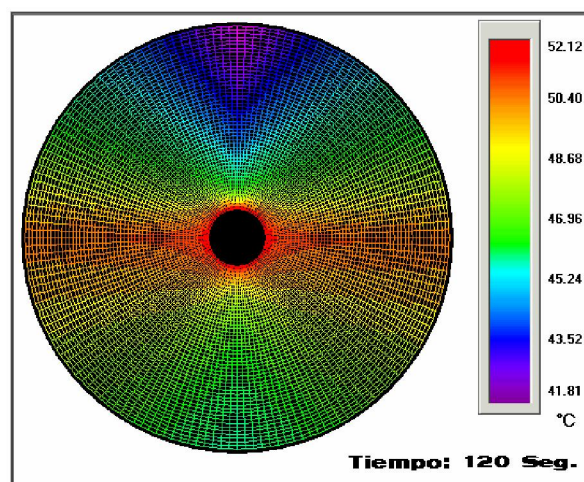


Figura 4.8 Termograma Pure de Pera a 120 segundos.

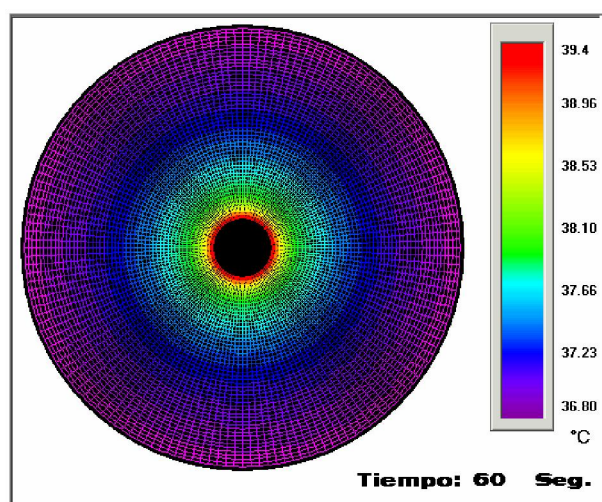


Figura 4.9 Termograma teórico

Si comparamos las distribuciones de temperatura (termogramas) con su microestructura podemos justificar el comportamiento de distribución caótica de los materiales vegetales, como se puede apreciar en las siguientes microfotografías.

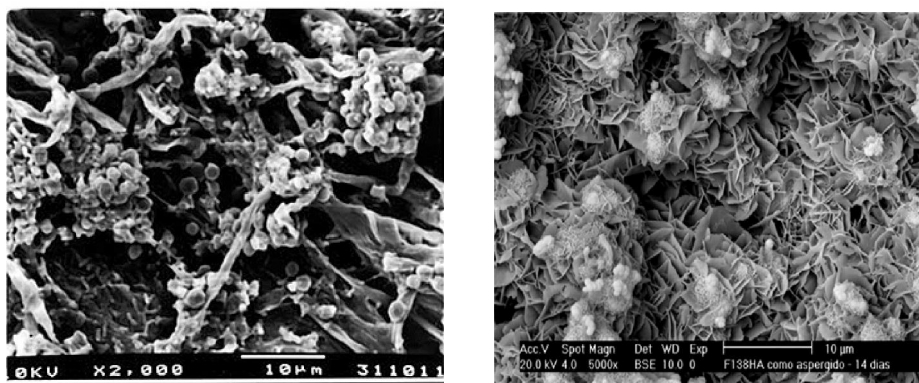


Figura 4.10 A) Microestructura de harina de trigo. B) Microestructuctura de lechuga.



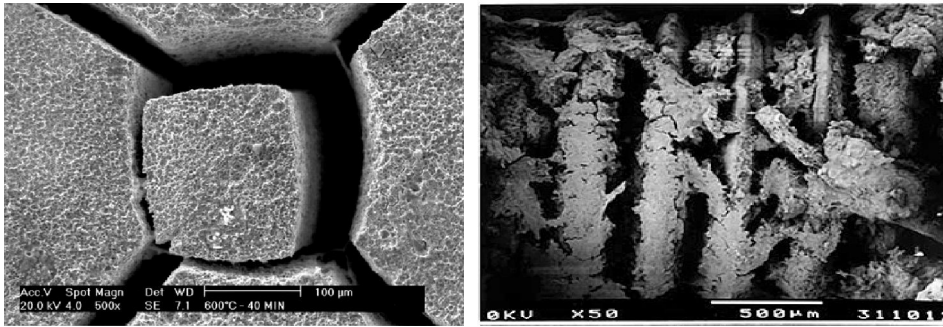


Figura 4.11 A) Microestructura de durazno B) Microestructura de galleta

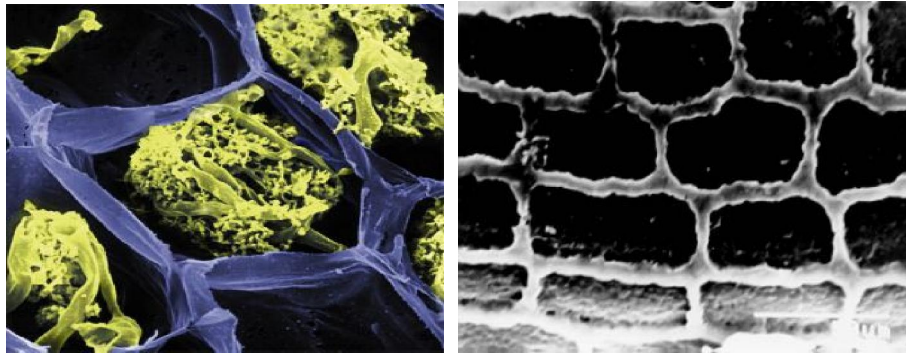


Figura 4.12 Red proteica vegetal nucleada

Al comparar las redes proteicas (figura 4.12) se puede apreciar que algunas poseen un núcleo y otras se encuentran vacías, en cada caso no existe una distribución homogénea de estos y por tanto no permite realizar un modelo general para la transferencia térmica. Si analizamos el fenómeno de conducción de calor en las microestructuras podemos decir:

El transporte de calor por la matriz proteica es por conducción en su mayoría, el fenómeno convectivo es despreciable ya que las dimensiones son tan pequeñas que la cantidad de aire no presenta resistencia, pero los centros nucleados también conducen calor y en la mayoría de los casos concentran el calor como pequeñas fuentes, las cuales

son absorbidas por efecto de radiación haciendo un frente dinámico de transporte de calor ,similar a la ecuación de Koch (vortices de flujo metereológico) [23] esquematizado en la figura 4.13, fenómeno que ocurre en cada poro o celda de la microestructura vegetal.

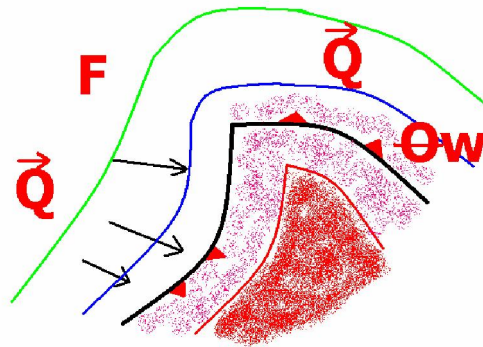


Figura 4.13 Función del vértice de Koch

Por todo lo antes expuesto los termogramas obtenidos representan una mejor aproximación, que la teoría de conducción radial y se puede extrapolar el comportamiento de los fractales meterológicos a la conducción de calor en alimentos semi solidos.

En el siguiente cuadro se muestra la comparación de manera estadística, los resultados obtenidos por el sistema presentan una menor desviación comparados con los valores obtenidos por técnicas precedentes, las misma que muestra una mayor desviación y menor replicabilidad.

**Tabla 4.1 Comparación de resultados entre el sistema y técnicas presedentes**

	<b>K</b> <b>(W/mK)</b> <b>Sistema</b>	<b>(m<sup>2</sup>/s)</b> <b>Sistema</b>	<b>Cp</b> <b>(Kj/kg K)</b> <b>Sistema</b>	<b>Desv</b>	<b>K</b> <b>(W/mK)</b> <b>Sweat</b>	<b>(m<sup>2</sup>/s)</b> <b>Adiabatico</b>	<b>Cp</b> <b>(Kj/kg K)</b> <b>Calorimétrico</b>	<b>Desv</b>
<b>Pure de manzana</b>	0.3472	0.67x10 <sup>-7</sup>	2.532	0.031	0.422	0.77x10 <sup>-7</sup>	3.724	0.085
<b>Harina</b>	0.1266	0.40x10 <sup>-7</sup>	2.359	0.034	0.541	1.01x10 <sup>-7</sup>	1.813	0.067
<b>Pulpa tomate</b>	0.6018	0.89x10 <sup>-7</sup>	2.788	0.033	0.605	1.4x10 <sup>-7</sup>	3.933	0.078
<b>Pure de pera</b>	0.4439	0.67x10 <sup>-7</sup>	2.214	0.035	0.595	1.11x10 <sup>-7</sup>	3.347	0.056
<b>Harina de Piñón</b>	0.1266	0.44x10 <sup>-7</sup>	2.383	0.032	0.287	1.21x10 <sup>-7</sup>	2.62	0.077

Valores promedio con n=3

Como se puede apreciar la repetición de información es superior al 96.5% con el sistema de captura de datos, comparado con el 92.5% que se obtiene en técnicas separadas, si bien la diferencia no es significativa ya que los resultados se encuentran en rangos aceptables en otras metodologías, no es así el tiempo de captura y procesamiento de la información así como la cantidad de muestra, ya que el tiempo de recolección de datos por el sistema no supera los 5 minutos incluido el análisis, en comparación al tiempo que otras técnicas necesitan por separado para capturar y analizar (sobre 4 horas por método), en cuanto a la cantidad de muestra el sistema requiere un máximo de 150 gramos de materia prima para las tres determinaciones versus 1500 gramos para realizar las determinaciones por separado.

Como se puede apreciar en el cuadro anterior el sistema propuesto es más estable y económico que las determinaciones por separado.



## ***Capítulo 5***

### **Conclusiones**

- ✓ El sistema de adquisición de temperatura permitió capturar las variaciones al interior de un alimento demostrando el anisotropismo de distribución caótica de los materiales vegetales analizados.
- ✓ El sistema de transformación de USB a RS232 Full Duplex, no presenta cambios o pérdida de señal y la tasa de transferencia llega a los 19200 baudios sin pérdida de información, aun que la versión de USB es la 1.1 es soportada en cualquier equipo con host USB, además la frecuencia de toma de datos superior a 20 datos por segundo es soportada por la velocidad del protocolo RS232, permitiendo que el BUS trabaje holgadamente.
- ✓ Ya que la velocidad de transferencia de datos entre los sensores y el host del ordenador trabaja sobre 19200 baudios, se deja la posibilidad libre para la modificación del sistema para combinar entre los protocolo USB y RS232 con el host del ordenador el cual posee la aplicación con el usuario.
- ✓ Al analizar los datos obtenidos con una prueba experimental se puede mencionar que el grado de certeza es alto comparado con los sistema primarios que dieron origen a este trabajo (sobre 90%).
- ✓ El logro más importante de este trabajo es que se pudo diseñar y construir un sistema de captura de datos para temperatura a través del protocolo USB de bajo

- coste y altamente confiable que permite disminuir el tiempo de muestreo y el error asociado al usuario, así como el análisis visual del fenómeno térmico.
- ✓ El análisis de la microestructura de los cuerpos vegetales permitió la incorporación del término de radiación a la ecuación general de calor, dando como resultado una imagen termográfica reconstruida por diferencia finita.
  - ✓ El comportamiento del termo grama reconstituido por diferencia finita es una mejor aproximación en comparación con el modelo teórico [5] utilizado por los autores clásicos de transferencia térmica.
  - ✓ El comportamiento de los resultados obtenidos son similares a los modelos de viento de la curva de Koch (teoría de los fractales)[23], pudiendo explicar el comportamiento de los mismos por esta teoría matemática.
  - ✓ Al analizar el fenómeno de transferencia térmica sin considerar una distribución estadística de tamaño de poros o distribución espacial de éstos, facilita la modelación al conocer las propiedades dinámicas de los materiales vegetales, evitando el estudio de singularidades presentes que son difíciles de predecir, arrojando datos erróneos como los modelos teóricos actuales.
  - ✓ Al comparar los resultados de las propiedades térmicas de los materiales vegetales tanto del sistema como teorías de modelación, se puede decir que los valores presentan una dispersión mucho menor si se obtienen por sistema propuesto que por cualquier otra técnica.
  - ✓ La determinación de la conductividad térmica por métodos transcientes presentan un error asociado a la distribución de las fibras del material, cualidad que

no se puede determinar por otra metodología que no sea la microscopia electrónica, encareciendo los resultados y aumentando el tiempo de prueba, lo que dificulta su aplicación en líneas de proceso con materias primas heterogéneas.

- ✓ El prototipo puede ser usado en líneas de proceso industrial y/o laboratorios de investigación o docencia, ya que permite una flexibilidad de trabajo y análisis de resultados.



## ***Capítulo 6***

### **Aportaciones**

Esta tesis Doctoral presenta algunos tipos de aportaciones que se pueden desglosar de la siguiente manera:

- Aportación Técnica
- Aportación Científica
- Aportación Práctica
- Aportación Matemática

**Aportación Técnica:** Al integrar en un trabajo los conocimientos de electricidad, electrónica, computación a la ingeniería de procesos de producción y manufacturas de alimentos; se logró un prototipo de adquisición de temperatura con tecnología de punta como es el USB, manteniendo los estándares internacionales de protocolos y encriptación permitiendo una versatilidad y fácil comunicación y reconocimiento entre el dispositivo y cualquier computador actual, el cual con un pequeño programa (driver) se controla la operación del mismo y permite al usuario cambiar entre rangos permitidos la configuración del equipo.

Todos los equipos de análisis de propiedades térmicas [16,17,24,29] utilizan la conducción radial del exterior al interior y no viceversa, al trabajar con la configuración radial de cinco sensores permite trabajar del interior al exterior, reduciendo el tiempo de exposición de la muestra.

Para la adquisición de temperaturas para la conductividad térmica, se utiliza la sonda de Sweat [27] la cual es la modificación de la sonda de Kotani[18], en la que ocupa una fuente calefactora y receptor unidos y no por separado; la modificación que se planteó fue unir cuatro sondas tipo Kotani y una Swaet, modificando una fuente calefactora y receptor en el centro y cuatro receptores equidistantes, logrando así una mejor aproximación al fenómeno de transferencia térmica además de permitir encontrar la difusividad y calor específico del material en un solo experimento.

**Aportación Científica:** En la actualidad las propiedades térmicas de los alimentos no se han considerado como propiedades replicables y peor aun no se han integrado en un solo equipo. Por otra parte la incorporación del término de microrradiación a la ecuación de calor para cuerpos porosos, permitió una mejor aproximación del fenómeno de transferencia térmica pudiendo entregar una distribución de temperaturas en forma de termograma.

Con esta herramienta el investigador puede correlacionar las fotografías electrónicas de las microestructuras con el termograma, facilitando el análisis matemático del fenómeno en estudio.

**Aportación Práctica:** Al incorporar las tres propiedades en una sola determinación el tiempo de análisis de la muestra así como su coste se ven drásticamente disminuidos.

Al incorporar procesos adicionales como: filtrado reconstructivo, termograma, visualización de datos, permiten a un estudiante o investigador profundizar en alguna o algunas experiencias prácticas que ayudarán a mejorar los procesos de manufactura de alimentos evitando el deterioro por una mala aplicación de tratamiento térmico

Con la disminución del tiempo de análisis térmico el ingeniero de producción podrá trabajar con datos reales de sus materias primas, y no de datos obtenidos por tablas que casi siempre, no corresponde a la materia prima utilizada para la manufactura, provocando desperfectos en la fabricación de comestibles.

Por otro lado el análisis de las propiedades al ser casi automático se minimizan los errores asociados al operador como: lecturas erróneas, truncado de decimales, perdida de información, al poseer un modulo de calibración evita además el error por desviación o conversión de señales, permitiendo la replicabilidad de los resultados.

Al ser un equipo de fácilmente transportable se puede usar como instrumento de laboratorio o instrumento de campo.

**Aportación Matemática:** El principio de trabajo es la conducción multidimensional de calor no estacionario, el cual plantea un sistema capacitivo resistivo en coordenadas cilíndricas, para ello es necesario utilizar las siguientes expresiones matemáticas:

Ítem	Coordenadas Cilíndricas	observaciones
Coordenadas	$r, \phi, z$	Radio, ángulo, espesor
Índices	$m, l, n$	
Elemento de control	$r_m, r, z$	Volumen finito
$R_{m+}$	$\frac{\Delta r}{(r_m + \Delta r/2) \Delta \phi \Delta z k}$	Resistencia radial positiva
$R_{m-}$	$\frac{\Delta r}{(r_m - \Delta r/2) \Delta \phi \Delta z k}$	Resistencia radial negativa
$R_{n+}$	$\frac{r_m \Delta \phi}{\Delta r \Delta z k}$	Resistencia angular positiva

Rn-	$\frac{rm\Delta\phi}{\Delta\lambda\Delta z h}$	Resistencia angular negativa
RI+	$\frac{\Delta z}{rm\Delta\phi\Delta r k}$	Resistencia de capa positiva
RI-	$\frac{\Delta z}{rm\Delta\phi\Delta r k}$	Resistencia de capa negativa

La ecuación de calor queda expresada como:

$$Cm \frac{T_m^{i+1} - T_m^i}{\Delta t} = \sum_n \frac{T_n^i - T_m^i}{R_{mn}^{cond}} + \sum_n \frac{T_n^i - T_m^i}{R_{mn}^{rad}} + \dot{Q}_v \Delta V_m$$

Donde

$$R_{mn}^{rad} = \frac{1}{A_n \Gamma_{mn} \sigma (T_n^2 + T_m^2)(T_n T_m)}$$

Como no existe otra fuente de generación de calor  $\dot{Q}''' = 0$ , por tanto la ecuación nodal para diferencia finita de los nodos interiores para el cálculo de la temperatura esta dada por:

$$T_m^{i+1} = \left( 1 - \frac{\Delta t}{Cm} \sum_n \frac{1}{R_{mn}} \right) T_m^i + \frac{\Delta t}{Cm} \sum_n \frac{T_n^i}{R_{mn}}$$

Valido para todo modelo computacional:



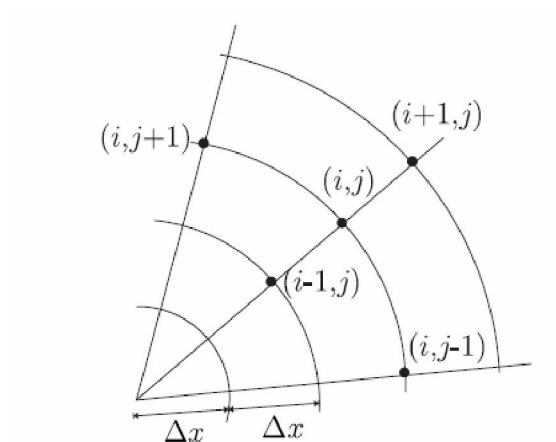


Figura 6.1 Malla de puntos en coordenadas polares

Siendo el nodo exterior  $\frac{1}{2}$  del interior por condiciones del experimento (sistema adiabático); para verificar la estabilidad del sistema por diferencias se utiliza la ecuación:

$$\Delta t \leq \left( \frac{Cm}{\sum_n \left( \frac{1}{R_{mn}} \right)} \right)_{\min}$$

Con ello se logra una distribución de 160 radios y 51 capas radiales con un total de 8211 volúmenes de control finito distribuidos como la siguiente figura.

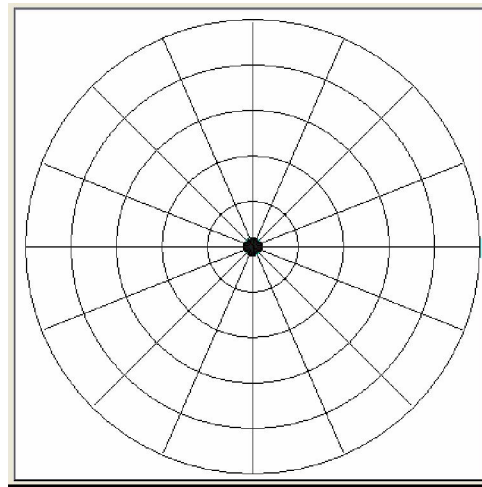


Figura 6.2 Distribución de nodos con volúmenes finitos.

## ***Glosario***

**ADB** (Apple Desktop Bus) Puerto serie de 4 hilos para enchufar el teclado en los Macs también conocido como 4-pin Mini Din o S-Video connector.

**Bit** (Binary digit) La mínima unidad de información de un computador, un 1 o un 0. Define una de las dos condiciones posibles: activado o desactivado.

**Firmware** Es un bloque de instrucciones de programa para propósitos específicos, grabado en una memoria tipo ROM, por ejemplo la BIOS del sistema.

**Funciones** (Functions) Dispositivo USB que proporciona funcionalidad al host como puede ser una conexión ISDN, un micrófono digital, altavoces o un ratón.

**Geoport** - Puerto de serie para los ordenadores Apple que proporciona un interfaz entre la línea telefónica y el ordenador.

**HID** (Human Interface Device) Dispositivos que son usados por el hombre para operar con un sistema informático. Algunos ejemplos pueden ser: teclados, ratones, trackballs, joysticks, guantes de datos, volantes y pedales. También son HIDs los lectores de códigos de barras, medidores de temperatura o voltímetros.

**HUB** (Concentrador) Dispositivos que expanden el número de puntos de conexión al bus mediante puertos adicionales.

**Hot Swappable** Capacidad de configuración que tienen algunos dispositivos de poder conectarse y ser reconocidos sin tener que apagar la máquina y reiniciarla, gracias entre otras cosas al bus sobre el cual se conectan (bus USB).

**IRPs** (I/O Request Packets) en español Petición de paquetes de E/S. Los detalles sobre los IRPs dependen del sistema operativo que se utilice. De modo que en este documento el término hace referencia a una petición identificable por el software cliente para mover datos entre el y un endpoint de un dispositivo en una dirección apropiada.

**ISDN** (Integrated Services Digital Network) en español RDSI. Red que facilita conexiones digitales extremo a extremo para proporcionar una amplia gama de servicios, tanto de voz como de otros tipos, y a la que los usuarios acceden a través de un conjunto de interfaces normalizados.

**PBX** (Private Branch eXchange) Red de teléfono privada usada dentro de una empresa cuyos usuarios pueden compartir un cierto número de líneas externas para hacer llamadas al exterior de la PBX.

**PDA (Personal Digital Assistant)** Ordenador de mano diseñado originalmente como agenda electrónica y que se hizo mucho mas versátil con el paso del tiempo. Tiene la capacidad de transferir datos con un portátil o un ordenador de sobremesa.

**PnP o Plug & Play** Sistema de reconocimiento por el que el sistema operativo detecta un nuevo hardware. Tras la detección, el sistema busca el software que lo gestione, y que suele estar en los discos que el fabricante ofrece con el dispositivos

**POTS (plain old telephone service)** Servicio telefónico estándar, también llamada red telefónica pública conmutada o public switched telephone network (PSTN).

**RS-232** Estándar de comunicación serie utilizado desde los primeros PCs hasta la actualidad

**SE0 (Single-ended 0)** Hace referencia a un estado en el que se pueden encontrar las líneas D+ y D- del cable USB cuando nos encontramos en el caso de velocidad media o baja. En concreto es el estado en el que  $D+ \text{ y } D- < VOL(max)$ .

**Unidad ZIP** Unidad de almacenamiento magnético similar a una disquetera cuyos discos tienen una capacidad de 100 o 250 MB según el tipo de unidad.

**USB-IF** USB Implementers Forum, Corporación fundada por un grupo de empresas que se unieron para el desarrollo de la especificación del USB.

**WAP (Wireless Application Protocol).** Estándar abierto internacional para aplicaciones que utilizan las comunicaciones inalámbricas, p.ej. acceso a Internet desde un teléfono móvil.

## **Bibliografía**

- 1) ANGULO, J.M., M. ANGULO., E. MARTIN., Microcontroladores PIC, La solución en un solo chip, Editorial Thomson Paraninfo S.A. 2001
- 2) ANGULO, J.M., M. ANGULO., Microcontroladores PIC, Diseño de aplicaciones, McGraw Hill/ INTERAMERICANA DE ESPAÑA, S.A.1997
- 3) BENNET, A. H., Chance, W.G. and Cubbedge, R. H., Estimating thermal conductivity of fruit and vegetable components\_ the Fitch method, ASHRAE J., 4(9), 80, 1962.
- 4) BIRCHENALL."Los Mecanismos de Difusión en el Estado Sólido". Metalurgical Reviews 3 : 235 - 277 (1958).
- 5) BRODKEY, R.S. y Hershey, H.C.: Transport Phenomena. A Unified Approach, McGraw 1988
- 6) CRANK, J. "The Mathematics of Diffusion." Oxford University Press.1964.
- 7) EMBEDDED Control Handbook, Microchip PIC 16/17 microcontroller data Book, Microchip.

- 8) FITCH, A.L., A new thermal conductivity apparatus, Am. Phys. Teacher, 3, 135,1935.
- 9) GASKELL, D.R.: An Introduction to Transport Phenomena in Materials Engineering; McMillan 1992.
- 10) GONZÁLEZ, J.A., Introducción a los microcontroladores, McGraw Hill 1997.
- 11) HID Usage Tables (located at [www.usb.org](http://www.usb.org))
- 12) HASS, E. and Felsenstein, G., Methods used to the properties of fruits and vegetables, Special publication No 103, Division of Scientific Publications, The Volcani Center, Bet Dagan, Israel, 1978.
- 13) HOOPER, F.C. and Lepper F.R., Transient heat flow apparatus for the determination of thermal conductivities, Trans. ASHRAE,56, 309, 1950.
- 14) LARMAN, Craing, Applyng UML and Patterns, An Introduction to Object-Oriented Analysis and Design and the Unified Process, Prentice Hall, Second Edition , 2003.
- 15) INCROPERA, F. P. y. DeWitt D. P: Fundamentals of Heat and Mass Transfer. Wiley, 1996(Traducido al español como Fundamentos de Transferencia de Calor por Prentice Hall de México en 1999).

- 16) INSTITUTO del Frío Argentina, Equipo de medida de coeficiente de conductividad, calor específico y entalpía, ([http://www.geoscopio.com/est/gmms/afre2/producto\\_2751.htm](http://www.geoscopio.com/est/gmms/afre2/producto_2751.htm)), [consulta 4/01/2005].
- 17) INSTITUTO Gemológico Español, descripción de equipos, (<http://www.ige.org/equipos/instrucciones/tester.doc>), [consulta,04/01/2005].
- 18) KONTANI, T., Termal conductivity measurement of high conductivity materials by a transient hot-wire method, Heat Transfer Japanese Research, 19(4), 366, 1990.
- 19) Mohsenin, N. N., Thermal Properties of Foods and Agricultural Material. Gordon and Breach Science Publisher, 1980
- 20) MURAKAMY, E.G. and Okos, M.R., Measurement and prediction of thermal properties of food, in Food Properties and Computer Aided Engineering of Food Processing Systems. Shingh, R. P. and Medina A.G., Eds., Kluwer Academic Publisher, NY, 1989
- 21) MPASM assembler. User's Guide, Microchip
- 22) MPLAB IDE User's Guide, Microchip.

- 23) NASA, "Bach to Chaos: Chaotic Variations on a Classical Theme", Science News, Dec. 24, 1994, pag. 428.
  
- 24) OMEGA, Equipos de instrumentación ambiental en Chile, (<http://www.tom.cl/meteorologia/thermalogic.htm>), [consulta 4/01/2005].
  
- 25) PROLIFIC Technology Inc, PL-2303 Edition, USB to Serial Bridge Controller, Rev 1.6, April 26, 2005, Taiwan <http://www.prolific.com.tw>
  
- 26) PERRY, R. C., Chilton C. H., "Chemical Engineers Handbook", Quinta Edición. McGraw Hill, New York, 1973.
  
- 27) RAHAMA, M. S., Thermo physical properties of seafood's, Ph.D thesis, University of New South Wales, Sydney, 1991a.
  
- 28) RAHAMA, M. S., Evaluation of the precision of the modified Fitch method for thermal conductivity measurement of food, J. Food Eng., 14, 71, 1991b
  
- 29) SIDMAR. Bernhard Pack S.L. Línea de equipos CTD, ([http://www.sidmar.es/FSI/MCTD2\\_+.htm](http://www.sidmar.es/FSI/MCTD2_+.htm)), [consulta, 04/01/2005].
  
- 30) TAVERNIER, C., Microcontroladores PIC, Editorial Paraninfo S.A. 1997.
  
- 31) TB055: PS/2® to USB Mouse Translator



- 32) TB056: Demonstrating the Set\_Report Request with a PS/2® to USB Keyboard Translator Example
- 33) TB057: USB Combination Devices Demonstrated by a Combination Mouse and Game Pad Device
- 34) TB058: Demonstrating the Soft Detach Function with a PS/2® to USB Translator Example
- 35) TEKCIEN., Cursos sobre Microcontroladores PIC, Niveles Básico y Avanzado, [Tekcien Ltda.](#)
- 36) TIMOTHY J. Maloney Electrónica Industrial Moderna, Prentice Hall, Tercera Edición, Mexico, 1997.
- 37) THOMSON, W. J.: Introduction to Transport Phenomena; Prentice Hall PTR, NJ 2000.
- 38) TONG, C.H and Sheen, S., Heat flux sensor to measure effective thermal conductivity of multilayered plastic container, J. Food Processing Preserv., 16(4), 233, 1992.
- 39) USB Complete, Second Edition, Jan Axelson; Lakeview Research, 2001 (www.lvr.com)

- 40) USB - Hardware & Software por John Garney, Ed Solari, Kosar Jaff y otros (Ed. Annabooks), 2001
  
- 41) USB Firmware User's Guide (located in USB Support Firmware zip file at [www.microchip.com](http://www.microchip.com))
  
- 42) USB Specification, Version 1.1: Chapter 9 (located at [www.usb.org](http://www.usb.org))
  
- 43) Device Class Definition for Human Interface Devices (located at [www.usb.org](http://www.usb.org))
  
- 44) Van del HELD, E.F.M. and Van Drunen. F.G., A method of measuring the thermal conductivity of liquids, Physica XV(10), 865, 1949.
  
- 45) ZURITZ, C.A., McCOY, S. C. Murakami, E. G. and Blaisdell, J. L., A modified Fitch device for measuring the thermal conductivity of small food particles, Trans .ASAE, 32(2), 711, 1989

## Apéndices y Anexos

### 1. Códigos de Firmware

Los siguientes códigos fueron desarrollados en base a códigos de ejemplo tanto del estandar como de varios autores.[25,31,32,33,34,39,40,41,42,43]

```

;***** Librería "BIN_BCD.INC" *****
;
;
;=====
;
;
; Un número binario natural de 8 bits es convertido a BCD. El resultado se guarda en tres
; Posiciones de memorias llamadas: BCD_Centenas, BCD_Decenas y BCD_Unidades.
;
;
;..

; Subrutina "BIN_a_BCD" -----

CBLOCK                                ; En las subrutinas no se debe fijar la dirección
BCD_Centenas                          ; de la RAM de usuario. Se toma a continuación de
BCD_Decenas                            ; la última asignada.
BCD_Unidades
ENDC

;BIN_a_BCD
Clr BCD_Centenas                      ; Carga los registros con el resultado inicial.
Clr BCD_Decenas                       ; En principio las centenas y decenas a cero.
Movwf BCD_Unidades                    ; Se carga el número binario a convertir.
BCD_Resta10
movlw .10                             ; A las unidades se les va restando 10 en cada
subwf BCD_Unidades,W                  ; pasada. (W)=(BCD_Unidades)-10.
Btfss STATUS,C                       ; ¿C = 1?, ¿(W) positivo?, ¿(BCD_Unidades)>=10?
goto BIN_BCD_Fin                     ; No, es menor de 10. Se acabó.
BCD_IncrementaDecenas
Movwf BCD_Unidades                    ; Recupera lo que queda por restar.
Incf BCD_Decenas,F                    ; Incrementa las decenas y comprueba si ha llegado
Movlw .10                             ; a 10. Lo hace mediante una resta.
Subwf BCD_Decenas,W                   ; (W)=(BCD_Decenas)-10.
Btfss STATUS,C                       ; ¿C = 1?, ¿(W) positivo?, ¿(BCD_Decenas)>=10?
goto BCD_Resta10                     ; No. Vuelve a dar otra pasada, restándole 10 a
BCD_IncrementaCentenas                ; las unidades.
Clr BCD_Decenas                        ; Pone a cero las decenas
Incf BCD_Centenas,F                   ; e incrementa las centenas.
goto BCD_Resta10                     ; Otra pasada: Resta 10 al número a convertir.
BIN_BCD_Fin
Swapf BCD_Decenas,W                  ; En el nibble alto de (W) también las decenas.
Addwf BCD_Unidades,W                 ; En el nibble bajo de (W) las unidades.

```

```

Return                                     ; Vuelve al programa principal.

;***** Librería "RETARDOS.INC" *****
;
;=====
;
; Librería con múltiples subrutinas de retardos, desde 4 microsegundos hasta 20 segundos.
; Además se pueden implementar otras subrutinas muy fácilmente.
;
; Se han calculado para un sistema microcontrolador con un PIC trabajando con un
; cristal de cuarzo a 4 MHz. Como cada ciclo máquina son 4 ciclos de reloj, resulta que
; cada ciclo máquina tarda  $4 \times 1/4\text{MHz} = 1 \mu\text{s}$ .
;
; En los comentarios, "cm" significa "ciclos máquina".
;
; ZONA DE DATOS *****

CBLOCK
R_ContA                                     ; Contadores para los retardos.
R_ContB
R_ContC
ENDC
;
; RETARDOS de 4 hasta 10 microsegundos -----
;
; A continuación retardos pequeños teniendo en cuenta que para una frecuencia de 4 MHZ,
; la llamada a subrutina "call" tarda 2 ciclos máquina, el retorno de subrutina
; "return" toma otros 2 ciclos máquina y cada instrucción "nop" tarda 1 ciclo máquina.
;
Retardo_10micros                           ; La llamada "call" aporta 2 ciclos máquina.
nop                                         ; Aporta 1 ciclo máquina.
nop                                         ; Aporta 1 ciclo máquina.
nop                                         ; Aporta 1 ciclo máquina.
nop                                         ; Aporta 1 ciclo máquina.
nop                                         ; Aporta 1 ciclo máquina.
Retardo_5micros                             ; La llamada "call" aporta 2 ciclos máquina.
nop                                         ; Aporta 1 ciclo máquina.
Retardo_4micros                             ; La llamada "call" aporta 2 ciclos máquina.
return                                     ; El salto del retorno aporta 2 ciclos máquina.
;
; RETARDOS de 20 hasta 500 microsegundos -----
;
Retardo_500micros                           ; La llamada "call" aporta 2 ciclos máquina.
Nop                                         ; Aporta 1 ciclo máquina.
Movlw d'164'                               ; Aporta 1 ciclo máquina. Este es el valor de "K".
Goto RetardoMicros                         ; Aporta 2 ciclos máquina.
Retardo_200micros                           ; La llamada "call" aporta 2 ciclos máquina.
nop                                         ; Aporta 1 ciclo máquina.
Movlw d'64'                               ; Aporta 1 ciclo máquina. Este es el valor de "K".
Goto RetardoMicros                         ; Aporta 2 ciclos máquina.
Retardo_100micros                           ; La llamada "call" aporta 2 ciclos máquina.
Movlw d'31'                               ; Aporta 1 ciclo máquina. Este es el valor de "K".
Goto RetardoMicros                         ; Aporta 2 ciclos máquina.
Retardo_50micros                           ; La llamada "call" aporta 2 ciclos máquina.
nop                                         ; Aporta 1 ciclo máquina.
Movlw d'14'                               ; Aporta 1 ciclo máquina. Este es el valor de "K".
Goto RetardoMicros                         ; Aporta 2 ciclos máquina.
Retardo_20micros                           ; La llamada "call" aporta 2 ciclos máquina.

```

```

Movlw d'5' ; Aporta 1 ciclo máquina. Este es el valor de "K".
;
; El próximo bloque "RetardoMicros" tarda:
;  $1 + (K-1) + 2 + (K-1) \times 2 + 2 = (2 + 3K)$  ciclos máquina.
;
RetardoMicros
Movwf R_ContA ; Aporta 1 ciclo máquina.
Rmicros_Bucle
Decfsz R_ContA,F ; (K-1)x1 cm (cuando no salta) + 2 cm (al saltar).
Goto Rmicros_Bucle ; Aporta (K-1)x2 ciclos máquina.
Return ; El salto del retorno aporta 2 ciclos máquina.
;
; En total estas subrutinas tardan:
; - Retardo_500micros:
; - Retardo_200micros:
; - Retardo_100micros:
; - Retardo_50micros :
; - Retardo_20micros :
; RETARDOS de 1 ms hasta 200 ms. -----
;
Retardo_200ms ; La llamada "call" aporta 2 ciclos máquina.
Movlw d'200' ; Aporta 1 ciclo máquina. Este es el valor de "M".
Goto Retardos_ms ; Aporta 2 ciclos máquina.
Retardo_100ms ; La llamada "call" aporta 2 ciclos máquina.
Movlw d'100' ; Aporta 1 ciclo máquina. Este es el valor de "M".
Goto Retardos_ms ; Aporta 2 ciclos máquina.
Retardo_50ms ; La llamada "call" aporta 2 ciclos máquina.
Movlw d'50' ; Aporta 1 ciclo máquina. Este es el valor de "M".
Goto Retardos_ms ; Aporta 2 ciclos máquina.
Retardo_20ms ; La llamada "call" aporta 2 ciclos máquina.
Movlw d'20' ; Aporta 1 ciclo máquina. Este es el valor de "M".
Goto Retardos_ms ; Aporta 2 ciclos máquina.
Retardo_10ms ; La llamada "call" aporta 2 ciclos máquina.
Movlw d'10' ; Aporta 1 ciclo máquina. Este es el valor de "M".
Goto Retardos_ms ; Aporta 2 ciclos máquina.
Retardo_5ms ; La llamada "call" aporta 2 ciclos máquina.
Movlw d'5' ; Aporta 1 ciclo máquina. Este es el valor de "M".
Goto Retardos_ms ; Aporta 2 ciclos máquina.
Retardo_2ms ; La llamada "call" aporta 2 ciclos máquina.
Movlw d'2' ; Aporta 1 ciclo máquina. Este es el valor de "M".
Goto Retardos_ms ; Aporta 2 ciclos máquina.
Retardo_1ms ; La llamada "call" aporta 2 ciclos máquina.
Movlw d'1' ; Aporta 1 ciclo máquina. Este es el valor de "M".
;
; El próximo bloque "Retardos_ms" tarda:
; supone 1002 ciclos máquina
; que a 4 MHz son 1002  $\mu$ s = 1 ms.
;
Retardos_ms
Movwf R_ContB ; Aporta 1 ciclo máquina.
R1ms_BucleExterno
Movlw d'249' ; Aporta Mx1 ciclos máquina. Este es el valor de "K".
Movwf R_ContA ; Aporta Mx1 ciclos máquina.
R1ms_BucleInterno nop ; Aporta KxMx1 ciclos máquina.
decfsz R_ContA,F ; (K-1)xMx1 cm (cuando no salta) + Mx2 cm (al saltar).
Goto R1ms_BucleInterno ; Aporta (K-1)xMx2 ciclos máquina.
Decfsz R_ContB,F ; (M-1)x1 cm (cuando no salta) + 2 cm (al saltar).
Goto R1ms_BucleExterno ; Aporta (M-1)x2 ciclos máquina.

```

```

Return                                     ; El salto del retorno aporta 2 ciclos máquina.
;
;En total estas subrutinas tardan:
; - Retardo_200ms:
; - Retardo_100ms:
; - Retardo_50ms:
; - Retardo_20ms:
; - Retardo_10ms:
; - Retardo_5ms:
; - Retardo_2ms:
; - Retardo_1ms:
; RETARDOS de 0.5 hasta 20 segundos -----
;
Retardo_20s                               ; La llamada "call" aporta 2 ciclos máquina.
Movlw d'200'                             ; Aporta 1 ciclo máquina. Este es el valor de "N".
Goto Retardo_1Decima                     ; Aporta 2 ciclos máquina.
Retardo_10s                               ; La llamada "call" aporta 2 ciclos máquina.
Movlw d'100'                             ; Aporta 1 ciclo máquina. Este es el valor de "N".
Goto Retardo_1Decima                     ; Aporta 2 ciclos máquina.
Retardo_5s                               ; La llamada "call" aporta 2 ciclos máquina.
Movlw d'50'                             ; Aporta 1 ciclo máquina. Este es el valor de "N".
Goto Retardo_1Decima                     ; Aporta 2 ciclos máquina.
Retardo_2s                               ; La llamada "call" aporta 2 ciclos máquina.
Movlw d'20'                             ; Aporta 1 ciclo máquina. Este es el valor de "N".
Goto Retardo_1Decima                     ; Aporta 2 ciclos máquina.
Retardo_1s                               ; La llamada "call" aporta 2 ciclos máquina.
Movlw d'10'                             ; Aporta 1 ciclo máquina. Este es el valor de "N".
Goto Retardo_1Decima                     ; Aporta 2 ciclos máquina.
Retardo_500ms                           ; La llamada "call" aporta 2 ciclos máquina.
Movlw d'5'                             ; Aporta 1 ciclo máquina. Este es el valor de "N".
;
; El próximo bloque "Retardo_1Decima" tarda:
; supone 100011 ciclos máquina que a 4 MHz son 100011  $\mu$ s = 100 ms = 0,1 s = 1 ;décima de segundo.
;
Retardo_1Decima
Movwf R_ContC                             ; Aporta 1 ciclo máquina.
R1Decima_BucleExterno2
Movlw d'100'                             ; Aporta Nx1 ciclos máquina. Este es el valor de "M".
movwf R_ContB                             ; Aporta Nx1 ciclos máquina.
R1Decima_BucleExterno
movlwd'249'                             ; Aporta MxNx1 ciclos máquina. Este es el valor de "K".
movwf R_ContA                             ; Aporta MxNx1 ciclos máquina.
R1Decima_BucleInterno
nop                                         ; Aporta KxMxNx1 ciclos máquina.
decfsz R_ContA,F                         ; (K-1)xMxNx1 cm (si no salta) + MxNx2 cm (al saltar).
goto R1Decima_BucleInterno               ; Aporta (K-1)xMxNx2 ciclos máquina.
decfsz R_ContB,F                         ; (M-1)xNx1 cm (cuando no salta) + Nx2 cm (al saltar).
goto R1Decima_BucleExterno               ; Aporta (M-1)xNx2 ciclos máquina.
decfsz R_ContC,F                         ; (N-1)x1 cm (cuando no salta) + 2 cm (al saltar).
goto R1Decima_BucleExterno2              ; Aporta (N-1)x2 ciclos máquina.
return                                   ; El salto del retorno aporta 2 ciclos máquina.
;
;En total estas subrutinas tardan:
; - Retardo_20s:
; - Retardo_10s:
; - Retardo_5s:
; - Retardo_2s:
; - Retardo_1s:

```

```
; - Retardo_500ms:
; ***** Librería "RS232.INC" *****
;
=====
;
; Estas subrutinas permiten realizar las tareas básicas de control de la transmisión
; serie asincrona según normas RS-232.
;
; Los parámetros adoptados para la comunicación son los siguientes:
; - Velocidad de transmisión de 9600 baudios. La duración de cada bit será 104 µs.
; - Un bit de inicio o Start a nivel bajo.
; - Dato de 8 bits.
; - Sin paridad.
; - Dos bits de final o Stop a nivel alto.
;
; El tiempo entre bit y bit debe coincidir con el periodo de la señal leída o enviada.
; Como la velocidad de transmisión o recepción es de 9600 baudios, el periodo será:
; 1/9600 Baudios = 104 µs. Se utilizará pues la subrutina Retardos_100micros.

CBLOCK
RS232_ContadorBits
RS232_Dato
ENDC

#define RS232_EntradaPORTA,3           ; Línea por la que se reciben los datos.
#define RS232_SalidaPORTA,4           ; Línea por la que se envían los datos.
;
; Subrutina "RS232_Inicializa" -----
;
; Configura las líneas de salida y entrada del microcontrolador.

RS232_Inicializa
Bsf STATUS,RP0
Bsf RS232_Entrada           ; Esta línea se configura como entrada.
Bcf RS232_Salida            ; Esta línea se configura como salida.
Bcf STATUS,RP0
return

; Subrutina "RS232_LeeDato" -----
;
; El microcontrolador lee el dato por la línea de entrada comenzando por el bit de menor
; peso. El dato leído se envía finalmente en el registro de trabajo W.
;
; El ordenador parte siempre de un nivel alto, que es el estado que tiene cuando no
; envía información. La secuencia utilizada es:
; 1º Espera que se ejecute el pulso negativo del bit Start o flanco de bajada.
; 2º Deja pasar un tiempo una y media veces mayor que el periodo de transmisión
; para saltarse el bit de Start y lee el primer bit en su mitad.
; 3º Lee el resto de los bits de datos, esperando un tiempo igual a la duración del
; Periodo entre lectura y lectura para testearlos en mitad del bit.
;
; Salida: En el registro de trabajo W el byte leído.

RS232_LeeDato
movlwd'8'                   ; Número de bits a recibir.
movwfs RS232_ContadorBits
RS232_EsperaBitStar
btfsc RS232_Entrada         ; Lee la entrada y espera a que sea "0".
```

```

gotoRS232_EsperaBitStart                ; No, pues espera el nivel bajo.
call Retardo_100micros                  ; El primer bit debe leerlo un tiempo igual a una
call Retardo_50micros                    ; vez y media el periodo de transmisión.

RS232_LeerBit
bcfSTATUS,C                             ; Ahora lee el pin. En principio supone que es 0.
btfscRS232_Entrada                       ; ¿Realmente es cero?
bsfSTATUS,C                             ; No, pues cambia a "1".
rrfRS232_Dato,F                         ; Introduce el bit en el registro de lectura.
callRetardo_100micros                   ; Los siguientes bits los lee un periodo más tarde.
decfsz RS232_ContadorBits,F             ; Comprueba que es el último bit.
gotoRS232_LeerBit                       ; Si no es el último bit pasa a leer el siguiente.
callRetardo_200micros                   ; Espera un tiempo igual al los 2 bits de Stop.
movfRS232_Dato,W                         ; El resultado en el registro W.
return

; Subrutinas "RS232_EnviaDato" y "RS232_EnviaNúmero" -----
;
; El microcontrolador envía un dato por la línea de salida comenzando por el bit de menor
; peso. En dato enviado será el que le llegue a través del registro de trabajo W.
; 1º.Envía un "0" durante un tiempo igual al periodo de la velocidad de transmisión.
;Este es el bit de "Start".
; 2º.Envía el bit correspondiente:
; - Si va a enviar un "0" permanece en bajo durante el periodo correspondiente.
; - Si va a escribir un "1" permanece en alto durante el periodo correspondiente.
; 3º.Envía dos bits "1" durante un tiempo igual al periodo de la velocidad de
;transmisión cada uno. Estos son los dos bits de Stop.
;
; Entrada:En (W) el dato a enviar.

RS232_EnviaNumero                        ; Envía el código ASCII de un número.
addlw'0'                                ; Lo pasa a código ASCII sumándole el ASCII del 0.
RS232_EnviaDato
movwfRS232_Dato                          ; Guarda el contenido del byte a transmitir.
movlwd'8'                                ; Este es el número de bits a transmitir.
movwfRS232_ContadorBits
bcfRS232_Salida                          ; Bit de Start.
callRetardo_100micros
RS232_EnviaBit                           ; Comienza a enviar datos.
rrfRS232_Dato,F                         ; Lleva el bit que se quiere enviar al Carry para
btfssSTATUS,C                           ; deducir su valor.
gotoRS232_EnviaCero
RS232_EnviaUno
bsfRS232_Salida                          ; Transmite un "1".
gotoRS232_FinEnviaBit
RS232_EnviaCero
bcfRS232_Salida                          ; Transmite un "0".
RS232_FinEnviaBit
callRetardo_100micros                    ; Este es el tiempo que estará en alto o bajo.
decfsz RS232_ContadorBits,F             ; Comprueba que es el último bit.
gotoRS232_EnviaBit                       ; Como no es el último bit repite la operación.
bsfRS232_Salida                          ; Envía dos bits de Stop.
callRetardo_200micros
return

```

```

;***** Librería "RS232MEN.INC" *****
;
=====

```



```

;
; Estas subrutinas permiten transmitir mensajes desde el microcontrolador hacia el
; ordenador a través del puerto serie RS232.
;
; Subrutina "RS232_Mensaje" -----
;
CBLOCK
RS232_ApuntaCaracter           ; Apunta al carácter a visualizar.
RS232_ValorCaracter           ; Valor ASCII del carácter a visualizar.
ENDC

RS232_Mensaje
movwfRS232_ApuntaCaracter      ; Posición del primer carácter del mensaje.
movlwMensajes                  ; Halla la posición relativa del primer carácter
subwfRS232_ApuntaCaracter,F    ; del mensaje respecto del comienzo de
; todos los mensajes (identificados mediante
; la etiqueta "Mensajes").
decfRS232_ApuntaCaracter,F     ; Para compensar la posición que ocupa la
RS232_VisualizaOtroCaracter    ; instrucción "addwf PCL,F".
movfRS232_ApuntaCaracter,W     ; Apunta al carácter a visualizar.
callMensajes                   ; Obtiene el código ASCII del carácter apuntado.
movwfRS232_ValorCaracter       ; Guarda el valor de carácter.
movfRS232_ValorCaracter,F      ; Lo único que hace es posicionar flag Z. En caso
btfscSTATUS,Z                 ; que sea "0x00", que es código indicador final
gotoRS232_FinMensaje          ; de mensaje, sale fuera.
RS232_NoUltimoCaracter
callRS232_EnviaDato            ; Visualiza el carácter ASCII leído.
incfRS232_ApuntaCaracter,F     ; Apunta a la posición del siguiente carácter
gotoRS232_VisualizaOtroCaracter ; dentro del mensaje.
RS232_FinMensaje
return

; Subrutina "RS232_LineasBlanco" -----
;
; Visualiza unas cuantas líneas en blanco en el monitor del ordenador.

CBLOCK
RS232_ContadorLineas
ENDC

RS232_LineasBlanco
movlwd'10'                     ; Por ejemplo este número de líneas en
movwfRS232_ContadorLineas      ; blanco.
R232_LineasBlancoLazo
movlw.10                       ; Código del salto de línea
callRS232_EnviaDato
decfszRS232_ContadorLineas,F
gotoR232_LineasBlancoLazo
movlw.13                       ; Código del retorno de carro.
callRS232_EnviaDato            ; Finaliza con un retorno de carro.
return

;***** Librería "BUS_I2C.INC" *****
;
;
;=====
;
; Estas subrutinas permiten realizar las tareas básicas de control del bus serie I2C,
; por parte de un solo microcontrolador maestro.

```

```

;
; ZONA DE DATOS
*****
;
CBLOCK
I2C_ContadorBits                ; Cuenta los bits a transmitir o a recibir.
I2C_Dato                        ; Dato a transmitir o recibido.
I2C_Flags                       ; Guarda la información del estado del bus I2C.
ENDC

#define I2C_UltimoByteLeerI2C_Flags,0
; - (I2C_UltimoByteLeer)=0, NO es el último byte a leer por el maestro.
; - (I2C_UltimoByteLeer)=1, SÍ es el último byte a leer por el maestro.

; La definición de las líneas SCL y SDA del bus I2C se puede cambiar según las
; necesidades del hardware.

#define SCLPORTA,3                ; Línea SCL del bus I2C.
#define SDAPORTA,4                ; Línea SDA del bus I2C.
;
; Subrutina "SDA_Bajo" -----
;
SDA_Bajo
bsfSTATUS,RP0                    ; Configura la línea SDA como salida.
bcfSDA
bcfSTATUS,RP0
bcfSDA                          ; SDA en bajo.
return
;
; Subrutina "SDA_AltaImpedancia" -----
;
SDA_AltaImpedancia
bsfSTATUS,RP0                    ; Configura la línea SDA entrada.
bsfSDA                          ; Lo pone en alta impedancia y, gracias a la
bcfSTATUS,RP0                    ; Rp de esta línea, se mantiene a nivel alto.
return
;
; Subrutina "SCL_Bajo" -----
;
SCL_Bajo
bsfSTATUS,RP0
bcfSCL                          ; Configura la línea SCL como salida.
bcfSTATUS,RP0
bcfSCL                          ; La línea de reloj SCL en bajo.
return
;
; Subrutina "SCL_AltaImpedancia" -----
;
SCL_AltaImpedancia
bsfSTATUS,RP0                    ; Configura la línea SCL entrada.
bsfSCL                          ; Lo pone en alta impedancia y, gracias a la Rp
bcfSTATUS,RP0                    ; de esta línea, se mantiene a nivel alto.
SCL_EsperaNivelAlto
btfssSCL                        ; Si algún esclavo mantiene esta línea en bajo
gotoSCL_EsperaNivelAlto        ; hay que esperar.
return
;
; Subrutina "I2C_EnviaStart" -----

```

```

;
; Esta subrutina envía una condición de Start o inicio.
;
I2C_EnviaStart
callSDA_AltaImpedancia          ; Línea SDA en alto.
callSCL_AltaImpedancia          ; Línea SCL en alto.
callRetardo_4micros             ; Tiempo tBUF del protocolo.
callSDA_Bajo                    ; Flanco de bajada de SDA mientras SCL está alto.
callRetardo_4micros             ; Tiempo tHD;STA del protocolo.
callSCL_Bajo                    ; Flanco de bajada del reloj SCL.
callRetardo_4micros
return
;
; Subrutina "I2C_EnviaStop" -----
;
; Esta subrutina envía un condición de Stop o parada.
;
I2C_EnviaStop
callSDA_Bajo
callSCL_AltaImpedancia          ; Flanco de subida de SCL.
callRetardo_4micros            ; Tiempo tSU;STO del protocolo.
callSDA_AltaImpedancia         ; Flanco de subida de SDA.
callRetardo_4micros            ; Tiempo tBUF del protocolo.
return
;
; Subrutina "I2C_EnviaByte" -----
;
; El microcontrolador maestro transmite un byte por el bus I2C, comenzando por el bit
; MSB. El byte a transmitir debe estar cargado previamente en el registro de trabajo W.
; De la subrutina ejecutada anteriormente I2C_EnviaStart o esta misma I2C_EnviaByte,
; la línea SCL se debe encontrar a nivel bajo al menos durante 5 µs.
;
I2C_EnviaByte
movwI2C_Dato                    ; Almacena el byte a transmitir.
movlw0x08                       ; A transmitir 8 bits.
movwI2C_ContadorBits
I2C_EnviaBit
rI2C_Dato,F                     ; Chequea el bit, llevándolo previamente al Carry.
btfscSTATUS,
gotoI2C_EnviaUno
I2C_EnviaCero
callSDA_Bajo                    ; Si es "0" envía un nivel bajo.
gotoI2C_FlancoSCL
I2C_EnviaUno
callSDA_AltaImpedancia          ; Si es "1" lo activará a alto.
I2C_FlancoSCL
callSCL_AltaImpedancia          ; Flanco de subida del SCL.
callRetardo_4micros            ; Tiempo tHIGH del protocolo.
callSCL_Bajo                    ; Termina el semiperiodo positivo del reloj.
callRetardo_4micros            ; Tiempo tHD;DAT del protocolo.
decfszI2C_ContadorBits,F       ; Lazo para los ocho bits.
gotoI2C_EnviaBit
callSDA_AltaImpedancia          ; Libera la línea de datos.
callSCL_AltaImpedancia         ; Pulso en alto de reloj para que el esclavo
callRetardo_4micros            ; pueda enviar el bit ACK.
callSCL_Bajo
callRetardo_4micros
return

```

```

;
; Subrutina "I2C_LeeByte" -----
;
; El microcontrolador maestro lee un byte desde el esclavo conectado al bus I2C. El dato
; recibido se carga en el registro I2C_Dato y lo envía a la subrutina superior a través
; del registro W. Se empieza a leer por el bit de mayor peso MSB.
; De alguna de las subrutinas ejecutadas anteriormente I2C_EnviaStart, I2C_EnviaByte
; o esta misma I2C_LeeByte, la línea SCL lleva en bajo al menos 5 µs.

I2C_LeeByte
movlw0x08                                ; A recibir 8 bits.
movwI2C_ContadorBits
callSDA_AltaImpedancia                  ; Deja libre la línea de datos.
I2C_LeeBit
callSCL_AltaImpedancia                  ; Flanco de subida del reloj.
bcfSTATUS,C                            ; En principio supone que es "0".
btfscSDA                                ; Lee el bit
bsfSTATUS,C                            ; Si es "1" carga 1 en el Carry.
rlfI2C_Dato,F                          ; Lo introduce en el registro.
callSCL_Bajo                            ; Termina el semiperiodo positivo del reloj.
callRetardo_4micros                    ; Tiempo tHD;DAT del protocolo.
decfszI2C_ContadorBits,F               ; Lazo para los 8 bits.
gotoI2C_LeeBit
;
; Chequea si este es el último byte a leer para enviar o no el bit de reconocimiento
; ACK en consecuencia.
;
btfss I2C_UltimoByteLeer                ; Si es el último, no debe enviar
                                         ; el bit de reconocimiento ACK.
callSDA_Bajo                            ; Envía el bit de reconocimiento ACK
                                         ; porque todavía no es el último byte a leer.
callSCL_AltaImpedancia                  ; Pulso en alto del SCL para transmitir el
callRetardo_4micros                    ; bit ACK de reconocimiento. Este es tHIGH.
callSCL_Bajo                            ; Pulso de bajada del SCL.
callRetardo_4micros
movfI2C_Dato,W                          ; El resultado se manda en el registro de
return                                  ; de trabajo W.

***** Conductivity.asm *****
;
;
; =====
;
; El microcontrolador lee constantemente las entradas analógicas PCF8591 y las
; envía por RS232 al Interprete USB
;
; ZONA DE DATOS
*****

__CONFIG _CP_OFF & _WDT_OFF & _PWRTE_ON & _XT_OSC
LIST P=16F84
INCLUDE <P16F84A.INC>

CBLOCK 0x0C
ENDC

RetornoCarroEQU.13                      ; Código de tecla "Enter" o "Retorno de Carro".
CambioLineaEQU.10                      ; Código para el cambio de línea.

```

```
PCF8591_DireccionEscrituraEQUb'10011110'
PCF8591_DireccionLecturaEQUb'10011111'
```

```
;
; ZONA DE CÓDIGOS
```

```
*****
```

```
ORG0
```

```
Inicio
```

```
callI2C_EnviaStart ; Va a configurar el PCF8591.
movlwPCF8591_DireccionEscritura ; Apunta al dispositivo.
```

```
callI2C_EnviaByte
movlwb'00000000' ; Carga la palabra de control utilizando la
callI2C_EnviaByte ; entrada AIN0 en modo simple.
callI2C_EnviaStop ; Termina la configuración
callI2C_EnviaStart ; Comienza a leer.
movlwPCF8591_DireccionLectura ; Apunta al dispositivo.
```

```
callI2C_EnviaByte
callI2C_LeeByte ; La primera lectura es incorrecta y por lo tanto
; la desecha
```

```
callRS232_Inicializa ; van a utilizar en la comunicación con el puerto
```

```
Principal
```

```
callI2C_LeeByte ; Lee la entrada analógica.
```

```
movlwMensaje0 ; Carga la primera posición del mensaje.
```

```
callRS232_Mensaje ; Lo visualiza en el ordenador.
```

```
callRetardo_5s
```

```
gotoPrincipal
```

```
; Subrutinas "Visualiza" -----
```

```
;
```

```
; Visualiza el valor que se le introduce por el registro de trabajo W en formato de tensión.
```

```
; Hay que tener en cuenta que el PCF8591 del esquema trabaja con una resolución de
```

```
; LSB=10mV, el valor de entrada será 10 veces menor que la tensión real expresada en
```

```
; milivoltios.
```

```
;
```

```
;
```

```
CBLOCK
```

```
Auxiliar
```

```
ENDC
```

```
Visualiza
```

```
movwfAuxiliar ; Lo guarda.
```

```
callRS232_EnviaDato ; Se sitúa al principio de la primera línea.
```

```
movlwMensajeTension ; Visualiza la tensión deseada.
```

```
callMensaje
```

```
movfAuxiliar,W ; Recupera el dato a visualizar y lo
```

```
callBIN_a_BCD ; pasa a BCD.
```

```
movfBCD_Centenas,W ; Visualiza las centenas que corresponden a las
```

```
callLCD_Nibble ; unidades de voltios.
```

```
movlw'.' ; Visualiza el punto decimal.
```

```
call LCD_Caracter
```

```
movfBCD_Decenas,W ; Visualiza las decenas que corresponden a las
```

```
callLCD_Nibble ; décimas de voltios.
```

```
movfBCD_Unidades,W ; Visualiza las unidades que corresponden a las
```

```
callLCD_Nibble ; centésimas de voltios.
```

```
movlw MensajeVoltios
```

```
call LCD_Mensaje
```

```
return
Mensajes
addwfPCL,F
MensajeVoltios
DT " V. ",0x00
```

```
INCLUDE <BUS_I2C.INC>
INCLUDE <RETARDOS.INC>
INCLUDE <BIN_BCD.INC>
    INCLUDE <RS232.INC>
INCLUDE <RS232MEN.INC>
END
```

### **3.3 Microcontrolador interprete UART USB.**

Una vez obtenido del dato por UART debe ser transformado a USB para que el host del equipo pueda interpretar y enviar el dato a la función.

USB

```
;=====
; HIDUART.ASM
;=====
```

CPU63001

;Define el flujo de control por software

```
include"usb1regs.inc"
include"ram.inc"
```

```
;=====
;
;
; Vector de Interrupción
;
;=====
```

org 00h

```
jmp reset ; Resetea el vector
jmp Isr128 ; 128us pausa
jmp Isr1ms ; 1024ms pausa
jmp IsrEp0 ; endpoint 0
jmp IsrEp1 ; endpoint 1
jmp reset ; reserved interrupt
jmp SerialReceiveByte ; interrupción I/O (serial rx startbit)
jmp IsrCext ; llama a todas las interrupciones
```

```
;=====
;
;
; Resetea el proceso
;
;=====
;
;
; Pone todos SFRs al estado inicial
; Limpia la ram
;
;=====
```

```

reset:

    mov a, DATA_STACK_POINTER_START ;Fija el valor inicial del puntero de datos
    swap a, dsp

iord scr
and a, RESET_MASK
cmp a, POR_B          ;Prueba el puerto y Resetea
jz por_reset

and A, USBR_B          ;Prueba USB y Resetea
jnz bus_reset

    mov a,00h
    iowr cext          ;Suspende y continua en la siguiente línea

jmp wdog_reset

;=====
; Interrupción de 128us
;
;=====

Isr128:
push a
.Reti:
mov a, [interrupt_mask]
;=====
;
;Isr1ms
;
;=====

Isr1ms:
    IF 0
push a

mov a, [interrupt_mask]ipret gier

    ENDIF
;=====
;
;IsrEp0
;
;=====

IsrEp0:
    push a

    mov a,0
    mov [suspend_counter], a

iord EP0_RX_Status          ;Lee el estatus de EP0 RX y chequea el bit
and a, SETUP_B

```

```

jz NotSetup                                ;si no es el bit va not setup

    call DecodeRequest    ;Analiza el paquete.

jmp Reti

NotSetup:
iord EP0_RX_Status
and a, OUT_B                                ; Si no es la salida del paquete prueba el siguiente.
jz InPacketReceived

                                ;El paquete de salida. Compruebe si esta completo. Si faltan bit,
                                ;Ignore y vuelva a la interuupcion.

iord EP0_RX_Status
and a,D0h                                ; Los datos seran siempre 2 o más. (El conteo incluye
jz Reti                                ; el CRC para los datos). Podemos determiniar el conteo
                                ; si tomamos los bits 7,6,4 del set en EP0_RX_Status.

iord gier                                ; desconecte la interrupción EP0 en el proceso de salida pkt
and a, ~EPOIE_B
iowr gier

iord USB_SCR                                ; No deje afuera los pkts
and a, ~ENOUTS_B
iowr USB_SCR

    ; en este punto recibimos la salida de los datos.
call ControlWriteRamDataStage

jmp Reti

InPacketReceived:                        ; Si no tenemos un paquete de salida se deja libre.

iord EP0_TX_Config                        ; no continúe a menos que el xfer del paket pasado sea
                                ;completo
and a, INEN_B                            ; observe la entrada permiten el bite cuando el anfitrión
                                ; ACKs se despejó
                                jnz Reti

    mov a, [current_reportid]            ; realiza la llamada para transmitir el reporte
    cmp a, RID_TRANSMIT                  ; (Se completa la transmicion?)
    jz ep0_reportdone

    cmp a, RID_COMMAND                   ; (Se ha completado la transmicion del reporte?)
    jnz readINdata                       ; EL host mira los datos(enumeration)

ep0_reportdone:

    mov a, ~BUSY_TRANSMITTING            ; informa que se cumplido un ciclo
    and [fwcr], a                        ; suspende la condición

    mov a, SERIAL0RXBIT                  ; habilita los bit's de parada
iowr P0_interrupt
    mov a, ENUMERATED_MASK              ; rehabilita los bit's de parada

```



```

    mov [interrupt_mask],a

IFDEF SOFTWARE_FLOW_CONTROL
    mov a,XON                ;Llama al registro serial con XON
    mov [serial_tx_reg],a    ;y transmite
    call SerialTransmitByte
ENDIF

    FLOW_ON                  ; habilita el flujo de control (hardware)

    jmp Reti

readINdata:
call ControlReadDataStage    ; continua enviando los paquetes de control de
                                ; lectura

Reti:
mov a, [interrupt_mask]      ; habilita los ceros de interrupción de endpoint
ipret gier

;=====
;
;IsrEp1
;
;
;
;=====

IsrEp1:
push a

iord EP1_TX_Config
xor a, DATA10_B
iowr EP1_TX_Config

mov a, [interrupt_mask]
ipret gier

;=====
;IsrCext
;
;This interrupt is not used in this code
;
;=====

IsrCext:
push a;
mov a, [interrupt_mask]
ipret gier

;=====
;
;process_reset
;
;=====

```

```

por_reset:
mov a, ffh
iowr P0_data           ;Fija todas las salidas en 1
iowr P1_data

    mov a, 00h
    iowr P0_interrupt
iowr P1_interrupt
iowr P0_pullup         ;fije los pullups en polaridad baja
iowr P1_pullup

mov a, 0fh             ;Fije todos los registros en el valor más alto
iowr P0_sink0
iowr P0_sink1
iowr P0_sink2
iowr P0_sink3
iowr P0_sink4
iowr P0_sink5
iowr P0_sink6
iowr P0_sink7

iowr P1_sink0
iowr P1_sink1
iowr P1_sink2
iowr P1_sink3
iowr P1_sink4
iowr P1_sink5
iowr P1_sink6
iowr P1_sink7
iowr wdt               ; limpia el timer wdt

mov a, bottom_of_ram mov x, top_of_ram
ram_clr_loop:

; limpia toda la ram
mov [x+0], a
dec x
jnc ram_clr_loop

iord scr
and a, USBR_B
jnz bus_reset

suspend_reset:
iowr wdt

    mov a,01h
    iowr cext
mov A, (RUN_B + SUSPEND_B)
iowr scr
nop
jmp suspend_reset ; espera hasta que el BUS sea resetreado
wdog_reset:
jmp suspend_reset     ;suspende y espera que el BUS USB sea reseteado

bus_reset:

```

```

mov A, RUN_B                                ; limpia todo y resetea los bit
iowr scr

    mov a,00h                                ; limpa y suspende una linea
    iowr cext

mov A, ENUMERATE_MASK
mov [interrupt_mask],A
iowr gier

;=====
;
;
; Inicialización
;
;=====

;FIRMWARE VARIABLES
;-----

    mov a, 0                                ; inicia el rx del buffer
    mov [rx_inptr],a                        ; colecciona todos los datos seriales
    mov [rx_outptr],a                      ; y los reune para leer el USSB

mov A, 00h
mov [serial_buf_datacount],A                ; Inicia la numeracion de bytes del rx en
                                           ;buffer cero

;CLEAR fwcr (Firmware Registro de Control)
mov A,00h
mov [fwcr],A

;SET INICIAL BAUDIO
mov a,BAUD_BITDELAY_DEFAULT
mov [baud_bitdelay],a                      ;programa el retraso

;GPIO Inicialización
;-----

;PORT 0 - PROGRAMA puestos I/O
mov a, P0PULLUPS                          ;P0.*, habilita todas arriba
iowr P0_pullup

    mov a, 0FDh                            ;P0.0=OUT, Config P0.1=IN, todas las demas OUT
iowr P0_data

;PORT 0 - especificacion de pin
iord P0_data
or a,(SERIAL0TXBIT+SERIAL0RXBIT)           ;Set TX pin a 5V

iowr P0_data

iord P1_data
and a, ~nRST                             ;Set resetea LOW (activo)
iowr P1_data

FLOW_OFF

```

```

DELAY10MS                ;Mantiene salida baja.

iord P1_data
or a, nRST                ;Fiaja la salida alta, listo para escuchar
iowr P1_data

;PORT 1 - PROGRAMA PUERTO I/O

mov a, 00h                ;habilita pullups
iowr P1_pullup

;PORT 1 - APLICACION ESPECIFICA SETEO INICIAL PIN

holdoff:
iowr wdt

mov A, [interrupt_mask]   ; ep0 las interrupciones fijarán la máscara de la interrupción a
ENUMERACION
cmp A, ENUMERATED_MASK   ; cada enumerador es configurado y estabilizado.
jnz holdoff

FLOW_ON                   ; indica que limpio lo enviado( rx)

; otra configuracion
mov a, SERIAL0RXBIT       ; Habilita interrupciones GPIO en P0.1
                           ; (pin serial rx )
iowr P0_interrupt

mov a, 0fh                ;Fija los registros en el valor más alto
iowr P0_sink0

UserReset:

;=====
;
;
;main
;
;=====

main:
iowr wdt                  ; llama al guardián

call SuspendCounter       ; Ausencia de actividad del usb
call CheckSuspend         ; ve si el usb ha sido detenido y necesita ser suspendido

iord EP1_TX_Config        ; si hay tx indica
and a, INEN_B             ; entonces no compruebe si hay actividad
jnzmain                  ; no hay manera segura de ponerse al día hasta que ha sucedido el TX.

mov a, [rx_inptr]
cmp a, [rx_outptr]
jz main

mov a, 0
mov [ep1_byte_count], a

```

```

mgetbyte:
    mov x, [rx_outptr]                ; recibe el byte del uart
    mov a, [X+RX_BUF_PTR_START]      ; suma el siguiente byte del uart

    inc X                            ; posiona el buffer
    swap a, X                        ; graba el byte (en x),
    cmp a, SERIAL_BUF_SIZE          ; EL buffer ha llago al final?
    jnz .updptr                      ; no, apenas continua
    mov a, 0                         ; si, cero direccionado ene l índice
.updptr:
    mov [rx_outptr], a

    swap a, x                        ; resibe el bite en a
    mov x, [ep1_byte_count]          ; empieza el contador ep1 fifo
    mov [X+(EP1_FIFO+2)], a          ; reemplaza el byte en FIFO

    inc [ep1_byte_count]             ; incrementa el contador de byte
    mov a, [ep1_byte_count]          ; en a y prueba
    cmp a, 6                         ; 1 RID + 1 largo + 6 datos
    jz msendem

    mov a, [rx_inptr]                ; Hay otro byte en al buffer?
    cmp a, [rx_outptr]
    jnz mgetbyte                    ; si, envielo

msendem:
    mov a, RID_RECEIVE               ; llama a FIFO y sube los datos
    mov [EP1_FIFO], a
    mov a, [ep1_byte_count]          ; valida los datos con el contador
    mov [EP1_FIFO+1], a

iord EP1_TX_Config                  ;Habilita EP1 a TX para enviar los bites contadors
and a, DATA10_B                   ; conserva los daos
or a, (INEN_B + EP1EN_B + (EP1_REPORT_LENGTH + 1))
iowr EP1_TX_Config

    FLOW_ON                        ; envia los datos una vez más.

jmp main

;=====
SuspendCounter:
;
;
; implementa la parada en ciclo contador del main
;
    mov a, [fwcr]                  ; esta realizando muchos trabajos del uart TX?
    and a, BUSY_TRANSMITTING      ; (si es asi el contador no es confiable,
    jnz sc_exit                   ; no necesitamos preocuparnos
                                ; suspendemos alrededor de todos modos)

    iord timer
    rlc
    jc lookforhigh
lookforlow:
    mov a, TIMER_SYNC
    and a, [fwcr]
    jnz sc_exit

```

```

    mov a, TIMER_SYNC
    or [fwcr], a

    jmp testactivity

lookforhigh:
    mov a, TIMER_SYNC
    and a, [fwcr]
    jz sc_exit

    mov a, ~TIMER_SYNC
    and [fwcr], a

testactivity:
iord USB_SCR
and a, BUSACT_B
jz .IncCtr                                ;no ha habido ninguna actividad del usb

    iord gier                                ; recoge todos los bits habiles

    push a                                    ; graba

    mov a, 0
    iowr gier
; desabilita la protección USB_SCR
iord USB_SCR
and a, ~BUSACT_B                            ; limpia la actividad bit
iowr USB_SCR                                ; reestablece
    pop a
    iowr gier

    mov a, 00h                                ; contador a cero
    mov [suspend_counter], a
    jmp sc_exit

.IncCtr:
inc [suspend_counter]                        ; si no hay actividad de suspende

sc_exit:

    ret

;=====
;
;
;Check Suspend
;
;=====

CheckSuspend:

    ;Chequea y suspende el timer y baja a actividad del bus
    mov a, [suspend_counter]
    sub a, SUSPEND_TIME
    jc .Ret

    ;Limpia y suspende el contador por 1ms

```

```

mov a, 00h
mov [suspend_counter], a

;Operacion general
;Graba el estatus del puerto I/O
iord P0_data
push a
iord P1_data
push a

iord P0_pullup
push a
; esta instrucción no trabaja en 3650.
; el 3650 pone P0 en pullup a 9h, codigo manual 8.

iord P1_pullup
push a

mov a, ffh
iowr P0_data
iowr P1_data
;enumeración mínima durante la suspensión
; todos los puestos en 1 lógico
; controlador externo.

mov a, 00h
iowr P0_pullup
iowr P1_pullup
    mov a, 01h
    iowr cext

iord gier
and a, SUSPEND_MASK
iowr gier

iord scr
or a, SUSPEND_B
iowr scr

nop
;Restablece el estatus de los puertos

    mov a, 00h
    iowr cext

    pop a
    iowr P1_pullup

op a
    mov a, P0PULLUPS
    iowr P0_pullup

pop a
iowr P1_data
pop a
iowr P0_data

;END OF GENERIC SUSPEND OPERATIONS

mov a, [interrupt_mask]
iowr gier

```

```
.Ret:
ret
```

```
;=====
;
;NO Enumerado
;
;=====
;
;Este modulo trabaja cuando la
; enumeracion es incompleta
;
;=====
```

```
NotEnumerated:
```

```
jmp main
```

```
;=====
```

```
handle_rid_command:
    cmp a, (end_baud_table-baud_table)
    jc hrc_lookup
    mov a, 0          ; index error
hrc_lookup:
    index baud_table
    mov [baud_bitdelay],a
    ret
```

```
XPAGEOFF
```

```
baud_table:
    db BAUD_BITDELAY_DEFAULT
    db BAUD_BITDELAY_2400
    db BAUD_BITDELAY_4800
    db BAUD_BITDELAY_9600
    db BAUD_BITDELAY_19200
    db BAUD_BITDELAY_38400
    db BAUD_BITDELAY_57600
end_baud_table:
```

```
XPAGEON
```

```
org 302h
```

```
;----- Rutinas del puerto Serial -----
;
; Sistema de transmision serial programable a diferentes tasa de Baudios
;
;
; El RX y TX son programados con pullups bajo.
; XPAGEOFF y XPAGEON se uda desconectado en el compilador XPAGE insertado
; en ciclos de tiempos criticos.
;
;=====
```



```

;=====
;
;
;SerialReceiveByte
;
; DESCRIPCION: recibe un byte serial en pin RX
;
;=====

```

SerialReceiveByte:  
 XPAGEOFF  
 push a

```

    iord EP0_RX_Status          ; limpia antes de realizar la llamada
    and a, ~(SETUP_B)          ; mira la coleccion de bits
    iowr EP0_RX_Status

```

push X

```

mov a, 0                        ; +4 deshabilita cualquier otro bit de interrupcion
iowr P0_interrupt              ; +5

```

```

    mov X,08h                  ;Inicia la recepción de 8 bits

```

```

    mov a,[baud_bitdelay]
    or a,0                     ; limpia el carro
    rrc                        ; divide bit en dos
    inc a
    BIT_DELAY                  ; espera el tiempo de 1/2 bit y envia un bit de inicio

```

SerialReceiveData:

```

    mov a,[baud_bitdelay]
    BIT_DELAY                  ; espera hata que se llena todos los bit

```

```

    iord EP0_RX_Status
    and a, SETUP_B
    jnz srx_exit

```

debug2toggle

```

    iord P0_data                ; testea el bit
    and a,SERIAL0RXBIT         ; limpia registro C
    mov a, [serial_rx_reg]      ; mueve registro a
    jnz GotOne

```

GotZero:

```

    nop
    jmp rotate

```

GotOne:

```

    cpl
    cpl

```

rotate:

```

    rrc                        ; rota el ultimo bit (en c) dentro de MSB
    mov [serial_rx_reg], a

```

```

    dec X                      ; decrementa el contador

```

```

    jnz SerialReceiveData                ; sino, sigue en otro

XPAGEON

    push a
    mov a, FLOW_CTL_OFF                 ; si el flujo de control para mantiene a
    and a, [fwcr]                       ; el byt es rx
    pop a                               ; (no afecta Z)
    jnz waitbit

    mov X, [rx_inptr]
    mov [X+RX_BUF_PTR_START], a

    inc X                               ; incrementa la posición en el buffer
    swap a, X
    cmp a, SERIAL_BUF_SIZE
    jnz updptr
    mov a, 0
updptr:
    mov [rx_inptr], a

    inc a
    cmp a, SERIAL_BUF_SIZE
    jnz calcptr
    mov a, 0
calcptr:                               ; siguiente posición
    cmp a, [rx_outptr]
    jnz waitbit
    FLOW_OFF

waitbit:
    iord P0_data
    and a, SERIAL0RXBIT
    jz waitbit

    mov a, SERIAL0RXBIT                 ; +4 habilita el bit de interrupcion
    iowr P0_interrupt                   ; +5

srx_exit:
; termina w/o y habilita las interrupciones
    pop X
    mov a, [interrupt_mask]
ipret gier

```

```

;=====
;
;
; SerialTransmitByte
;
; DESCRIPCION:  Transmite 8N1.  Transmite un byte en el pin TX
;               primer LSB.
;
;
; USAGE:       call SerialTransmitByte
; USES:        [serial_tx_reg]
;
;=====

```

```

SerialTransmitByte:
    push X
    mov X,08h                                ; 8 bits a tx

    iord P0_data                                ; BIT de inicio
    and a,~SERIAL0TXBIT                        ; TX pin a 0V
    or a,SERIAL0RXBIT                          ;reprograma pin RX a entrada.
    iowr P0_data                              ;escribe el nuevo valor al puerto

XPAGEOFF
    mov a,[baud_bitdelay]
    inc a
    BIT_DELAY                                ; espera hata que se llena todos los bit

    nop

SerialTransmitBit:
    mov a,[serial_tx_reg]                    ;5 envia byte a tx
    rrc a                                    ;4 5 rota el bit al interior del carro
    mov [serial_tx_reg],a                    ;5 9 graba el nuevo bit
    jnc SerialTransmit0                      ;5(4) 14 si limpia el carro, envia un 0

SerialTransmit1:
    iord P0_data                                ;5 18 lee el estado del puerto
    or a,(SERIAL0TXBIT+SERIAL0RXBIT)          ;4 23 pin TX pin a 5V
    iowr P0_data                              ;5 28 escribe el nuevo valor en el puerto
    nop
    jmp txbit                                ;5 33

SerialTransmit0:
    iord P0_data                                ;4 19 lee el estado del puerto
    and a,~SERIAL0TXBIT                        ;4 23 pin TX pin a 0V
    or a,SERIAL0RXBIT                          ;4 27 pin RX estado alto
    iowr P0_data                              ;5 32 escribe el nuevo valor en el puerto

    nop
    nop

txbit:
    mov a,[baud_bitdelay]                    ; 38/ 37
    inc a
    BIT_DELAY                                ; espera hasta que se llena todos los bit

    dec X
    jnz SerialTransmitBit
XPAGEON

    iord P0_data                                ;lee el estado del puerto
    or a,SERIAL0TXBIT                          ;pin TX a 5V
    or a,SERIAL0RXBIT                          ;Reprograma pin RX a entrada.
    iowr P0_data                              ;escribe el nuevo valor en el puerto

    mov a,[baud_bitdelay]                    ; bits de parada
    BIT_DELAY                                ; espera hasta que se llena todos los bit
    mov a,[baud_bitdelay]
    BIT_DELAY                                ; espera hasta que se llena todos los bit
    mov a,[baud_bitdelay]

```

```

BIT_DELAY                                ; espera hasta que se llena todos los bit

pop X
ret                                       ;retorna a la llamada de la subrutina

;===== final de rutinas puerto Serial =====

include "usbcode.asm"

;=====
; RAM.INC
;=====;

; Declaración de variables Macros en Ram;
;=====

;

PL-2303_VID:        equ 0C04h
PL-2303_PID:        equ 0100h
PL-2303_VER:        equ 0100h                ; USB Firmware Version No. (BCD)

bottom_of_ram:      equ 00h

protocol_status:    equ 18h
remote_wakeup_status: equ 19h
suspend_counter:    equ 1Ah
interrupt_mask:     equ 1Bh
loop_counter:       equ 1Ch
endp0_data_toggle:  equ 1Dh
data_count:         equ 1Eh
data_start:         equ 1Fh
configuration_status: equ 20h

;APLICACION ESPECIFICA DE LAS VARIABLES RAM
serial_rx_reg:      equ 21h
serial_tx_reg:      equ 21h
ep1_byte_count:     equ 22h

rx_inptr:           equ 23h
rx_outptr:          equ 24h
current_reportid:   equ 25h

baud_bitdelay:      equ 26h                                ;Ajusta el valor de los baudios.
serial_buf_datacount: equ 27h                                ;valida los datos de rx en buffer buffer
fwcr:               equ 28h                                ;Control de registro.
transmit_count:     equ 29h                                ; El número de bytes en el set del Puerto a TX

RX_BUF_PTR_START:   equ 2Bh
RX_BUF_PTR_END:     equ 67h
SERIAL_BUF_SIZE:    equ (RX_BUF_PTR_END - RX_BUF_PTR_START)

```

```

DATA_STACK_POINTER_START: equ 70h

;=====
;
; Declaración FIFO
;
;=====

EP0_FIFO:          equ 70h      ; control endpoint 0 fifo
EP1_FIFO:          equ 78h      ; dato 1 endpoint fifo

bmRequestType:     equ EP0_FIFO
bRequest:          equ EP0_FIFO + 1
wValue:            equ EP0_FIFO + 2                ; valor por defecto w (8-bits)
wValueHi:          equ EP0_FIFO + 3
wIndex:            equ EP0_FIFO + 4                ; index por defecto (8-bits)
wIndexHi:          equ EP0_FIFO + 5
wLength:           equ EP0_FIFO + 6                ; largo por defecto (8-bits)
wLengthHi:         equ EP0_FIFO + 7

top_of_ram:        equ 7fh

;=====
;
; DEclaracion de constantes
;
;=====

EP1_REPORT_LENGTH: equ 07h

;Aplicacion especifica de mascara
ENUMERATE_MASK:    equ EP0IE_B
ENUMERATED_MASK:   equ EP0IE_B+GPIOIE_B+EP1IE_B
TRANSMITTING_MASK: equ EP0IE_B                ; use durante transmision uart
SUSPEND_MASK:      equ 00h
INTDISABLE_MASK:    equ 00h

;=====
;
; USB Declaración de constantes
;
;=====

; valores para bmRequestType

HOST_TO_DEVICE:    equ 00h
HOST_TO_INTERFACE: equ 01h
HOST_TO_ENDPOINT:  equ 02h

DEVICE_TO_HOST:     equ 80h
INTERFACE_TO_HOST:  equ 81h
ENDPOINT_TO_HOST:   equ 82h

CLASS:              equ 20h

```

```

VENDOR:          equ 40h

; otra EP0 constantes

EPI_STALLED:      equ 0    ; recipiente endpoint
REMOTE_WAKEUP:    equ 1    ; recipiente recurso

DISABLE_REMOTE_WAKEUP: equ 0    ; bit[1] = 0
ENABLE_REMOTE_WAKEUP: equ 2    ; bit[1] = 1

SUSPEND_TIME:     equ 15h

;=====
;
;
;  Aplicacion Especific;
;=====

POPULLUPS:        equ 00h    ;P0.*, Habilita todas a pullups

; Port 0
SERIAL0TXBIT:      equ 01h    ;P0_dato
SERIAL0RXBIT:      equ 02h    ;P0_dato

; Port 1
nCTS:              equ 01h
nDATA_AVAILABLE:  equ 02h
nOE:               equ 04h
nRST:              equ 08h

XON:               equ 11h          ;valor ascii para XON
XOFF:              equ 13h          ;valor ascii para XOFF

; FWCR propositogeneral de los bit de registro
TIMER_SYNC:        equ 01h
FIRSTBYTE_FLG_MASK: equ 02h
BUSY_TRANSMITTING: equ 04h

FLOW_CTL_OFF:      equ 08h          ; rx buffer esta completo

RID_COMMAND:       equ 3h          ; reservado para uso futuro.
RID_TRANSMIT:      equ 2h          ; Este reporte ID es usado por host por tx bytes
                                   (salida uart)
RID_RECEIVE:       equ 1h          ; dato ep1 para uart rx

BAUD_BITDELAY_57600: equ 05
BAUD_BITDELAY_38400: equ 09
BAUD_BITDELAY_19200: equ 22
BAUD_BITDELAY_9600:  equ 48
BAUD_BITDELAY_4800:  equ 100
BAUD_BITDELAY_2400:  equ 205

BAUD_BITDELAY_DEFAULT: equ BAUD_BITDELAY_9600

;*****
;
; MACROS

MACRO debug1on
    iord P0_data

```

```

    and a, 0fbh
    or a, SERIAL0RXBIT
    iowr P0_data
ENDM

```

```

MACRO debug1off
    iord P0_data
    or a, (4+SERIAL0RXBIT)
    iowr P0_data
ENDM

```

```

MACRO debug2on
    iord P0_data ; +5
    and a, 0f7h ; +4
    or a, SERIAL0RXBIT ; +4
    iowr P0_data ; +5
ENDM

```

```

MACRO debug2off
    iord P0_data
    or a, (8+SERIAL0RXBIT)
    iowr P0_data
ENDM

```

```

MACRO debug2toggle
    iord P0_data
    or a, SERIAL0RXBIT
    xor a, 8
    iowr P0_data
ENDM

```

```

MACRO BIT_DELAY
;=====
;
;
; bit retardo
;
; DESCRIPCION: ajusta el bit de retardo
;
; USAGE: BIT_DELAY
;
; retarda un ciclo mas 24 =2us
;
;
;=====
    cmp a, 0 ; +5 instrucción insertada para 5 ciclos

bdloop:
    nop ; +4
    nop ; +4
    cmp a, [0] ; +7

    dec a ; +4
    jnz bdloop ; +5

```

ENDM

MACRO DELAY10MS

```

;=====
; Retarda10ms - aproximadamente
; no posee proteccion XPAGE
;=====

```

```

    mov x,051                                ;Llamada a contador
    mov a,FFh

```

```

.Delay10ms_msB:
iowr wdt                                    ; marca el guardian
.Delay10ms_lsB:
    dec A
    jnc .Delay10ms_lsB
    dec X
    jnc .Delay10ms_msB

```

ENDM

MACRO FLOW\_OFF

```

    iord P1_data
    or a,nCTS
    iowr P1_data

    mov a, FLOW_CTL_OFF
    or [fwcr], a

```

ENDM

MACRO FLOW\_ON

```

    iord P1_data
    and a,~nCTS
    iowr P1_data

    mov a, ~FLOW_CTL_OFF
    and [fwcr], a

```

ENDM



```

;=====
; USB1REGS.INC
;
;=====
;usb1regs.h
;
;=====
;Declaracion General de Constantes
;
;=====

BIT0:equ01h
BIT1:equ02h
BIT2:equ04h
BIT3:equ08h
BIT4:equ10h
BIT5:equ20h
BIT6:equ40h
BIT7:equ80h

;=====
;
;
;SFR Declaracion
;
;=====

; I/O ports

P0_data:      equ 00h    ; GPIO dato port 0
P0.0:equBIT0
P0.1:equBIT1
P0.2:equBIT2
P0.3:equBIT3
P0.4:equBIT4
P0.5:equBIT5
P0.6:equBIT6
P0.7:equBIT7

P1_data:      equ 01h    ; GPIO dato port 1
P1.0:equBIT0
P1.1:equBIT1
P1.2:equBIT2
P1.3:equBIT3
P1.4:equBIT4
P1.5:equBIT5
P1.6:equBIT6
P1.7:equBIT7

P0_interrupt: equ 04h    ; Habilita la interrupción port 0
IE0.0:equBIT0
IE0.1:equBIT1
IE0.2:equBIT2
IE0.3:equBIT3
IE0.4:equBIT4
IE0.5:equBIT5
IE0.6:equBIT6
IE0.7:equBIT7

```

```

P1_interrupt: equ 05h ; Habilita la interrupcion en port 1
IE1.0:equBIT0
IE1.1:equBIT1
IE1.2:equBIT2
IE1.3:equBIT3
IE1.4:equBIT4
IE1.5:equBIT5
IE1.6:equBIT6
IE1.7:equBIT7

P0_pullup: equ08h ; resistencias en Pullup para port 0
PULL0.0:equBIT0
PULL0.1:equBIT1
PULL0.2:equBIT2
PULL0.3:equBIT3
PULL0.4:equBIT4
PULL0.5:equBIT5
PULL0.6:equBIT6
PULL0.7:equBIT7

P1_pullup: equ 09h ; resistencias en Pullup para port 1
PULL1.0:equBIT0
PULL1.1:equBIT1
PULL1.2:equBIT2
PULL1.3:equBIT3
PULL1.4:equBIT4
PULL1.5:equBIT5
PULL1.6:equBIT6
PULL1.7:equBIT7

; USB ports

EP0_TX_Config: equ 10h ; EP0 configura para transmitir
EP0_TX_COUNT:equ0fh
ERR_B: equBIT4
STALL_B:equBIT5
DATA10_B:equBIT6
INEN_B: equBIT7

EP1_TX_Config: equ 11h ; EP1 configuración para transmitir
EP1_TX_COUNT:equ0fh
EP1EN_B:equBIT4
STALL_B:equBIT5 ;declara EP0 listo
DATA10_B:equBIT6 ; declara EP0 listo
INEN_B: equBIT7 ; declara EP0 listo

USB_DA: equ 12h ; USB asigna registro del host
ADR0_B: equBIT0
ADR1_B: equBIT1
ADR2_B: equBIT2
ADR3_B: equBIT3
ADR4_B: equBIT4
ADR5_B: equBIT5
ADR6_B: equBIT6

USB_SCR: equ 13h ; USB estatus y control de registro
BUSACT_B:equBIT0

```

FORCEK\_B:equBIT1  
 FORCEJ\_B:equBIT2  
 STATOUTS\_B:equBIT3  
 ENOUTS\_B:equBIT4

EP0\_RX\_Status: equ 14h ; USB EP0 estatus de recepción  
 SETUP\_B:equBIT0  
 OUT\_B:equBIT1  
 IN\_B:equBIT2  
 TOGGLE\_B:equBIT3  
 EP0\_RX\_COUNT:equf0h

; control ports

gier: equ 20h ; habilita interrupción global  
 I28IE\_B:equBIT1  
 I024IE\_B:equBIT2  
 EP0IE\_B:equBIT3  
 EP1IE\_B:equBIT4  
 GPIOIE\_B:equBIT6  
 CEXTIE\_B:equBIT7

wdt: equ 21h ; limpia el guardián  
 timer: equ 23h ; libera el reloj  
 cext:equ22h

;Dirección de registro.

; GPIO Isink registro

P0\_sink0: equ 30h;Port 0 Bit 0  
 P0\_sink1: equ 31h  
 P0\_sink2: equ 32h  
 P0\_sink3: equ 33h  
 P0\_sink4: equ 34h  
 P0\_sink5: equ 35h  
 P0\_sink6: equ 36h  
 P0\_sink7: equ 37h

;Port 0 Bit 7

P1\_sink0: equ 38h  
 P1\_sink1: equ 39h  
 P1\_sink2: equ 3Ah  
 P1\_sink3: equ 3Bh  
 P1\_sink4: equ 3Ch  
 P1\_sink5: equ 3Dh  
 P1\_sink6: equ 3Eh  
 P1\_sink7: equ 3Fh

;Port 1 Bit 0

;Port 0 Bit 7

; control port

scr:equFFh  
 RUN\_B:equBIT0  
 SUSPEND\_B:equBIT3  
 POR\_B:equBIT4  
 USBR\_B:equBIT5  
 WDR\_B:equBIT6  
 RESET\_MASK:equ70h

;Estatus y Control de registro

=====

```
; USBCODE.ASM
;=====
;   MODULE Decode Request
;
;=====

POLLING_INTERVAL: EQU 0Ah      ; Polling intervalo en milisegundos


DecodeRequest:
    mov a, [bmRequestType]
    and a, 60h
    jnz ClassRequest

;=====
; Decodificador estandar
;=====

StandardRequest:
    mov a, [bRequest]
    asl a                      ;multiplica indice por 2 para la tabla jump
    jnc SendStall              ; chequea el rango

    jacc standard_request_table

XPAGEOFF
standard_request_table:
    jmp GetStatus
    jmp ClearFeature
    jmp SendStall              ;reservado
    jmp SetFeature
    jmp SendStall              ;reservado
    jmp SetAddress
    jmp GetDescriptor
    jmp SetDescriptor
    jmp GetConfiguration
    jmp SetConfiguration
    jmp GetInterface
    jmp SetInterface
end_standard_request_table:

;=====
; Decodificador HID Class
;=====

ClassRequest:
    cmp a, 20h
    jnz VendorRequest

    mov a, [bRequest]
    asl a                      ; multiplica por 2 el indice
    jnc SendStall

    jacc HID_request_table
    db 00,00,00,00,00,00,00,00,00,00
```

```

XPAGEOFF
HID_request_table:
    jmp SendStall                                ;reservado
    jmp GetReport
    jmp GetIdle
    jmp GetProtocol
    jmp SendStall                                ;reservado
    jmp SendStall                                ;reservado
    jmp SendStall                                ;reservado
    jmp SendStall                                ;reservado
    jmp SendStall                                ;reservado
    jmp SetReport
    jmp SetIdle
    jmp SetProtocol
end_HID_request_table:
XPAGEON

```

; Tabla nop para limpiar estatus

```

nop
nop
nop
nop
nop
nop
nop
nop
nop
nop
nop
nop
nop
nop
nop
nop
nop
nop

```

```

;=====
; Decodificador Vendor
;=====

```

VendorRequest:

```

    jmp SendStall

```

```

;=====
;   Handle Request - Get Status
;=====
.

```

GetStatus:

```

    mov A, 2                                ; envia dos byte
    mov [data_count], A

```

```

    mov a, [bmRequestType]
    cmp a, DEVICE_TO_HOST
    jz GetDeviceStatus

```

```

    cmp a, ENDPOINT_TO_HOST
    jz GetEndpointStatus

```

```

    cmp a, INTERFACE_TO_HOST
    jnz SendStall          ;80, 81, 82 solo es valido en tipos
;=====
; Recipienet = Interface
;=====
;La interface es de 16 bit (dos bytes); cero para la raiz de bytes.

GetInterfaceStatus:
    mov A, (get_interface_status_table - control_read_table)
    jmp SendRomData        ; envia el estatus de la interface a host
;=====
; Recipient = Device
;=====

GetDeviceStatus:
    mov A, (get_dev_status_table - control_read_table)
    jmp SendRomData        ; envia registro de datos
;=====
; Recipiente = Endpoint
;=====

GetEndpointStatus:
    iord EP1_TX_Config
    and a, STALL_B
    jz .Send
    mov a, (stalled - get_endpoint_status_table) ;EP1

.Send:
    add A, (get_endpoint_status_table - control_read_table)
    jmp SendRomData        ; envia el estatus de endpoint a host
;=====; Encabezado –
Limpia a futuro
;=====;

ClearFeature:
    mov a, [bmRequestType]
    cmp a, HOST_TO_DEVICE
    jz ClearRemoteWakeup

    cmp a, HOST_TO_ENDPOINT
    jnz SendStall          ;00, 02 solo si valida typo bmRequest
;=====
; Recipiente = Endpoint

ClearEndpointStall:
    mov A, [wValue]
    cmp A, EP1_STALLED
    jnz SendStall

    call NoDataControl

    iord EP1_TX_Config
    and A, ~(STALL_B + DATA10_B) ; limpia 0/1 bits

```

```

iowr EP1_TX_Config

iord USB_SCR                                ; Set up EP1 a NAK INs
and A,0EFh                                ; entrada en ACKed con nuevos datos, no limpiar
                                           ; USBACT BIT
                                           ; nunca limpie el USBACT bit.
iowr USB_SCR                                ; listo.

ret

;=====
; Recipiente = Device
;=====

ClearRemoteWakeup:
    jmp SendStall

    call NoDataControl

    mov A, DISABLE_REMOTE_WAKEUP           ; desabilita wakeup
    mov [remote_wakeup_status], A

    ret

;=====
;   Handle Request - Set Feature
;=====
SetFeature:
    mov a, [bmRequestType]
    cmp a, HOST_TO_DEVICE
    jz SetRemoteWakeup

    cmp a, HOST_TO_ENDPOINT
    jnz SendStall

;=====
; Recipiente = Endpoint
;=====

SetEndpointStall:
    mov A, [wValue]
    cmp A, EP1_STALLED                     ; test valida las características
    jnz SendStall
    call NoDataControl                     ; peticion en host

    mov A, 30h                             ; un endpoint de parada
    iowr EP1_TX_Config
    ret

;=====
; Recipiente = Device
;=====

SetRemoteWakeup:
    mov A, [wValue]
    cmp A, REMOTE_WAKEUP                   ; test valida las características
    jnz SendStall

```

```

call NoDataControl                                ; peticion en host

mov A, ENABLE_REMOTE_WAKEUP                      ; habilita wakeup
mov [remote_wakeup_status], A
ret

;=====
;   Peticion de direccion
;=====

SetAddress:
mov a, [bmRequestType]
cmp a, HOST_TO_DEVICE
jnz SendStall

mov a, [wValue]                                ;chequea rango de direcciones
and a, 80h                                      ; rango valido de 0 a 7F
jnz SendStall ;

call NoDataControl

.wait:
iowr wdt
iord EP0_TX_Config                                ; espera por ACK.
and A, 80h                                         ; otra espera ACK si cambia la direccion
jnz .wait                                         ; envia ACK a la antigua direccion.

mov A, [wValue]                                ; escribe en la nueva direccion del USB
iowr USB_DA ;
ret

;=====
;   Peticion- Get Descriptor
;=====

GetDescriptor:
mov a, [bmRequestType]
cmp a, DEVICE_TO_HOST
jnz GetClassDescriptor

mov A, [wValueHi]
asl a                                             ; multiplica el index por 2
cmp a, (end_desc_type_table - desc_type_table + 1)
jnc SendStall                                   ; valida el descriptor

jacc desc_type_table

GetClassDescriptor:
mov a, [bmRequestType]
cmp a, INTERFACE_TO_HOST
jnz SendStall                                   ; valida solo si hay interfase

mov A, [wValueHi]                                ; llama a los tipos de descriptores
and a, dfh
asl a
cmp a, (end_class_desc_type_table - class_desc_type_table + 1)
jnc SendStall

```



```

jacc class_desc_type_table

XPAGEOFF
desc_type_table:
    jmp SendStall ;sin descriptor index 0
    jmp GetDeviceDescriptor
    jmp GetConfigurationDescriptor
    jmp GetStringDescriptor
end_desc_type_table:

class_desc_type_table:
    jmp SendStall ; sin descriptor index 0
    jmp GetHIDDescriptor
    jmp GetReportDescriptor
    jmp GetPhysicalDescriptor
end_class_desc_type_table:
XPAGEON

;=====
; Get Descriptor = Device
;=====

GetDeviceDescriptor:
    mov A, (end_device_desc_table - device_desc_table)
    mov [data_count], A ; graba el largo del descriptor

    mov A, (device_desc_table - control_read_table)
    jmp SendRomData ; pasa el puntero a los datos

;=====
; Get Descriptor = Configuracion
;=====

GetConfigurationDescriptor:
    mov A, (end_config_desc_table - config_desc_table)
    mov [data_count], A ; graba el largo del descriptor

    mov A, (config_desc_table - control_read_table)
    jmp SendRomData ; pasa el puntero al descriptor datos

;=====
; Get Descriptor = String
;=====

GetStringDescriptor:
    mov A, [wValue] ; genera index descriptor string
    cmp a, (end_string_length_table - string_length_table + 1)
    jnc SendStall ; chequea rango del index

    index string_length_table
    mov [data_count], A

    mov A, [wValue]
    index string_offset_table
    jmp SendRomData

```

```

XPAGEOFF
string_length_table:
    db (USBStringDescription1 - USBStringLanguageDescription)
    db (USBStringDescription2 - USBStringDescription1)
    db (USBStringDescription3 - USBStringDescription2)
    db (USBStringDescription4 - USBStringDescription3)
    db (USBStringDescription5 - USBStringDescription4)
    db (USBStringEnd - USBStringDescription5)
end_string_length_table:

string_offset_table:
    db (USBStringLanguageDescription - control_read_table)
    db (USBStringDescription1 - control_read_table)
    db (USBStringDescription2 - control_read_table)
    db (USBStringDescription3 - control_read_table)
    db (USBStringDescription4 - control_read_table)
    db (USBStringDescription5 - control_read_table)
end_string_offset_table:
XPAGEON

;=====
; Get Descriptor = Report
;=====

GetReportDescriptor:
    mov A, (end_hid_report_desc_table - hid_report_desc_table)
    mov [data_count], A                                ; graba el largo del descriptor

    mov A, (hid_report_desc_table - control_read_table)
    jmp SendRomData                                    ; envia descriptor a host

;=====
; Get Descriptor = HID
;=====

GetHIDDescriptor:
    mov A, (Endpoint_Descriptor - Class_Descriptor)
    mov [data_count], A                                ; graba largo del descriptor

    mov A, (Class_Descriptor - control_read_table)
    jmp SendRomData                                    ; envia descriptor a host

;=====
; Get Descriptor = Fisico
;=====

GetPhysicalDescriptor:
    jmp SendStall                                      ;no soportado por ventanas

;=====
;   Peticion - Set Descriptor
;=====

SetDescriptor:
    jmp SendStall                                      ;no soportado por ventanas

```

```

;=====
;   Handle Request - Get Configuration
;=====

GetConfiguration:
    mov a, [bmRequestType]
    cmp a, DEVICE_TO_HOST
    jnz SendStall

    mov A, 1                                ; envia un byte
    mov [data_count], A

    mov A, (get_configuration_status_table - control_read_table)
    add A, [configuration_status]           ; consigue la configuracion correcta

    jmp SendRomData                        ; envia configuracion al host

;=====
;   Handle Request - Get Configuration
;=====

SetConfiguration:
    mov a, [bmRequestType]
    and a, HOST_TO_DEVICE
    jnz SendStall

    call NoDataControl

    mov A, [wValue]
    mov [configuration_status], A           ; almacena la configuracion del byte

    iord EP1_TX_Config                     ; limpia el dato 0/1 bit
    and A, ~(DATA10_B + STALL_B)
    iowr EP1_TX_Config

    mov A, [configuration_status]
    and a, fffh
    jnz ConfigureDevice

UnconfigureDevice:
    iord EP1_TX_Config
    and A, ~INEN_B                         ; desabilita endpoint 1
    iowr EP1_TX_Config

    mov A, [interrupt_mask]                ; desabilita interrupcion endpoint
    and A, ~EP1IE_B
    mov [interrupt_mask], A

    ret

ConfigureDevice:
    iord EP1_TX_Config                     ; NAK IN
    and A, 7Fh                             ; lee un endpoint
    or A, 10h                             ; habilita el endpoint
    iowr EP1_TX_Config

    mov A, [interrupt_mask]                ; habilita interrupciones gpio, ep1, 1ms 128us
    or A, ENUMERATED_MASK

```

```

mov [interrupt_mask], A

iord USB_SCR                                ; NAK IN
and A,0EFh                                ; listo el endpoint, nunca limpia USBACT bit
iowr USB_SCR

ret

;=====
;   Peticion - Get Interface
;=====

GetInterface:
mov a, [bmRequestType]
cmp a, INTERFACE_TO_HOST
jnz SendStall

mov A, [wIndex]                            ;busco interface
cmp A, 0                                    ;solo la interface definida 0
jnz SendStall

mov A, 1                                    ;envia un bit
mov [data_count], A

mov A, (get_interface_table - control_read_table)
jmp SendRomData                            ;envia alternando entre host e interface

;=====
;   Peticion - Set Interface
;=====

SetInterface:
mov a, [bmRequestType]
cmp a, DEVICE_TO_HOST
jnz SendStall

mov a,[wIndex]                            ; busco interface ?
cmp a,0                                    ; solo interface definida 0
jnz SendStall

mov a,[wValue]                            ; busco interface ?
cmp a, 0                                    ; solo interface definida 0
jnz SendStall

jmp NoDataControl

;=====
;   Handle Request - Get Report
;=====

GetReport:
mov a, [bmRequestType]
cmp a, CLASS + INTERFACE_TO_HOST
jnz SendStall ;Send a stall if this is not valid request

;no necesita ayuda del reporte

jmp SendStall

```

```

;=====; Peticion - Get
Idle
;=====;
;Este HID Class request. Solo soporta informacion de Microsoft y es opcional instalarlo

GetIdle:
    jmp SendStall

;=====
; Peticion - Get Protocol
;=====

GetProtocol:
    mov a, [bmRequestType]
    cmp a, CLASS + INTERFACE_TO_HOST
    jnz SendStall

    mov A, 1 ; envia un bit
    mov [data_count], A

    mov A, (get_protocol_status_table - control_read_table)
    add A, [protocol_status] ; captura la configuracion correcta
    jmp SendRomData

;=====
; Peticion - Set Report
;=====

SetReport:
    mov a, [bmRequestType]
    cmp a, CLASS + HOST_TO_INTERFACE
    jnz SendStall

;DETERMINA LA IDENTIFICACION DEL REPORTE

    mov a, [wValue] ;Lee el reporte correcto y lo guarda para despues.
    mov [current_reportid], a

    mov a, [wLength] ;Lee el descriptor y lo guarda para despues.
    mov [data_count], a ;Cuenta cuantos bit.

;LIMPIA EL CONJUNTO DE BIT
    mov a, 00h
    iowr EP0_RX_Status ;

    mov a, [current_reportid]
    cmp a, RID_TRANSMIT ; Es transmitido el dato serial?
    jz hid_report_incoming ; si, este es e conjunto

    cmp a, RID_COMMAND ; cambia la tasa de transmicion?
    jnz await_outs ; no, continue de lo contrario
                    ; este es el nuevo set de datos

hid_report_incoming:
    mov a, TRANSMITTING_MASK

```

```

    mov [interrupt_mask],a

    mov a, 0                                ; desabilita las interrupciones
iowr P0_interrupt

IFDEF SOFTWARE_FLOW_CONTROL

    mov a,XOFF                                ;Llama al registro serial con XOFF
    mov [serial_tx_reg],a                    ;y lo transmite
    call SerialTransmitByte                  ; XOFF
ENDIF

    FLOW_OFF                                ; ensiende el flujo (hardware)

    mov a,FIRSTBYTE_FLG_MASK                ;indica que recibio el primer conjunto
    or [fwcr], a                            ;la secuencia es por RID

await_outs:

    iord USB_SCR                            ;
    and a, ~STATOUTS_B                      ;
    or A, ENOUTS_B
    iowr USB_SCR

    ret

;=====
;   Peticion - Set Idle
;=====

SetIdle:
    jmp SendStall

;=====
;   Peticion - Get Protocol
;=====

SetProtocol:
    mov a, [bmRequestType]
    cmp a, CLASS + HOST_TO_INTERFACE
    jnz SendStall

    mov A, [wValue]
    mov [protocol_status], A                ; escribe el nuevo protocolo

    jmp NoDataControl

;=====
;   ***** USB menu, libreria, rutinas *****
;=====

;=====
; Funcion - Send ROM Data
;=====

```

```

SendRomData:
    mov [data_start], A

    call GetDescriptorLength
    call ControlRead

    ret

;=====
; Function Send Stall
;=====
SendStall:
    mov A, INEN_B + STALL_B
    iowr EP0_TX_Config
    ret

;=====
; Funcion Get Descriptor Length
;=====

GetDescriptorLength:
    mov A, [wLengthHi]
    cmp A, 0                                ; confirma si el bit alto es 0
    jnz UseActualLength                    ; no requiere un largo de 256b

    mov A, [wLength]                        ; prueba si el bit bajo es 0 otra vez
    cmp A, 0
    jz UseActualLength                      ; pide los datos

    cmp A, [data_count]                    ; compara los datos
    jnc UseActualLength

    mov [data_count], A

UseActualLength:
    ret

;=====
; Funcion - NoDataControl
;=====

NoDataControl:
    iord EP0_RX_Status
    and a, ~SETUP_B
    iowr EP0_RX_Status

    mov A, INEN_B + DATA10_B              ; transfiere el registro a data1
    iowr EP0_TX_Config                      ; y transfiere byte 0

    ret

```

```

;=====
; Function - Control Read
;=====

ControlRead:
    mov A, 00h                                ; limpia data 0/1 bit, A veces empieza con etiqueta
                                                ; DATA0

    mov [endp0_data_toggle], A

ControlReadDataStage:
                                                ; Solo 8 bytes. (8 bytes/packet)
                                                ; para los paquetes de N incluye este código

    push X
    mov X, 00h
    mov A, 00h

    mov [loop_counter], A                    ; contador de ciclos a 00
    iowr EP0_RX_Status                       ; limpia el conjunto de bits

    iord EP0_RX_Status                        ; chequea estatus rx
    and A, 01h
    jnz ControlReadStatusStage

    iord USB_SCR
    or a, STATOUTS_B                         ; Controla automáticamente la salida.
    iowr USB_SCR

    mov A, [data_count]
    cmp A, 00h
    jz DmaLoadDone

DmaLoadLoop:
                                                ; llamada cíclica de datos en el buffer

    mov A, [data_start]
    index control_read_table
    mov [X + EP0_FIFO], A                    ; llama dma en buffer

    inc [data_start]
    inc X
    inc [loop_counter]

    dec [data_count]                         ; sale del descriptor
    jz DmaLoadDone                           ; verifica si perdió algún bit.

    mov A, [loop_counter]                    ; envía 8 bytes
    cmp A, 08h
    jnz DmaLoadLoop

DmaLoadDone:
    iord EP0_RX_Status
    and A, 01h
    jnz ControlReadStatusStage

```



```

mov A, [endp0_data_toggle]
xor A, DATA10_B
mov [endp0_data_toggle], A

or A, 80h
or A, [loop_counter]
iowr EP0_TX_Config

ControlReadStatusStage:
    pop X
    ret

;=====
; Funcion - Control Write
;=====

ControlWrite:

ControlWriteRamDataStage:
    push X

    mov a, [current_reportid]
    cmp a, RID_COMMAND                ; Cambia la tasa de transferencia?
    jnz ridtransmit

    mov a, [EP0_FIFO+1]                ; comando put en A
    call handle_rid_command
    jmp cwhandshake

ridtransmit:
    iord EP0_RX_Status                ; Cuenta los bit de salida
    rrc
    rrc
    rrc
    rrc
    and a, 0fh                        ; enmascara los bit
    sub a, 02h                        ; remueve 2 bit de las lista
    mov [loop_counter], A             ;llama el contador a 00. (con incremento de 8)

    mov X, 00h                        ;lee el conjunto USB FIFO
    .
    mov a, [fwcr]
    and a, FIRSTBYTE_FLG_MASK
    jz DmaWriteRamLoop                ;RID responde al informe

    mov a, [fwcr]
    and a, ~FIRSTBYTE_FLG_MASK        ; muestra el primer byte de a mascara
    or a, BUSY_TRANSMITTING
    mov [fwcr], a

    mov a, [X+EP0_FIFO]                ; reporte serial tx ?
    cmp a, RID_TRANSMIT
    jnz cwhandshake

    dec [data_count]                  ; decrementa los bit del reporte en HID
    dec [loop_counter]                ; decrementa el contador de paquetes
    inc X                             ; 2nd byte en la primera salida tx

```

```

    mov a, [X+EP0_FIFO]
    mov [transmit_count], a

    dec [data_count]                ; decremento en el contador HID
    dec [loop_counter]             ; decremento en el contador de paquetes
    inc X                          ; puntero en el proximo byte tx

    mov a, INTDISABLE_MASK
    iowr gier                      ; serial tx

DmaWriteRamLoop:
    mov a, [transmit_count]
    cmp a, 0
    jz DmaWriteRam_next

    iowr wdt                      ; envia al guardian el siguiente byte
    mov A, [X + EP0_FIFO]         ; envia 1-byte serial

    mov [serial_tx_reg], A        ; ejecuta un byte a la vez
    call SerialTransmitByte
    dec [transmit_count]          ; decremento en el contador tx

DmaWriteRam_next:
    inc X
    dec [data_count]
    dec [loop_counter]
    jnz DmaWriteRamLoop

DmaWriteRamDone:
    mov A, [data_count]
    cmp A, 00h
    jnz cwenableouts

cwhandshake:
    call NoDataControl
cwenableouts:
    iord USB_SCR
    or A, ENOUTS_B
    iowr USB_SCR

cwexit:
    pop X
    ret

;=====
; Tablas ROM
;=====
;

XPAGEOFF

control_read_table:
device_desc_table:
    db (end_device_desc_table - device_desc_table) ; Descriptor largo (18 bytes)
    db 01h                                           ; Descriptor tipo (Device)
    db 10h,01h                                       ; Con USB 1.1

```

```

db 00h ; Class code (0)
db 00h ; Subclass code (0)
db 00h ; Protocol (No specific protocol)
db 08h ; Max. packet size for EP0 (8 bytes)
DWL PL-2303_VID ; vendor ID
DWL PL-2303_PID ; product ID
DWL PL-2303_VER ; firmware version (BCD)
db 01h
db 02h
db 00h
db 01h
end_device_desc_table:

```

```

config_desc_table:
db (Interface_Descriptor - config_desc_table) ; Descriptor largo (9 bytes)
db 02h ; Descriptor tipo (Configuracion)
db (end_config_desc_table - config_desc_table),00h
db 01h ; Interface suportada (1)
db 01h ; Configuracion valor (1)
db 04h
db 80h ; Configuracion (Bus poder)
db 32h ; Max consumo MUST be < 100mA

```

```

Interface_Descriptor:
db (Class_Descriptor - Interface_Descriptor) ; Descriptor largo (9 bytes)
db 04h ; Descriptor tipo (Interface)
db 00h ; Numero de interface (0)
db 00h ;
db 01h ; Numero de interface endpoints (1)
db 03h ; Class code ()
db 00h ; Subclass code ()
db 00h ; Protocol code ()
db 05h ; Index of string()

```

```

Class_Descriptor:
db (Endpoint_Descriptor - Class_Descriptor) ; Descriptor largo (9 bytes)
db 21h ; Descriptor tipo (HID)
db 00h,01h ; HID class release number (1.00)
db 00h ;Codigo de pais (None)
db 01h ; # de HID class descriptor (1)
db 22h ; Reporet descriptor (HID)
dwl (end_hid_report_desc_table - hid_report_desc_table)
; Total largo del reporte(2 bytes)

```

```

Endpoint_Descriptor:
db (end_config_desc_table - Endpoint_Descriptor) ; Descriptor largo (7 bytes)
db 05h ; Descriptor tipo (Endpoint)
db 81h ; Direccion codificada (Responde a IN, 1 endpoint)
db 03h ; Endpoint atributo (Interrupcion transferencia)
db EP1_REPORT_LENGTH+1,00h ; Maximo packet (report len + id = 8 bytes)
db POLLING_INTERVAL

```

end\_config\_desc\_table:

; Este es parte del anexo A12 del HID Manual

```

hid_report_desc_table:
    db 06h, 00h, ffh                ; USAGE_PAGE (Vendor Defined Page 1)

    db 09h, 01h                      ; USAGE (Vendor Usage 1)
    db a1h, 01h                      ; COLLECTION (Application)

    db 85h, RID_RECEIVE              ; REPORT_ID (1) - streaming UART rx data
    db 9h, 1h                        ; USAGE (Vendor Usage 1)
    db 15h, 0h                       ; LOGICAL_MINIMUM (0)
    db 26h, ffh, 0h                  ; LOGICAL_MAXIMUM (255)
    db 75h, 8h                       ; REPORT_SIZE (8) = bits
    db 95h, EP1_REPORT_LENGTH ; REPORT_COUNT (7) = 1 report id byte + 7 data bytes
    db 81h, 6h                       ; INPUT (Data,Var,Rel)
    db 9h, 1h                        ; USAGE (Vendor Usage 1)
    db b1h, 6h                       ; FEATURE (Data,Var,Rel)

    db 85h, RID_COMMAND              ; REPORT_ID (3) Command (setreport)
    db 09h, 01h                      ; USAGE (Vendor Usage 1)
    db 15h, 00h                       ; LOGICAL_MINIMUM (0)
    db 26h, 8h, 00h                  ; LOGICAL_MAXIMUM (255)
    db 75h, 08h                      ; REPORT_SIZE (8) bits
    db 95h, 01h                      ; REPORT_COUNT (1) byte
    db 09h, 01h                      ; USAGE (Vendor Usage 1)
    db b1h, 06h                      ; FEATURE (Data,Var,Rel)

    db 85h, RID_TRANSMIT             ; REPORT_ID (2) serial transmit these bytes (SetReport)
    db 09h, 03h                      ; USAGE (Vendor Usage 1)
    db 96h, 07h, 00h                ; REPORT_COUNT (7) 1 count byte, 6 data
    db b1h, 06h                      ; FEATURE (Data,Var,Rel)

    db c0h                          ; END_COLLECTION
end_hid_report_desc_table:

;=====

get_dev_status_table:
    db 00h, 00h    ; remote wakeup disabled, bus powered
wakeup_enabled:
    db 02h, 00h    ; remote wakeup enabled, bus powered

get_interface_status_table:
    db 00h, 00h    ; always return both bytes zero

get_endpoint_status_table:
    db 00h, 00h    ; not stalled
stalled:
    db 01h, 00h    ; stalled

get_configuration_status_table:
    db 00h    ; not configured
    db 01h    ; configured

get_protocol_status_table:
    db 00h    ; boot protocol
    db 01h    ; report protocol

get_interface_table:

```

```

db 00h ; no alternate setting

;=====

USBStringLanguageDescription:
db (USBStringDescription1 - USBStringLanguageDescription)
db 03h ; Type (3=string)
db 09h ; Language: English
db 04h ; Sub-language: Default

; string 1
USBStringDescription1: ; IManufacturerName
db (USBStringDescription2 - USBStringDescription1)
db 03h ; Type (3=string)
dsu "Moto" ;

; string 2
USBStringDescription2: ; IProduct
db (USBStringDescription3 - USBStringDescription2)
db 03h ; Type (3=string)
dsu "MO1002" ;

;string 3
USBStringDescription3: ; serial number
; If a SN is used, this must be unique
; for every device or the device may
; not enumerate properly

; string 4
USBStringDescription4: ; configuration string descriptor
db (USBStringDescription5 - USBStringDescription4)
db 03h ; Type (3=string)
dsu "2400" ;

;string 5
USBStringDescription5: ; interface string descriptor
db (USBStringEnd - USBStringDescription5)
db 03h ; Type (3=string)
dsu "USB ART" ;

USBStringEnd:
XPAGEON

' =====
'
' File..... DS2760TC_5.BPE
' Purpose... Thermocouple temperature measurement using the DS2760
' Author.... RUDI RADRIGAN.
' E-mail.... rradriga@udec.cl
' Started...
' Updated... 10 dec 2006
'
' { $STAMP BS2px, KTablePos.BPE, volmeter.BPE }
' { $PBASIC 2.5 }
'
' =====

```

```
' -----[ Program Description ]-----
'
' This program lets a BS2p or BS2pe read [positive] temperature from the
' Parallax DS2760 thermocouple module. User input of thermocouple type
' (K, J, or T) and temperature display is via the Debug terminal window.

' -----[ Revision History ]-----

' -----[ I/O Definitions ]-----

OW      VAR   Nib      ' 1-Wire buss pin
s      VAR   Nib      ' stop

' -----[ Constants ]-----

ReadNet  CON   $33      ' read OW net address
SkipNet  CON   $CC      ' skip OW net address
RdReg    CON   $69      ' read register
type     CON   1

' -----[ Variables ]-----

idx      VAR   Nib      ' loop counter
char     VAR   Byte     ' display byte/char
vIn      VAR   Word     ' in millivolts
tmpCJ    VAR   Word     ' device temp in C
tCuV     VAR   Word     ' thermocouple millivolts
sign     VAR   Word     ' TC sign bit
cjComp   VAR   Word     ' temp compensation
tempC    VAR   Word     ' temp in Celsius
tempF    VAR   Word     ' temp in Fahrenheit
tblLo    VAR   Word     ' table pointers
tblHi    VAR   Word
eePntr   VAR   Word
testVal  VAR   Word     ' test value from table
error    VAR   Bit      ' 1 = out of range

' -----[ EEPROM Data ]-----

' -----[ Initialization ]-----

Check_Device:
FOR OW =0 TO 4
    OWOUT OW, %0001, [ReadNet]      ' get serial number
    OWIN  OW, %0010, [SPSTR 8]      ' store in SPRAM
    GET idx, char                    ' read device type
    IF (char <> $30) THEN            ' if not $30, wrong device
        DEBUG "No DS2760 found."
    ENDIF
NEXT                                ' stop program
```

Menu:

STORE type ' point READ to table

' ----[ Program Code ]-----

Main:

OW= 0

DO

Ow=OW

GOSUB Read\_TC\_Volts

' read Seebeck voltage

GOSUB Read\_CJ\_Temp

' read cold junction temp

READ (tmpCJ \* 2), Word cjComp

' get compensation voltage

' combine cjComp and tCuV

,

IF sign THEN

' TC below cold junction

IF (tCuV < cjComp) THEN

cjComp = cjComp - tCuV

ELSE

cjComp = 0

' limit to 0C

ENDIF

ELSE

' TC above cold junction

cjComp = cjComp + tCuV

ENDIF

LOOKUP type, [1023, 1023, 400], tblHi

' set high end of search

GOSUB TC\_Lookup

' reverse lookup of table

IF (error = 0) THEN

'DEBUG CRSRXY, 0, OW,

' "Temp °C..... ", SDEC tempC, CLREOL

DEBUG DEC3 tempC,"A"

DEBUG DEC2 tempF,"B"

ELSE

'DEBUG CRSRXY, 0, OW,

' "Temp °C..... Out of Range", CLREOL

DEBUG "xx","A"

ENDIF

OW=OW+1

IF OW >4 THEN END'OW =0

LOOP

END

' ----[ Subroutines ]-----

' Reads device input voltage (Vin pin)

' -- mV in millivolts (max reading is 4.75 volts)

Read\_Vin:

OWOUT OW, %0001, [SkipNet, RdReg, \$0C]

```

OWIN OW, %0010, [vIn.BYTE1, vIn.BYTE0]
IF (vIn.BIT15) THEN
    vIn = 0
ELSE
    vIn = vIn >> 5 * / $4E1
ENDIF
RETURN

' Reads current register to get TC voltage
' -- each raw bit = 15.625 uV
' -- tCuV in microvolts

Read_TC_Volts:
OWOUT OW, %0001, [SkipNet, RdReg, $0E]
OWIN OW, %0010, [tCuV.BYTE1, tCuV.BYTE0]
sign = tCuV.BIT15
tCuV = tCuV >> 3
IF sign THEN
    tCuV = tCuV | $F000
ENDIF
tCuV = ABS tCuV * / 4000
RETURN

' Reads cold junction (device) temperature
' -- each raw bit = 0.125 degrees C
' -- returns tmpCJ in whole degrees C

Read_CJ_Temp:
OWOUT OW, %0001, [SkipNet, RdReg, $18]
OWIN OW, %0010, [tmpCJ.BYTE1, tmpCJ.BYTE0]
IF (tmpCJ.BIT15) THEN
    tmpCJ = 0
ELSE
    tmpCJ = tmpCJ.HIGHBYTE
ENDIF

OWOUT OW, %0001, [SkipNet, RdReg, $19]
OWIN OW, %0010, [tempF.BYTE1, tempF.BYTE0]
IF (tempF.BIT15) THEN
    tempF = 0
ELSE
    tempF = tempF.HIGHBYTE
ENDIF
RETURN

' Search currently selected TC table for nearest entry
' -- uses modified binary algorithm to find cjComp
' -- high end of search set before calling (tblHi)
' -- successful search sets tempC

TC_Lookup:
tblLo = 0
tempC = 22

READ (tblHi * 2), Word testVal
IF (cjComp > testVal) THEN

```

' check sign  
' disallow negative

' x 4.88 millivolts

' read current register

' save sign bit  
' correct alignment

' pad 2's-compliment bits

' x 15.625 uV

' integer number

' check sign  
' disallow negative

' >> 5 x 0.125 (>> 3)

' decimal number

' check sign  
' disallow negative

' >> 5 x 0.125 (>> 3)

' low entry of table  
' default to room temp

' check max temp



```

error = 1                                     ' out of range
ELSE
DO
    eePntr = (tblLo + tblHi) / 2               ' midpoint of search span
    READ (eePntr * 2), Word testVal           ' read value from midpoint

    IF (cjComp = testVal) THEN
        EXIT                                  ' found it!
    ELSEIF (cjComp < testVal) THEN
        tblHi = eePntr                        ' search lower half
    ELSE
        tblLo = eePntr                        ' search upper half
    ENDIF

    IF ((tblHi - tblLo) < 2) THEN               ' span at minimum
        eePntr = tblLo
        EXIT
    ENDIF
LOOP
tempC = eePntr
ENDIF
RETURN

```

```

' =====
'
' File..... KTablePos.BPE
' Purpose... thermocouple data (0C reference)
' Author.... RUDI RADRIGAN.
' E-mail.... rradriga@udec.cl
' Started...
' Updated... 10 dec 2006
'
' {$STAMP BS2px}
' {$PBASIC 2.5}
' =====

' tC      +0      +1      +2      +3      +4
'          +5      +6      +7      +8      +9

```

K0000 DATA Word 00000, Word 00039, Word 00079, Word 00119, Word 00158,  
Word 00198, Word 00238, Word 00277, Word 00317, Word 00357

K0010 DATA Word 00397, Word 00437, Word 00477, Word 00517, Word 00557,  
Word 00597, Word 00637, Word 00677, Word 00718, Word 00758

K0020 DATA Word 00798, Word 00838, Word 00879, Word 00919, Word 00960,  
Word 01000, Word 01040, Word 01080, Word 01122, Word 01163

K0030 DATA Word 01203, Word 01244, Word 01284, Word 01326, Word 01366,  
Word 01407, Word 01448, Word 01489, Word 01530, Word 01570

K0040 DATA Word 01612, Word 01653, Word 01694, Word 01735, Word 01776,  
Word 01816, Word 01858, Word 01899, Word 01941, Word 01982

K0050 DATA Word 02023, Word 02064, Word 02105, Word 02146, Word 02188,  
Word 02230, Word 02270, Word 02311, Word 02354, Word 02395

K0060 DATA Word 02436, Word 02478, Word 02519, Word 02560, Word 02601,  
Word 02644, Word 02685, Word 02726, Word 02767, Word 02810

K0070 DATA Word 02850, Word 02892, Word 02934, Word 02976, Word 03016,  
Word 03059, Word 03100, Word 03141, Word 03184, Word 03225

K0080 DATA Word 03266, Word 03307, Word 03350, Word 03391, Word 03432,  
Word 03474, Word 03516, Word 03557, Word 03599, Word 03640

K0090 DATA Word 03681, Word 03722, Word 03765, Word 03806, Word 03847,  
Word 03888, Word 03931, Word 03972, Word 04012, Word 04054

K0100 DATA Word 04096, Word 04137, Word 04179, Word 04219, Word 04261,  
Word 04303, Word 04344, Word 04384, Word 04426, Word 04468

K0110 DATA Word 04509, Word 04549, Word 04591, Word 04633, Word 04674,  
Word 04714, Word 04756, Word 04796, Word 04838, Word 04878

K0120 DATA Word 04919, Word 04961, Word 05001, Word 05043, Word 05083,  
Word 05123, Word 05165, Word 05206, Word 05246, Word 05288

K0130 DATA Word 05328, Word 05368, Word 05410, Word 05450, Word 05490,  
Word 05532, Word 05572, Word 05613, Word 05652, Word 05693

K0140 DATA Word 05735, Word 05775, Word 05815, Word 05865, Word 05895,  
Word 05937, Word 05977, Word 06017, Word 06057, Word 06097

K0150 DATA Word 06137, Word 06179, Word 06219, Word 06259, Word 06299,  
Word 06339, Word 06379, Word 06419, Word 06459, Word 06500

K0160 DATA Word 06540, Word 06580, Word 06620, Word 06660, Word 06700,  
Word 06740, Word 06780, Word 06820, Word 06860, Word 06900

K0170 DATA Word 06940, Word 06980, Word 07020, Word 07059, Word 07099,  
Word 07139, Word 07179, Word 07219, Word 07259, Word 07299

K0180 DATA Word 07339, Word 07379, Word 07420, Word 07459, Word 07500,  
Word 07540, Word 07578, Word 07618, Word 07658, Word 07698

K0190 DATA Word 07738, Word 07778, Word 07819, Word 07859, Word 07899,  
Word 07939, Word 07979, Word 08019, Word 08058, Word 08099

K0200 DATA Word 08137, Word 08178, Word 08217, Word 08257, Word 08298,  
Word 08337, Word 08378, Word 08417, Word 08458, Word 08499

K0210 DATA Word 08538, Word 08579, Word 08618, Word 08659, Word 08698,  
Word 08739, Word 08778, Word 08819, Word 08859, Word 08900

K0220 DATA Word 08939, Word 08980, Word 09019, Word 09060, Word 09101,  
Word 09141, Word 09180, Word 09221, Word 09262, Word 09301

K0230 DATA Word 09343, Word 09382, Word 09423, Word 09464, Word 09503,  
Word 09544, Word 09585, Word 09625, Word 09666, Word 09707

K0240 DATA Word 09746, Word 09788, Word 09827, Word 09868, Word 09909,  
Word 09949, Word 09990, Word 10031, Word 10071, Word 10112

K0250 DATA Word 10153, Word 10194, Word 10234, Word 10275, Word 10316,

Word 10356, Word 10397, Word 10439, Word 10480, Word 10519

K0260 DATA Word 10560, Word 10602, Word 10643, Word 10683, Word 10724,  
Word 10766, Word 10807, Word 10848, Word 10888, Word 10929

K0270 DATA Word 10971, Word 11012, Word 11053, Word 11093, Word 11134,  
Word 11176, Word 11217, Word 11259, Word 11300, Word 11340

K0280 DATA Word 11381, Word 11423, Word 11464, Word 11506, Word 11547,  
Word 11587, Word 11630, Word 11670, Word 11711, Word 11753

K0290 DATA Word 11794, Word 11836, Word 11877, Word 11919, Word 11960,  
Word 12001, Word 12043, Word 12084, Word 12126, Word 12167

K0300 DATA Word 12208, Word 12250, Word 12291, Word 12333, Word 12374,  
Word 12416, Word 12457, Word 12499, Word 12539, Word 12582

K0310 DATA Word 12624, Word 12664, Word 12707, Word 12747, Word 12789,  
Word 12830, Word 12872, Word 12914, Word 12955, Word 12997

K0320 DATA Word 13039, Word 13060, Word 13122, Word 13164, Word 13205,  
Word 13247, Word 13289, Word 13330, Word 13372, Word 13414

K0330 DATA Word 13457, Word 13497, Word 13539, Word 13582, Word 13624,  
Word 13664, Word 13707, Word 13749, Word 13791, Word 13833

K0340 DATA Word 13874, Word 13916, Word 13958, Word 14000, Word 14041,  
Word 14083, Word 14125, Word 14166, Word 14208, Word 14250

K0350 DATA Word 14292, Word 14335, Word 14377, Word 14419, Word 14461,  
Word 14503, Word 14545, Word 14586, Word 14628, Word 14670

K0360 DATA Word 14712, Word 14755, Word 14797, Word 14839, Word 14881,  
Word 14923, Word 14964, Word 15006, Word 15048, Word 15090

K0370 DATA Word 15132, Word 15175, Word 15217, Word 15259, Word 15301,  
Word 15343, Word 15384, Word 15426, Word 15468, Word 15510

K0380 DATA Word 15554, Word 15596, Word 15637, Word 15679, Word 15721,  
Word 15763, Word 15805, Word 15849, Word 15891, Word 15932

K0390 DATA Word 15974, Word 16016, Word 16059, Word 16102, Word 16143,  
Word 16185, Word 16228, Word 16269, Word 16312, Word 16355

K0400 DATA Word 16396, Word 16439, Word 16481, Word 16524, Word 16565,  
Word 16608, Word 16650, Word 16693, Word 16734, Word 16777

K0410 DATA Word 16820, Word 16861, Word 16903, Word 16946, Word 16989,  
Word 17030, Word 17074, Word 17115, Word 17158, Word 17201

K0420 DATA Word 17242, Word 17285, Word 17327, Word 17370, Word 17413,  
Word 17454, Word 17496, Word 17539, Word 17582, Word 17623

K0430 DATA Word 17667, Word 17708, Word 17751, Word 17794, Word 17836,  
Word 17879, Word 17920, Word 17963, Word 18006, Word 18048

K0440 DATA Word 18091, Word 18134, Word 18176, Word 18217, Word 18260,  
Word 18303, Word 18346, Word 18388, Word 18431, Word 18472

- K0450 DATA Word 18515, Word 18557, Word 18600, Word 18643, Word 18686,  
Word 18728, Word 18771, Word 18812, Word 18856, Word 18897
- K0460 DATA Word 18940, Word 18983, Word 19025, Word 19068, Word 19111,  
Word 19153, Word 19196, Word 19239, Word 19280, Word 19324
- K0470 DATA Word 19365, Word 19408, Word 19451, Word 19493, Word 19536,  
Word 19579, Word 19621, Word 19664, Word 19707, Word 19750
- K0480 DATA Word 19792, Word 19835, Word 19876, Word 19920, Word 19961,  
Word 20004, Word 20047, Word 20089, Word 20132, Word 20175
- K0490 DATA Word 20218, Word 20260, Word 20303, Word 20346, Word 20388,  
Word 20431, Word 20474, Word 20515, Word 20559, Word 20602
- K0500 DATA Word 20643, Word 20687, Word 20730, Word 20771, Word 20815,  
Word 20856, Word 20899, Word 20943, Word 20984, Word 21027
- K0510 DATA Word 21071, Word 21112, Word 21155, Word 21199, Word 21240,  
Word 21283, Word 21326, Word 21368, Word 21411, Word 21454
- K0520 DATA Word 21497, Word 21540, Word 21582, Word 21625, Word 21668,  
Word 21710, Word 21753, Word 21795, Word 21838, Word 21881
- K0530 DATA Word 21923, Word 21966, Word 22009, Word 22051, Word 22094,  
Word 22137, Word 22178, Word 22222, Word 22265, Word 22306
- K0540 DATA Word 22350, Word 22393, Word 22434, Word 22478, Word 22521,  
Word 22562, Word 22606, Word 22649, Word 22690, Word 22734
- K0550 DATA Word 22775, Word 22818, Word 22861, Word 22903, Word 22946,  
Word 22989, Word 23032, Word 23074, Word 23117, Word 23160
- K0560 DATA Word 23202, Word 23245, Word 23288, Word 23330, Word 23373,  
Word 23416, Word 23457, Word 23501, Word 23544, Word 23585
- K0570 DATA Word 23629, Word 23670, Word 23713, Word 23757, Word 23798,  
Word 23841, Word 23884, Word 23926, Word 23969, Word 24012
- K0580 DATA Word 24054, Word 24097, Word 24140, Word 24181, Word 24225,  
Word 24266, Word 24309, Word 24353, Word 24394, Word 24437
- K0590 DATA Word 24480, Word 24523, Word 24565, Word 24608, Word 24650,  
Word 24693, Word 24735, Word 24777, Word 24820, Word 24863
- K0600 DATA Word 24905, Word 24948, Word 24990, Word 25033, Word 25075,  
Word 25118, Word 25160, Word 25203, Word 25245, Word 25288
- K0610 DATA Word 25329, Word 25373, Word 25414, Word 25457, Word 25500,  
Word 25542, Word 25585, Word 25626, Word 25670, Word 25711
- K0620 DATA Word 25755, Word 25797, Word 25840, Word 25882, Word 25924,  
Word 25967, Word 26009, Word 26052, Word 26094, Word 26136
- K0630 DATA Word 26178, Word 26221, Word 26263, Word 26306, Word 26347,  
Word 26390, Word 26432, Word 26475, Word 26516, Word 26559
- K0640 DATA Word 26602, Word 26643, Word 26687, Word 26728, Word 26771,  
Word 26814, Word 26856, Word 26897, Word 26940, Word 26983

- K0650 DATA Word 27024, Word 27067, Word 27109, Word 27152, Word 27193, Word 27236, Word 27277, Word 27320, Word 27362, Word 27405
- K0660 DATA Word 27447, Word 27489, Word 27531, Word 27574, Word 27616, Word 27658, Word 27700, Word 27742, Word 27784, Word 27826
- K0670 DATA Word 27868, Word 27911, Word 27952, Word 27995, Word 28036, Word 28079, Word 28120, Word 28163, Word 28204, Word 28246
- K0680 DATA Word 28289, Word 28332, Word 28373, Word 28416, Word 28416, Word 28457, Word 28500, Word 28583, Word 28626, Word 28667
- K0690 DATA Word 28710, Word 28752, Word 28794, Word 28835, Word 28877, Word 28919, Word 28961, Word 29003, Word 29045, Word 29087
- K0700 DATA Word 29129, Word 29170, Word 29213, Word 29254, Word 29297, Word 29338, Word 29379, Word 29422, Word 29463, Word 29506
- K0710 DATA Word 29548, Word 29589, Word 29631, Word 29673, Word 29715, Word 29757, Word 29798, Word 29840, Word 29882, Word 29923
- K0720 DATA Word 29964, Word 30007, Word 30048, Word 30089, Word 30132, Word 30173, Word 30214, Word 30257, Word 30298, Word 30341
- K0730 DATA Word 30382, Word 30423, Word 30466, Word 30507, Word 30548, Word 30589, Word 30632, Word 30673, Word 30714, Word 30757
- K0740 DATA Word 30797, Word 30839, Word 30881, Word 30922, Word 30963, Word 31006, Word 31047, Word 31088, Word 31129, Word 31172
- K0750 DATA Word 31213, Word 31254, Word 31295, Word 31338, Word 31379, Word 31420, Word 31461, Word 31504, Word 31545, Word 31585
- K0760 DATA Word 31628, Word 31669, Word 31710, Word 31751, Word 31792, Word 31833, Word 31876, Word 31917, Word 31957, Word 32000
- K0770 DATA Word 32040, Word 32082, Word 32124, Word 32164, Word 32206, Word 32246, Word 32289, Word 32329, Word 32371, Word 32411
- K0780 DATA Word 32453, Word 32495, Word 32536, Word 32577, Word 32618, Word 32659, Word 32700, Word 32742, Word 32783, Word 32824
- K0790 DATA Word 32865, Word 32905, Word 32947, Word 32987, Word 33029, Word 33070, Word 33110, Word 33152, Word 33192, Word 33234
- K0800 DATA Word 33274, Word 33316, Word 33356, Word 33398, Word 33439, Word 33479, Word 33521, Word 33561, Word 33603, Word 33643
- K0810 DATA Word 33685, Word 33725, Word 33767, Word 33807, Word 33847, Word 33889, Word 33929, Word 33970, Word 34012, Word 34052
- K0820 DATA Word 34093, Word 34134, Word 34174, Word 34216, Word 34256, Word 34296, Word 34338, Word 34378, Word 34420, Word 34460
- K0830 DATA Word 34500, Word 34542, Word 34582, Word 34622, Word 34664, Word 34704, Word 34744, Word 34786, Word 34826, Word 34866
- K0840 DATA Word 34908, Word 34948, Word 34999, Word 35029, Word 35070,

- Word 35109, Word 35151, Word 35192, Word 35231, Word 35273
- K0850 DATA Word 35313, Word 35353, Word 35393, Word 35435, Word 35475,  
Word 35515, Word 35555, Word 35595, Word 35637, Word 35676
- K0860 DATA Word 35718, Word 35758, Word 35798, Word 35839, Word 35879,  
Word 35920, Word 35960, Word 36000, Word 36041, Word 36081
- K0870 DATA Word 36121, Word 36162, Word 36202, Word 36242, Word 36282,  
Word 36323, Word 36363, Word 36403, Word 36443, Word 36484
- K0880 DATA Word 36524, Word 36564, Word 36603, Word 36643, Word 36685,  
Word 36725, Word 36765, Word 36804, Word 36844, Word 36886
- K0890 DATA Word 36924, Word 36965, Word 37006, Word 37045, Word 37085,  
Word 37125, Word 37165, Word 37206, Word 37246, Word 37286
- K0900 DATA Word 37326, Word 37366, Word 37406, Word 37446, Word 37486,  
Word 37526, Word 37566, Word 37606, Word 37646, Word 37686
- K0910 DATA Word 37725, Word 37765, Word 37805, Word 37845, Word 37885,  
Word 37925, Word 37965, Word 38005, Word 38044, Word 38084
- K0920 DATA Word 38124, Word 38164, Word 38204, Word 38243, Word 38283,  
Word 38323, Word 38363, Word 38402, Word 38442, Word 38482
- K0930 DATA Word 38521, Word 38561, Word 38600, Word 38640, Word 38679,  
Word 38719, Word 38759, Word 38798, Word 38838, Word 38878
- K0940 DATA Word 38917, Word 38957, Word 38996, Word 39036, Word 39076,  
Word 39115, Word 39164, Word 39195, Word 39234, Word 39274
- K0950 DATA Word 39314, Word 39353, Word 39393, Word 39432, Word 39470,  
Word 39511, Word 39549, Word 39590, Word 39628, Word 39668
- K0960 DATA Word 39707, Word 39746, Word 39786, Word 39826, Word 39865,  
Word 39905, Word 39944, Word 39984, Word 40023, Word 40061
- K0970 DATA Word 40100, Word 40140, Word 40179, Word 40219, Word 40259,  
Word 40298, Word 40337, Word 40375, Word 40414, Word 40454
- K0980 DATA Word 40493, Word 40533, Word 40572, Word 40610, Word 40651,  
Word 40689, Word 40728, Word 40765, Word 40807, Word 40846
- K0990 DATA Word 40885, Word 40924, Word 40963, Word 41002, Word 41042,  
Word 41081, Word 41119, Word 41158, Word 41198, Word 41237
- K1000 DATA Word 41276, Word 41315, Word 41354, Word 41393, Word 41431,  
Word 41470, Word 41509, Word 41548, Word 41587, Word 41626
- K1010 DATA Word 41665, Word 41704, Word 41743, Word 41781, Word 41820,  
Word 41859, Word 41898, Word 41937, Word 41976, Word 42014
- K1020 DATA Word 42053, Word 42092, Word 42131, Word 42169

```

' =====
'
' File..... volmeter.BPE
' Purpose... caption voltage
' Author.... RUDI RADRIGAN.
' E-mail.... rradriga@udec.cl
' Started...
' Updated... 10 dec 2006
'
' {$STAMP BS2px}
' {$PBASIC 2.5}
' =====

ADres VAR Byte                                ' A-to-D result: one byte.
fi VAR Bit
'CS CON 5                                     ' Chip select is pin 0.
'AData CON 6                                 ' ADC data output is pin 1.
'CLK CON 7                                  ' Clock is pin 2.
fi=0
again:
HIGH 5                                       ' Deselect ADC to start.
LOW 5                                       ' Activate the ADC0831.
LOW 7                                       ' Pin out
PULSOUT 7, 210
SHIFTIN 6,7,MSBPOST,[ADres\8]              ' Shift in the data.
IF fi =1 THEN as
fi=fi+1
GOTO again                                  ' Do it again.
as:
DEBUG ? ADres
PAUSE 500

```

## 2. Algunos componentes de la aplicación desarrollados en Visual Basic 6.0®

La aplicación busca y verifica en que puerto esta conectado al interior del host y le asigna una dirección a la cual se conecta para almacenar los datos desde la Ram a un puerto virtual que genera el host, el código en detalle de esta y otras funciones está explícito a continuación:

```
Private Sub Command1_Click()
```

```
Text9.Text = ""
```

```
Command7.Enabled = True
```

```
Command9.Enabled = True
```

```
Dim MEN As String
```

```
MEN = MsgBox("Presione el botón de Enlace del Sistema de Control",  
vbOKOnly + vbInformation, "Confirmación")
```

```
Dim comen
```

```
If MEN = vbOK Then
```

```
comen = Time
```

```
End If
```

```
Command1.Visible = False
```

```
frmdiagnostico.Label4.Visible = True
```

```
frmdiagnostico.Label5.Visible = False
```

```
Dim aa, amp, i, pp, vol As Integer
```

```
Dim Mensaje1, Zz, Dw As String
```

```
Dim db As Database
```

```
Dim rs As Recordset
```

```
Dim A() As Variant
```

```
pp = Val(Label5)
```



```

Hi = Val(frmFormulario.cbohumedadinicial)
aa = 0
FU = 0
vol = 0
TiempoTotal = 0
amp = 0
Patron = 0
Do
10
    TiempoPausa = 1
    Inicio = Timer
    cbohorainicio = comen
    Do While Timer < Inicio + TiempoPausa
    DoEvents
    i = i + 1
        MSComm1.CommPort = pp
        MSComm1.Settings = "19200,n,8,1"
        MSComm1.InputLen = 0
        MSComm1.PortOpen = True
        MSComm1.Output = "atencion" + Chr$(13)
    Do
    DoEvents
    Loop Until MSComm1.InBufferCount >= 1
        aa = aa + 1
    If aa = 1 Then
        vol = MSComm1.Input
        Text8.Text = vol
    End If
    If Fuu = 3 Then
        Mensaje1 = MsgBox("Presione Enlace", vbExclamation +
vbDefaultButton1, "Ingreso de información")
    End If

```

```
Fuu = 0
If vol < 1 Then
    MSComm1.Output = Chr$(52)
    frmdiagnostico.Label8.Visible = True
    frmdiagnostico.Label9.Visible = False
Else
    MSComm1.Output = Chr$(49)
    frmdiagnostico.Label8.Visible = False
    frmdiagnostico.Label9.Visible = True
End If
```

```
If aa = 2 Then
    amp = MSComm1.Input
    Text5.Text = amp
    If amp < -1 Then
        frmdiagnostico.Label7.Visible = False
        frmdiagnostico.Label6.Visible = True
        MSComm1.Output = Chr$(50)
    End If
```

```
    If amp > -1 Then
        frmdiagnostico.Label6.Visible = False
        frmdiagnostico.Label7.Visible = True
        MSComm1.Output = Chr$(51)
    End If
End If
```

```
If aa = 3 Then
    temp = MSComm1.Input
    temp = Format((temp / 20.40184802), "#0.00") ' convierte vol a
temperatura del tipo T
    Text7.Text = temp
    MSComm1.Output = Chr$(49)
```

```

End If
94
MSComm1.PortOpen = False
'imprime en la base de datos los valores obtenidos
If i > 4 Then
    Zz = frmFormulario.archivo.Text
    Set db = OpenDatabase("c:\Archivos de programa\ conductividad\
Historial\" + Zz + ".mdb")
    Set rs = db.OpenRecordset("Informe")
    rs.AddNew
        rs.Fields(0) = TiempoTotal
        Text3.Text = TiempoTotal
        rs.Fields(2) = Text8.Text
        rs.Fields(3) = Text5.Text
        rs.Fields(4) = Text7.Text
    rs.Update
End If
Label24.Caption = tt
Label38.Caption = Label24
Label25.Caption = tt + 1
Label37.Caption = Label25
Label26.Caption = tt + 2
Label36.Caption = Label26
Label27.Caption = tt + 4
Label35.Caption = Label27
Label33.Caption = tt + 3
Label34.Caption = Label33
If j > 1 Then
    pantalla.ForeColor = QBColor(1)
    pantalla.ScaleMode = 4
    pantalla.ScaleHeight = 700
    pantalla.ScaleWidth = 2145

```

```

    Dat = 710 - temp * 6.5
    Dat1 = 710 - Oto * 6.5
    Dat2 = 693 - HRcor * 6.5
    Dat3 = 693 - Fac1 * 6.5
    pantalla.Line (j, Dat)-(j - 1), Dat1), &HFFFFFFF
    pantalla.Line (j, Dat2)-(j - 1), Dat3), &HFFFF&
    pantalla1.ForeColor = QBColor(1)
    pantalla1.ScaleMode = 4
    pantalla1.ScaleHeight = 2200
    pantalla1.ScaleWidth = 2145
    pantalla1.Line (j, ((2200 - bb) * 1.222))-(j - 1), ((2200 - fac3) * 1.222)),
&HFFFF80
End If
    fac3 = temp
    Fac1 = vol
    Oto = temp
If j > 2145 Then
    j = 0
    pantalla.Cls
    pantalla1.Cls
    tt = tt + 5
End If
If Text9.Text = "detener" Then
    GoTo 321
End If

Text7 = frmFormulario.cbotemperaturadeprocso
Loop
    Final = Time
    TiempoTotal = TiempoTotal + 1
    Text3.Text = TiempoTotal
If Minim > bb Then

```

```

        Zz = frmFormulario.archivo.Text
        Set db = OpenDatabase("c:\Archivos de
programa\Conductividad\Historial\" + Zz + ".mdb")
        Set rs = db.OpenRecordset("Informe")
        rs.AddNew
        rs.Fields(0) = TiempoTotal
        rs.Update

        If timepototal = 60 Then
            Mensaje1 = MsgBox("Tiempo Concluido, ¡¡APAGUE EL EQUIPO!!",
vbOKOnly + vbInformation, "El Proceso a finalizado")
            INSITU.menucalculos.Enabled = True
            Command6.Enabled = False
            Command7.Enabled = False
            Command8.Enabled = False
            Command9.Enabled = False
            GoTo 800
        End If
    End If
Loop
321
800
End Sub

Private Sub Command6_Click()
    Dim MEN
    MEN = MsgBox("Esta seguro volver al Formulario", vbInformation +
vbYesNo, "Confirmación")
    If MEN = vbYes Then
        frmFormulario.Show
    End If
End Sub

Private Sub Command7_Click()

```

```

Dim MEN As String
MEN = MsgBox("Esta seguro de salir del Programa", vbCritical +
vbYesNo, "Confirmación")
If MEN = vbYes Then
    MEN = MsgBox("¡¡APAGUE EL EQUIPO!!", vbOKOnly +
vbInformation, "Confirmación")
    MSComm1.PortOpen = False
End
End If
End Sub
Private Sub Command8_Click()
    frmdiagnostico.Show
End Sub
Private Sub Command9_Click()
    Text9.Text = "detener"
    Dim MEN As String
    MEN = MsgBox("¡¡APAGUE EL EQUIPO!!", vbOKOnly +
vbInformation, "Confirmación")
    INSITU.menucalculos.Enabled = True
    Command8.Enabled = False
    Command7.Enabled = False
    Command6.Enabled = False
    Command9.Enabled = False
End Sub
Private Sub Form_Load()
    Command7.Enabled = False
    Command9.Enabled = False
    Command1.Visible = True
    Data1.Visible = False
    INSITU.menucalculos.Enabled = False
    INSITU.cbooperador2 = frmFormulario.cbooperador1
    INSITU.cboproducto2 = frmFormulario.cboproducto1

```

```

Shape1.Visible = False
' verifica el dispositivo conectado y el numero del puerto activo
Dim p As Integer
    On Error GoTo 0 ' Desactiva la detección de errores.
    On Error Resume Next ' Retarda detección de errores.
    p = 1
5
    If p < 10 Then
        MSComm1.CommPort = p
        MSComm1.PortOpen = True
        If MSComm1.PortOpen = True Then
            If MSComm1.InBufferCount = 1 Then ' pregunta por dispositivo
                MSComm1.PortOpen = False
            Else
                GoTo 15
            End If
        Else
            GoTo 10
        End If
    10
        If Err.Number = 8002 Then
            ' Indica al usuario lo que ha ocurrido. Luego borra el objeto Err.
15    MSComm1.PortOpen = False
            p = p + 1
            Err.Clear ' Borra campos del objeto Err
            GoTo 5
        End If
        Label5.Caption = p
        Shape2.Visible = False
        Shape1.Visible = True
        GoTo 25
    End If

```

```

        Shape2.Visible = True
        Dim MEN As String
        MEN = MsgBox("No existe conexión con el equipo", vbCritical +
vbYesNo, "Confirmación")
        25
    End Sub

    Private Sub menu_salir_Click()
        Dim MEN
        MEN = MsgBox("Esta seguro de salir del Programa", vbCritical +
vbYesNo, "Confirmación")
        If MEN = vbYes Then
            End
        End If
    End Sub

    Private Sub menucalculos_Click()
        Frmcalculos.Show
        Unload Me

    End Sub

```

El modulo de análisis de datos llama a la base, comprueba que esta sea compatible con el programa y despliega los datos en un informe, calcula la regresion por minimos cuadrados, este procedimiento es llamado como modulo público matematico, los codigos se muestran a continuación:

```

Private Sub cmdOpen_Click()

'-----

    Dim db As Database

    Dim rs As Recordset

```



```

'-----

'CODIGO PARA ABRIR EL ARCHIVO

On Error GoTo ControlError

Frame1.Visible = True


Command4.Enabled = False

Command1.Enabled = True

Command1.Visible = False

grdDataGrid.Visible = False

CommonDialog1.Filter = "Bases de Datos (*.mdb)|*.mdb|"

CommonDialog1.FilterIndex = 1

CommonDialog1.InitDir      =      "c:\Archivos      de
programa\conductividad\Historial\"

CommonDialog1.ShowOpen

archivo = CommonDialog1.FileName

'-----

Data1.DatabaseName = archivo

'-----

Label1.Caption = archivo ' ARCHIVO SE DEFINIO COMO UNA
VARIABLE PUBLICA

'-----

'ESTE REFRESCA EL DATA, PARA MOSTRARLO EN EL DATA
GRID

Data1.Refresh

'-----

If archivo = "" Then

```

```

Frame1.Visible = False

Command4.Enabled = False

Command1.Enabled = False

grdDataGrid.Visible = False

imgcilindro.Visible = False

MsgBox "Seleccione una Base de Datos!", vbExclamation +
vbApplicationModal + vbOKOnly

GoTo 20

End If

Dim A() As Variant

'toma los valores de la basa

Set db = OpenDatabase(archivo)

Set rs1 = db.OpenRecordset("Datos")

rs1.MoveFirst

Text9.Text = (rs1.Fields(4))

Text8.Text = (rs1.Fields(5))

Close

'GENERADOR DE ERROR

'POSIBLE ERROR

Set db = OpenDatabase(archivo)

Set rs = db.OpenRecordset("Informe")

GoTo 50

Exit Sub

```

ControlError:

Dim strError As String

Dim errBucle As Error

For Each errBucle In Errors

With errBucle

strError = \_

"Número de Error " & 3078 & vbCr

End With

'AQUI SE INAVILITA TODO

Frame1.Visible = False

Command4.Enabled = False

Command1.Enabled = False

grdDataGrid.Visible = False

imgcilindro.Visible = False

Picture1.Visible = False

MsgBox " La Base de Datos no es Compatible con TC 1.0!",  
vbExclamation + vbApplicationModal + vbOKOnly

Label1.Caption = ""

Next

Resume 25

50

'CALCULO DE FRACCION DE HUMEDAD

```

rs.MoveFirst

'(COLUMNAS,FILAS)

Dim v, vol, Am, amp As Double

ReDim A(0 To 9, 1 To rs.RecordCount())

v = 0

vol = 0

Am = 0

amp = 0

For i = 1 To rs.RecordCount()

    A(0, i) = rs.Fields(0) 'tiempo

    A(1, i) = rs.Fields(2) 'vol

    A(2, i) = rs.Fields(3) 'amp

'reescribir en campos nuevos

rs.Edit

rs.Fields(1) = (Log(A(0, i)))

'asigna valor a campo nuevo

v = A(1, i)

vol = vol + v

Am = A(2, i)

amp = amp + Am
    
```

rs.Update

rs.MoveNext

Next i

20

ESTE CAPTURA EL TIEMPO TOTAL DEL PROCESO Y  
PROMEDIOS DE VOL Y AMPER

Dim qq As Double

qq = A(0, i - 1)

vol = Format((vol / (i - 1)), "#0.0000")

amp = Format((amp / (i - 1)), "#0.0000")

Text1.Text = qq

Text2.Text = vol

Text6.Text = amp

Label12.Caption = vol

Label13.Caption = amp

Command1.Enabled = True

Command4.Enabled = True

Command1.Visible = True

Command4.Visible = True

25

End Sub

ESTA PARTE DEL DATA GRID

Private Sub Command4\_Click()

```

grdDataGrid.Refresh

grdDataGrid.Visible = True

Command4.Visible = False

Command7.Visible = True

End Sub

Private Sub Command5_Click()

Unload Me

End Sub


Private Sub Command7_Click()

Command4.Visible = True

Command7.Visible = False

grdDataGrid.Visible = False

End Sub


Private Sub Form_Load()

'INAVILITA TODO LOS BOTONES DEL FORMULARIO

Command7.Visible = False

Frame1.Visible = False

Command4.Enabled = False

Command1.Enabled = False

imgcilindro.Visible = False

grdDataGrid.Visible = False

Picture1.Visible = False

```

Command1.Visible = False

'-----

CommonDialog1.FilterIndex = 1

archivo = CommonDialog1.FileName

Data1.DatabaseName = archivo

Label1.Caption = archivo

Data1.Refresh

'-----

End Sub

'DETERMINACION DEL PROCESO

Private Sub Command1\_Click()

Dim db As Database

Dim rs As Recordset

Dim A() As Double

Dim coef() As Double

Dim datos() As Double

Command4.Enabled = True

Picture1.Visible = True

imgcilindro.Visible = True

grdDataGrid.Visible = False

archivo = [Label1]

Set db = OpenDatabase(archivo)

Set rs = db.OpenRecordset("Informe")

```
rs.MoveFirst
```

```
'(COLUMNAS,FILAS)
```

```
ReDim A(1 To 10, 1 To rs.RecordCount())
```

```
ReDim datos(1 To 10, 1 To rs.RecordCount())
```

```
For i = 1 To rs.RecordCount()
```

```
    A(1, i) = rs.Fields(1)
```

```
    A(2, i) = rs.Fields(4)
```

```
    datos(1, i) = A(1, i)
```

```
    datos(2, i) = A(2, i)
```

```
    ' Se mueve al siguiente registro...
```

```
    rs.MoveNext
```

```
Next i
```

```
coef = regresionPolinomial(datos, 1)
```

```
Text7.Text = Format((coef(1, 2)), "#, #0.0000")
```

```
Text10.Text = Format(((Val(Label13) ^ 2) * ((Val(Label12)) /  
(Val(Label13)))) / (4 * 3.1416 * 0.9259 * Text7.Text), "#0.00000")
```

```
End Sub
```

```
Private Sub text7_KeyPress(KeyAscii As Integer)
```

```
    If KeyAscii > Asc("9") Then
```

```
        If Not IsNumeric("0" & Text7.Text & Chr(KeyAscii)) Then
```



```

        '... se desecha esa tecla y se avisa de que no es correcta

        Beep

        KeyAscii = 0

    End If

End If

End Sub

```

```

Private Sub Text8_KeyPress(KeyAscii As Integer)

    If KeyAscii > Asc("9") Then

        If Not IsNumeric("0" & Text8.Text & Chr(KeyAscii)) Then

            '... se desecha esa tecla y se avisa de que no es correcta

            Beep

            KeyAscii = 0

        End If

    End If

End Sub

```

```

'*

'* Módulo    : math.bas

'* Descripción : Módulo estandar que implementa funciones
matemáticas.

'* Fecha      : 28 de Agosto del 2005

'*

```

'\*

'\* Implementación de funciones

'\*

\*\*\*\*\*

'\* Función que implementa la sumatoria de x a la k en

'\* un conjunto de datos...

Public Function sumXK(ByRef datos() As Double, k As Integer) As Double

'\* Variables

Dim i As Integer

Dim suma As Double

'\* Implementación de la suma

suma = 0

For i = 1 To UBound(datos, 2)

    suma = suma + datos(1, i) ^ k

Next i

'\* Se retorna el valor ...

sumXK = suma

End Function

\*\*\*\*\*

'\* Función que implementa la suma de x a la k por y

'\* en un conjunto de datos...

Public Function sumXKY(ByRef datos() As Double, k As Integer) As Double

'\* Variables

Dim i As Integer

Dim sum As Double

'\* Implementación de la suma

suma = 0

For i = 1 To UBound(datos, 2)

    suma = suma + datos(1, i) ^ k \* datos(2, i)

Next i

'\* Se retorna el valor...

sumXKY = suma

End Function

\*\*\*\*\*

'\* Función que genera la suma de todos los valores de y

'\* en un set de datos...

Public Function sumY(ByRef datos() As Double) As Double

'\* Variables

Dim i As Integer

Dim suma As Double

'\* Implementación

suma = 0

For i = 1 To UBound(datos, 2)

    suma = suma + datos(2, i)

Next i

'\* Se retorna el valor de la suma...

sumY = suma

End Function

\*\*\*\*\*

'\* Función que genera los coeficientes de para la

'\* regresión polinomial

Public Function regresionPolinomial(datos() As Double, grado As Integer) As Double()

'\* Variables

ReDim coef(1 To 1, 1 To grado + 1) As Double

ReDim inv(1 To 1, 1 To grado + 1, 1 To grado + 1) As Double

ReDim vector(1 To 1, 1 To grado + 1) As Double ' Vector que almacena la otra parte de la regresión

ReDim matriz(1 To grado + 1, 1 To grado + 1) As Double ' Matriz necesaria para generar la regresión

Dim i As Integer, j As Integer, k As Integer, p As Integer

'\* Implementación

'\* Se genera el vector...

p = 0

For i = 1 To (grado + 1)

vector(1, i) = sumXKY(datos, p)

p = p + 1

Next i

'\* Se genera la matriz...

matriz(1, 1) = UBound(datos, 2) ' Número de datos en el set...

'\* Se termina de llenar la primera fila...

p = 1

For i = 2 To (grado + 1)

matriz(1, i) = sumXK(datos, p)

p = p + 1

Next i

'\* Se completa el resto de la matriz

k = 1

For i = 2 To (grado + 1)

p = k

For j = 1 To grado

k = k + 1

matriz(i, j) = matriz(i - 1, j + 1)

Next j

matriz(i, grado + 1) = sumXK(datos, k)

p = p + 1

k = p

Next i

'\* Se multiplica el vector por la inversa de la matriz

'\* y se generan los coeficientes de la regresión...

inv = inversa(matriz, 0.01)

coef = producto(vector, inv)

regresionPolinomial = coef

End Function

\*\*\*\*\*

'\* Función que genera la inversa de una matriz

```
Public Function inversa(ByRef matriz() As Double, tolerancia As
Double) As Double()
```

```
    '**
```

```
    '* Precondición : - La matriz es cuadrada.
```

```
    '**
```

```
    '* Variables
```

```
    Dim i, j, k As Integer
```

```
    Dim valor As Double
```

```
    ReDim identidad(1 To UBound(matriz, 1), 1 To UBound(matriz, 2))
As Double
```

```
    '* Implementación
```

```
    '* Se genera la matriz identidad...
```

```
    For i = 1 To UBound(matriz, 1)
```

```
        For j = 1 To UBound(matriz, 2)
```

```
            identidad(i, j) = 0
```

```
        Next j
```

```
        identidad(i, i) = 1
```

```
    Next i
```

```
    '* Se genera la matriz identidad pero en la matriz pasada
```

```
    '* por parámetro, y en la matriz identidad recién creada se
```

```
    '* realizan las mismas operaciones que para la matriz parámetro...
```

```
    For i = 1 To UBound(identidad, 1) - 1
```

```
        For j = i + 1 To UBound(identidad, 1)
```

```
            valor = (-1) * matriz(j, i) / matriz(i, i)
```

```
            For k = 1 To UBound(identidad, 2)
```

```

        matriz(j, k) = matriz(j, k) + valor * matriz(i, k)

        identidad(j, k) = identidad(j, k) + valor * identidad(i, k)

    Next k

Next j

Next i

For i = UBound(identidad, 1) To 2 Step -1

    For j = i - 1 To 1 Step -1

        valor = (-1) * matriz(j, i) / matriz(i, i)

        For k = UBound(identidad, 1) To 1 Step -1

            matriz(j, k) = matriz(j, k) + valor * matriz(i, k)

            identidad(j, k) = identidad(j, k) + valor * identidad(i, k)

        Next k

    Next j

Next i

'* Se normalizan los valores para la matriz identidad

For i = 1 To UBound(matriz, 1)

    For j = 1 To UBound(matriz, 2)

        identidad(i, j) = identidad(i, j) / matriz(i, i)

    Next j

    matriz(i, i) = 1

Next i

'* Se retorna la matriz inversa

inversa = identidad

End Function

```

```
*****
```

```
'* Función que retorna la matriz con el producto entre 2 matrices...
```

```
Public Function producto(matriz1() As Double, matriz2() As Double) As  
Double()
```

```
'* Variables
```

```
Dim i, j, k As Integer
```

```
Dim suma As Double
```

```
ReDim prod(1 To UBound(matriz1, 1), 1 To UBound(matriz2, 2)) As  
Double
```

```
'* Implementación
```

```
For i = 1 To UBound(matriz1, 1)
```

```
    For j = 1 To UBound(matriz2, 1)
```

```
        For k = 1 To UBound(matriz1, 2)
```

```
            suma = suma + matriz1(i, k) * matriz2(k, j)
```

```
        Next k
```

```
        prod(i, j) = suma
```

```
        suma = 0
```

```
    Next j
```

```
Next i
```

```
'* Se retorna la matriz con el producto
```

```
producto = prod
```

```
End Function
```