UNIVERSIDAD de VALLADOLID

ESCUELA de INGENIERÍAS INDUSTRIALES

INGENIERO TÉCNICO INDUSTRIAL, ESPECIALIDAD EN ELECTRÓNICA

PROYECTO FIN DE CARRERA

ESTUDIO DEL MICROCONTROLADOR AVANZADO DSPIC

Autores:

Rodríguez Vega, Juan José

Del Campo García, Mariano

Tutor:

Plaza Pérez, Francisco Tecnología Electrónica

JUNIO – 2013

ÍNDICE GENERAL:

1. **MEMORIA:** *82 PÁGINAS.*

2. **PLANOS:** *3 PÁGINAS.*

3. **PLIEGO DE CONDICIONES:** *34 PÁGINAS.*

4. **ANEXOS:** *5 PÁGINAS.*

MEMORIA

ÍNDICE:

1. ENUNCIADO Y JUSTIFICACIÓN DEL PROYECTO 05
1.1. ENUNCIADO 05
1.2. OBJETIVO 05
1.3. JUSTIFICACIÓN 05
2. CREACION DE LA PAGINA WEB 06
2.1. INTRODUCCION 06
2.2. MAPA WEB 07
2.3. ASPECTOS GENERALES 09
2.4. TRATAMIENTO DE IMÁGENES 10
2.5. ESTILOS CSS 11
2.6. SERVIDORES DE PRUEBA: LOCAL Y REMOTO 11
2.7. COLORES BÁSICOS DE LA WEB 14
2.8. EVALUACIÓN Y PRUEBAS 14
2.8.1. Validación CSS 15
2.8.2. Verificación de enlaces 15
2.8.3. Análisis de resoluciones de pantalla 16
2.8.4. Estudio de navegadores web 17
2.9. PROBLEMAS AL REALIZAR LA WEB 17
2.10. PROGRAMAS UTILIZADOS 18
2.10.1 Dreamweaver 18
2.10.2 Adobe Photoshop 20
2.10.3 FileZilla 21
2.10.4 XAMPP Control Panel 21
2.11. TECNOLOGIAS UTILIZADAS 22
2.11.1 HTML 22
2.11.2 CSS 23
2.11.3 JavaScript 24
3. ESTUDIO DEL MICROCONTROLADOR AVANZADO dsPIC 24
3.1. ESTUDIO DEL dsPIC30F6010 24
3.1.1. Memoria de Datos 25
3.1.2. Memoria de Programa 25
3.1.3. Camino de Datos 26
3.1.4. Puertas de E/S Multifunción 26
3.1.5. Periféricos diversos 27
3.1.6. Gestión del sistema y la energía 28
3.1.7. Otros recursos a destacar 28

4. CUESTIONARIO **29**

4.1. PROGRAMAS UTILIZADOS 30

4.1.1. Hotpotatoes 30

5. APLICACIONES **30**

5.1. ENTRADAS Y SALIDAS MULTIFUNCION 30

5.1.1. Descripción 30

5.1.2. Desarrollo 31

5.2. TEMPORIZADORES. TIMER1 DE 16 BITS 33

5.2.1. Descripción 33

5.2.2. Desarrollo 33

5.3. TEMPORIZADORES. TIMER2/3 DE 32 BITS 36

5.3.1. Descripción 36

5.3.2. Desarrollo 36

5.4. CONVERSOR A/D DE 10 BITS 38

5.4.1. Descripción 39

5.4.2. Desarrollo 40

5.5. MÓDULO DE CAPTURA DE ENTRADA 43

5.5.1. Descripción 43

5.5.2. Desarrollo 44

5.6. MÓDULO DE COMPARACION DE SALIDA 47

5.6.1. Descripción 47

5.6.2. Desarrollo 48

5.7. MÓDULO PWM. CONTROL DE MOTORES 51

5.7.1. Descripción 52

5.7.2. Desarrollo 52

5.8. MÓDULO DE COMUNICACIÓN. UART 57

5.8.1. Descripción 57

5.8.2. Desarrollo 58

5.9. MÓDULO DE COMUNICACIÓN. SPI 61

5.9.1. Descripción 61

5.9.2. Desarrollo 62

5.10. MÓDULO DE COMUNICACIÓN. I^2C 66

5.10.1. Descripción 66

5.10.2. Desarrollo 67

5.11. INTERRUPCIONES Y EXCEPCIONES 75

5.11.1. Descripción 75

5.11.2. Desarrollo 75

5.12. PROGRAMAS UTILIZADOS 76

5.12.1. Proteus y entorno VSM 77

6. MANUAL DE USUARIO 78

6.1. REQUERIMIENTOS 78

6.2. PÁGINA PRINCIPAL 78

6.3. ACCESO 81

6.4. MENU DE NAVEGACIÓN 81

1. ENUNCIADO Y JUSTIFICACIÓN DEL PROYECTO.

1.1. ENUNCIADO.

Se redacta el siguiente proyecto de "ESTUDIO DEL MICROCONTROLADOR AVANZADO DSPIC", atendiendo a la solicitud formulada por D. Francisco Plaza, por los Ingenieros Técnicos Industriales D. Mariano Del Campo García y D. Juan José Rodríguez Vega.

El proyecto consta de varias partes:

- Realización de una página web en la que ira el estudio del dsPIC.
- Estudio teórico del microcontrolador avanzado dsPIC.
- Cuestionario, que va incluido en la web, con auto corrector y selección de temas para la evaluación de los conocimientos teóricos del estudio.
- Ejemplos prácticos sencillos de simulación con Proteus de la familia dspic33f, para la comprensión y estudio de los conocimientos teóricos adquiridos en el proyecto.

1.2. OBJETIVO.

La finalidad de este proyecto, es que esta página web con su estudio del microcontrolador avanzado dsPIC, sirva como módulo formativo para estudiantes de ingeniería, y en especial a los de la rama de electrónica industrial; estudiantes que quieran introducirse en el mundo de los controladores digitales de señal (DSC), como apoyo para impartir una asignatura libre/optativa o para algún curso especializado; en la que el estudiante, con esta web pueda autónomamente aprender a programar y conocer más o menos en profundidad estos microcontroladores especiales.

Este proyecto, que se ve materializado en una web, es el conjunto de muchos manuales, tutoriales, libros, páginas web, trabajo y dedicación de dos estudiantes de Ingeniería Técnica con especialidad en electrónica industrial.

1.3. JUSTIFICACIÓN DEL ESTUDIO.

El ¿Por qué? de este estudio: la electrónica avanza según avanza la demanda y la tecnología. Hace no mucho tiempo, no se requería tanta precisión en cálculos, ni de tanta calidad de imagen en las televisiones, por poner un ejemplo; antes con la calidad y almacenaje que nos proporcionaba el DVD, no necesitábamos de una TV de alta de definición, después evoluciona y aparece el blue-ray con mucha más capacidad y calidad, por lo que asociado a esto, necesitamos de una TV más potente y que nos rinda esta mejora,… ¿Qué quiero decir con esto?, pues quiero decir que el dsPIC nace como una necesidad de evolución de los microcontroladores anteriores ante el próspero avance tecnológico que estamos viviendo.

El dsPIC aparte de mejorar en tamaño, robustez, capacidad de periféricos, rapidez…etc, lo que hace es procesar las señales digitalmente, y esto es muy importante cuando el volumen de datos a manejar (cada vez mayor por el avance tecnológico) es grande, y es lo que le diferencia sobre todo de los antecesores.

Para no estancarnos en el pasado, en el básico aunque necesario estudio del PIC, hacemos un salto y nos ponemos a estudiar este microcontrolador avanzado.

2. CREACIÓN DE LA PAGINA WEB.

2.1. INTRODUCCION.

Para la creación de la página web hemos seguido la pauta del Tutor y la lógica, al ser una web formativa, se trata más de enseñar la información en ella contenida que el continente. No hemos hecho un diseño espectacular ni complicado, que lo que hace a nuestro modo de verlo es ocupar más memoria y ser más complicada la obtención de la información. Se trataba de confeccionar algo sencillo fácilmente programable y editable, para darle más versatilidad al proyecto y que pueda servir de plantilla a otros estudios futuros.

Hemos intentado que desde la primera página se acceda rápidamente y a simple vista a todos los contenidos fundamentales del proyecto; una introducción al mundo de los dsPIC, la descripción de la arquitectura interna del dsPIC30f (el eje de nuestro estudio) realizando un estudio exhaustivo sobre las memorias, el camino de datos, las puertas de E/S, los periféricos integrados, la gestión del sistema y la energía y por último un link de descargas que contiene las aplicaciones incluyendo los manuales y el software en versión demo para poder modificarlas y ejecutarlas. Al pasar el ratón por encima de los botones del menú, se desplegaran otros submenús importantes, para poder acceder a toda la información incorporada en la WEB consiguiendo que con una simple pasada sepamos donde está todo y se pueda acceder rápida y cómodamente a cualquier tema o bloque, y no como en otras páginas WEB más complicadas que basan su funcionamiento en ir abriendo nuevas ventanas Ver Img.1.

Img.1. Captura del diseño de la Pagina Web.

2.2. MAPA WEB.

Lo primero que vamos a visionar es el mapa del sitio de la página Web, para ver la raíz de ésta.

- Index
  - Mundo dsPIC
    - Introducción dsPIC (Index)
    - Familia dsPIC30F
    - Familia dsPIC33F/E
  - Arquitectura interna
  - Memorias
    - Memoria de Datos
      - Direccionamiento modular
      - Inversión de acarreo
    - Memoria de Programa

- Grabación de FLASH y EEPROM

- Camino de Datos
  - Banco de Registros "W"
  - ALU de 16 bits
  - Motor DSP
    - Multiplicador de 17x17 bits
    - Registro de desplazamiento de 40 bits
    - Sumador/Restador de 40 bits
    - Acumuladores de 40 bits (A y B)
    - Lógica de redondeo
    - Lógica de saturación
  - Unidad de división
- Puertas E/S
- Periféricos integrados
  - Temporizadores
    - Módulo Timer 1
    - Módulo Timer 2/3
    - Módulo Timer 4/5
  - Conversor A/D de 10 bits
  - Módulo de Captura
  - Módulo de Comparación/PWM
  - Módulo PWM Control de Motores
  - Módulos de Comunicación
    - Módulo UART

- Módulo SPI
- Módulo I²C
- Módulo CAN
- Módulo DCI

- Módulo QEI

- Gestión del sistema y la energía
  - El sistema Oscilador
  - El sistema de Reset
  - Temporizador WDT y Módulo LVD.
  - Modos de bajo consumo de energía
  - Interrupciones y excepciones
- Descargas

2.3. ASPECTOS GENERALES.

Nuestra página web ha sido realizada a través del programa Dreamweaver CS5.5. Dicha página está basada en una plantilla (diseño.dwt) que nosotros hemos creado con las siguientes características:

- Una cabecera (header) en la que incluimos el título “Estudio del dsPIC” y una imagen que describe gráficamente que el dsPIC es un dispositivo híbrido formado por un MCU de 16 bits y un DSP de gama media/baja.

- Después de la cabecera, dentro del cuerpo de la página (body) incluimos un menú desplegable de SPRY (MenuBar1) gobernado por funciones JavaScript prediseñadas por el propio programa, en el que incluimos la distribución del “Mapa WEB” anteriormente descrito, mediante vínculos a los correspondientes HTML.

- Posteriormente creamos una región editable (también dentro del body), que será la parte de la plantilla que podrá ser editada cuando creemos las correspondientes páginas (HTML) de la WEB basándonos en la plantilla, y además, también creamos unas regiones opcionales en las que incluimos los botones de:
  - Descarga (de aplicaciones).

- Ir Arriba.
- Cuestionario.

Los botones, dotados de imágenes decorativas, serán incluidos en las páginas en las que sea necesario.

- Por último, y no menos importante, hemos creado un pie de página (footer) en el que incluimos las imágenes corporativas de la Universidad de Valladolid (UVA) y de la Escuela de ingenierías Industriales (EII), en las cuales insertamos un "Link" para acceder de manera rápida a las paginas oficiales de ambos estamentos. También incluimos los nombres de los dos diseñadores de la WEB, un validador CSS y la licencia de la WEB (Creative Commons).

Una vez descrito el contenido de la WEB pasamos a comentar el desarrollo de las múltiples páginas incluidas en la WEB, para ello nos sobra con hacer una descripción general de ellas, pues funcionan todas de la misma manera, excepto que poseen diferentes contenidos de texto, imágenes, tablas, enlaces y en algún caso videos en formato FLASH.

- Todas las páginas están basadas en HTML 1.0 transicional, creadas a través de la plantilla que hemos descrito anteriormente, en las que incluimos básicamente texto e imágenes para describir cada apartado correctamente.
- Cada página está dotada de su correspondiente título, tanto en el encabezado como en la barra del navegador.
- Nos hemos limitado a trabajar con tablas para centrar las imágenes y los videos con mayor sencillez y comodidad (no incluimos etiquetas DIV para el posicionamiento).
- Por último y como pilar central del diseño, hemos trabajado con estilos CSS específicos, creados para cada tipo de objeto insertado en la WEB (texto, imágenes, videos, links…etc.).

Con todo esto tenemos una WEB sencilla, fácil de manejar y modificar, en la que incluimos los contenidos de una forma ordenada, clara y concisa.

Como último aspecto en este apartado, cabe destacar que los formularios que hemos integrado en la WEB, han sido creados mediante la aplicación "HotPotatoes 6", y cuentan con estilos CSS y funciones JavaScript independientes de la propia programación de la WEB, para que puedan ser ejecutados sin alterar el funcionamiento de la WEB (están basados en HTML 1.1).

2.4. TRATAMIENTO DE IMÁGENES.

Todas las imágenes han sido optimizadas para la WEB con Photoshop CS5.1.

Para la mayoría de las páginas, las imágenes han sido obtenidas de la propia WEB del fabricante (www.microchip.com) y posteriormente traducidas a nuestro idioma (Español) y

retocadas para obtener la mayor calidad posible con el menor peso (que se abran rápidamente en nuestra WEB).

Otra parte de las imágenes ha sido creada por nosotros, para ampliar la parte visual de la WEB, basándonos en la teoría a la que acompañan.

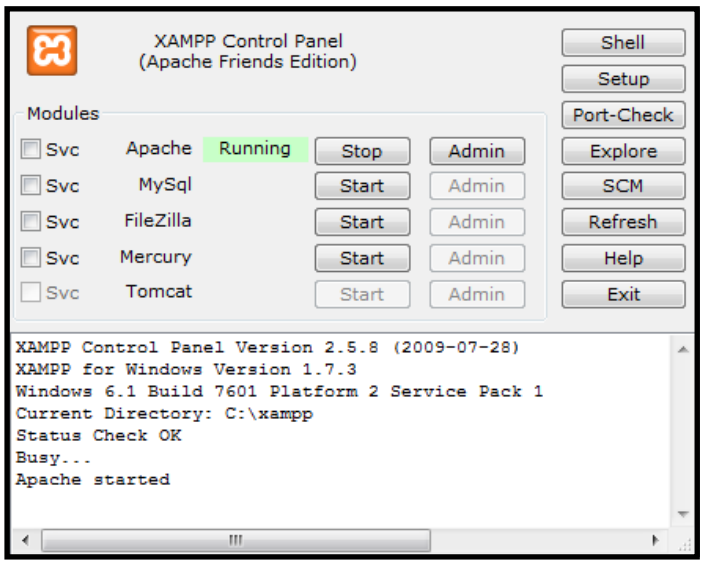
2.5. ESTILOS CSS.

Para la creación de la Web nos hemos ayudado de distintos estilos CSS, para ello lo que hemos hecho es crear archivos "*.CSS", independientes para cada parte de la WEB, como se muestra a continuación:

- *Estilos.css:* Incluye los estilos creados para el diseño general de la WEB (textos, imágenes, vídeos, tablas…etc.).
- *SpryMenuBarHorizontal.*css: Incluye los estilos propios del menú desplegable de SPRY, que nosotros hemos modificado para adecuar el estilo del menú al de la WEB.
- *Cuestionario.*css: Incluye los estilos creados para el diseño de los cuestionarios.

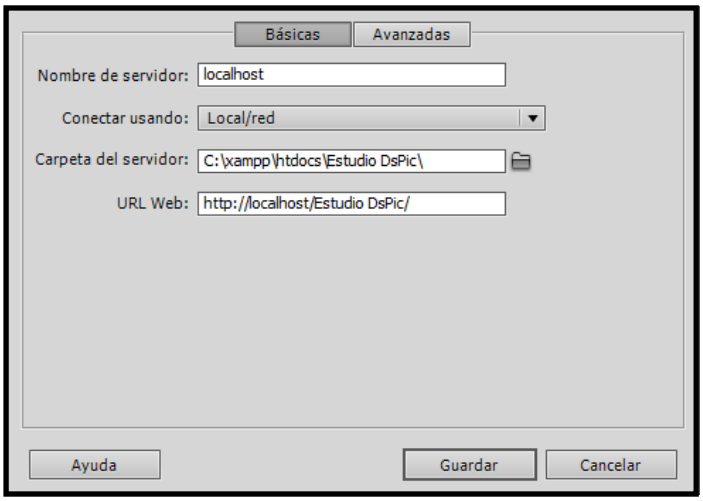
2.6. SERVIDORES DE PRUEBA: LOCAL Y REMOTO.

- *Servidor Local lanzado desde Dreamweaver:* Se trata de un servidor llamado "Apache" con licencia GNU (gratuito y de código abierto), incluido en el paquete de aplicaciones del programa "XAMPP", utilizado para trabajar con nuestro sitio de manera local. Ver Img. 2.



Img.2. Captura servidor local APACHE.

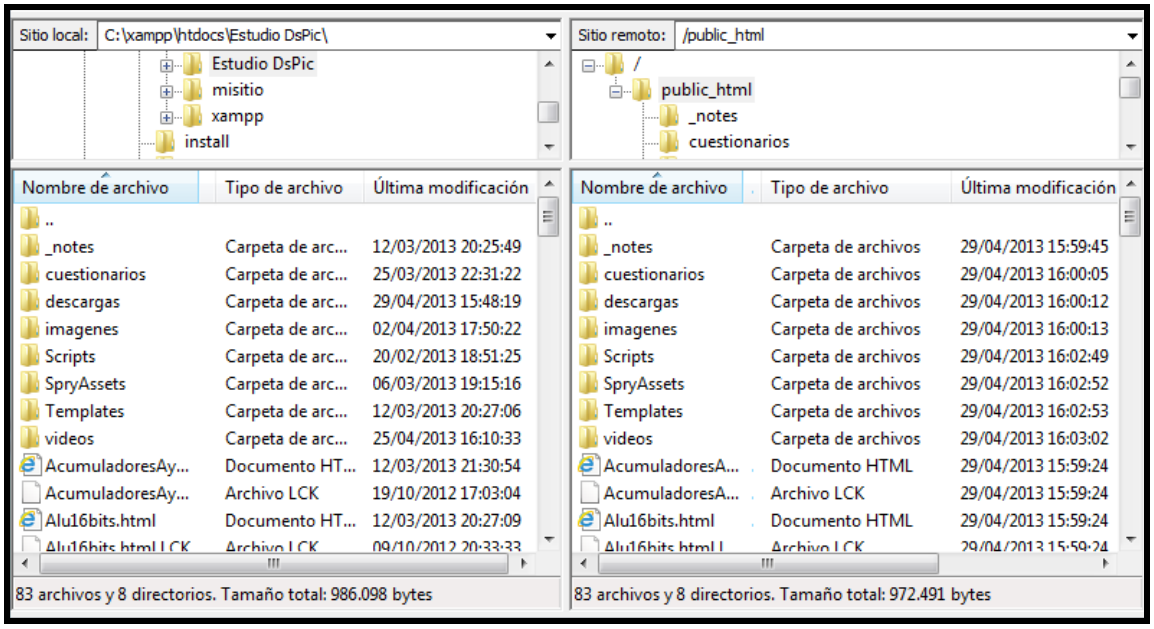
Dicho servidor le hemos configurado desde Dreamweaver para trabajar como servidor local de prueba y poder ir viendo cómo se comporta nuestra WEB sin tener que subir los archivos a un sitio remoto. Ver Img.3.



Img.3. Captura configurador servidor local APACHE.

- *Servidor remoto (WEB):* Almacena principalmente documentos HTML (son documentos a modo de archivos con un formato especial para la visualización de páginas web en los navegadores de los clientes), imágenes, videos, texto, presentaciones, y en general todo tipo de información. Además se encarga de enviar estas informaciones a los clientes. Nosotros tenemos creado un servicio de almacenamiento web y hosting gratuito en www.hostinger.es → *http://www.estudiodspic.16mb.com.*

A través del cliente FTP (Protocolo de transferencia de archivos entre un cliente y un servidor, permitiendo al cliente descargar el archivo desde el servidor o al servidor recibir un archivo enviado desde un cliente) “FileZilla” hemos subido nuestros archivos al servidor WEB simplemente arrastrando los archivos del sitio local al remoto, dentro de la carpeta public_html. Ver Img.4.



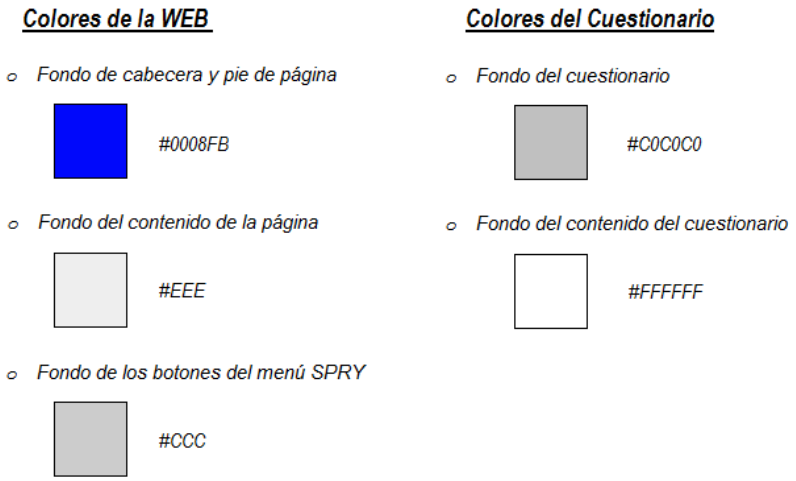
Img.4. Captura servidor FTP FILEZILLA.

Para conectar con el sitio remoto, es necesario introducir los siguientes campos dentro del cliente FTP “FileZilla (proporcionados por nuestro proveedor de almacenamiento WEB, en nuestro caso “Hostinger”):

- *Nombre Host FTP:* estudiodspic.16mb.com
- *Usuario FTP:* u709177076
- *Contraseña:* dsPIC2012

Por último cabe decir que nuestra página WEB ha sido probada en los siguientes navegadores WEB: Internet Explorer, Google Chrome y Firefox. El único navegador que nos respeta la página tal cual la hemos creado es el *Google Chrome*, los demás navegadores crean pequeños errores en la reproducción de los videos FLASH y en la ejecución de los cuestionarios.

2.7. COLORES BÁSICOS DE LA WEB.



2.8. EVALUACIÓN Y PRUEBAS.

Concluida la fase de desarrollo de la WEB, se inicia una etapa final de análisis y comprobaciones, donde se verifica minuciosamente cada sub-apartado o *hilo* en que puede descomponerse el portal. Se examinan asimismo aspectos como la usabilidad, la *performance* del sitio WEB (website) a nivel de ancho de banda consumido, la accesibilidad atendiendo a criterios de resolución de pantalla, entre otros.

De este modo, se han comprobado uno a uno, todas las opciones de actuación disponibles para el usuario en cada una de las secciones de la WEB, siento el resultado el esperado en todos los casos.

Las imágenes y contenidos multimedia han sido optimizados en pro de ocupar el mínimo espacio posible, sin perjudicar la calidad de forma perceptible. Para ello se han buscado los formatos de almacenamiento más apropiados para cada elemento, así como también se ha recurrido a cierto grado de compresión, según el caso.

Todas estas medidas, redundan en un mejor rendimiento (performance) en lo que se refiere a la experiencia de navegación por el website, requiriendo un menor ancho de banda y por lo tanto, menos demora en los procesos.

Se ha recurrido a aplicaciones de terceros o a consorcios internacionales como ahora W3C, con el fin de ratificar la corrección del código desde las perspectivas más habituales y potencialmente más recurridas (evitar enlaces rotos, cumplimiento de estándares, compatibilidad con navegadores).

Se ha introducido el concepto de reusabilidad, con el fin de que el usuario consiga "comprender" los aspectos de funcionamiento y comportamiento, en el menor tiempo posible. De esta forma, son muy aparentes los diseños guardando uniformidad en sus elementos, tanto en los cuestionarios como en la gran parte de elementos interactivos del resto de la WEB.

2.8.1. VALIDACIÓN CSS.

El World Wide Web Consortium (W3C) es un consorcio internacional que produce recomendaciones para la World Wide Web, entre otras, las conocidas validaciones de código orientadas a conseguir una web más “limpia” y “accesible”. En este sentido, estas validaciones garantizaran, en otro orden de cosas, que indistintamente del navegador web empleado, una página web sea representada correctamente.

La implementación del código CSS del portal objeto de estudio ha sido validada con éxito utilizando el Servicio de Validación del propio W3C, obteniendo la calificación de “CSS Versión 3 valido”, sin ningún error encontrado. Ver Img.5.

Img.5. Captura servicio de validación de CSS.

Para realizar la prueba de validación anterior, se ha recurrido a un servicio de alojamiento gratuito (hostinger), transfiriendo allí todos los contenidos del portal y validándolo a continuación empleando el Servicio ofrecido por el W3C.

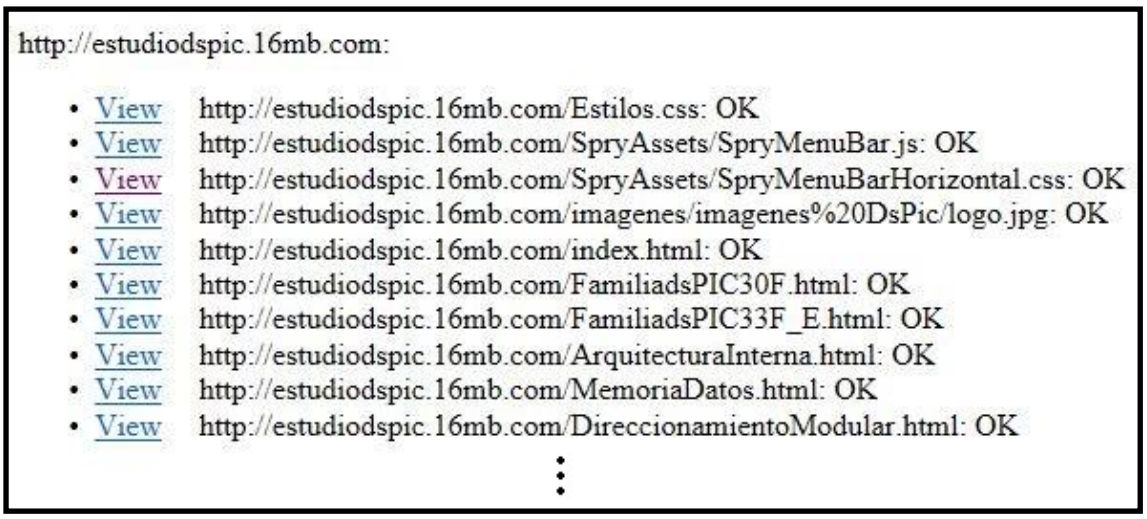
Asimismo, se ha ubicado en la WEB, el icono que el propio consorcio W3C pone a disposición de cualquier desarrollador web, el cual representa un indicativo para que los potenciales usuarios del portal web comprueben que el mismo cumple con el ejemplo, abogando por la interoperabilidad y la accesibilidad.

2.8.2. VERIFICACIÓN DE ENLACES.

La World Wide Web es volátil, un lugar en constantes cambios. Páginas y Websites enteros cambian, se mueven y desaparecen a diario, con el potencial riesgo de enlaces “muertos” que esto conlleva.

Existen diversas aplicaciones web (online) que chequean y verifican la cantidad de enlaces inválidos (que ya no “apuntan” a una página valida), indicando además cuales es su ruta. Una de estas aplicaciones de gran popularidad es que puede encontrarse en página WEB www.anybrowser.com. Ver Img.6. e Img.7. En resumen, un total de 63 enlaces referenciados, todos ellos válidos, apuntados desde el portal web objeto de estudio.

Img.6. Captura aplicación validación de enlaces.



Img.7. Captura resumen validación de enlaces de nuestra WEB.

2.8.3. ANÁLISIS DE RESOLUCIONES DE PANTALLA.

Durante el desarrollo de la aplicación, se buscó en todo momento que la misma pudiera ser visualizada de forma correcta en ordenadores portátiles y de sobremesa (de 15,4" o superiores), puesto que él fin de la WEB es educativo y trabajar en pantallas más pequeñas nos resultaría bastante incómodo (ordenadores portátiles de tamaños reducidos, popularmente

conocidos como notebooks y sus análogos, donde el tamaño de pantalla no acostumbra a superar las 10" y una resolución de 1024 x 768 ppp), por lo que hemos optimizado la WEB para resoluciones de 1280 x 800 ppp o superiores.

En el momento de la presente edición, el ranking mundial de resoluciones de pantalla (Junio 2012) más extendido queda como sigue:

1. 1366×768: 14.56%
2. 1024×768: 13.39%
3. 1280×800: 11.80%
4. 1280×1024: 7.97%
5. 1440×900: 6.21%

De seguir la tendencia de crecimiento marcada por las últimas clasificaciones 1024×768 podría incluso perder en los próximos meses su segunda posición frente a 1280×800.

2.8.4. ESTUDIO DE NAVEGADORES WEB.

Otro aspecto fundamental a evaluar redunda en la importancia de que nuestra aplicación sea visualizada y se comporte con normalidad y de forma análoga por cualquier navegador web disponible, preferentemente por aquellos que ostentan un mayor porcentaje de cuota de utilización por parte de usuarios.

En el momento de edición de esta memoria, el ranking mundial (Top 5) de los navegadores más utilizados queda como sigue:

1º) Internet Explorer: 55.83%
2º) Mozilla Firefox 20.21%
3º) Google Chrome 16.45%
4º) Safari 5.31%
5º) Opera 1.74%

La aplicación ha sido utilizada y testeada intensamente en todos los casos anteriores, siendo el resultado satisfactorio y según lo previsto al 100% con Google Chrome, apareciendo pequeños errores en la visualización de los cuestionarios y en la ejecución de los videos en formato FLASH con los demás navegadores.

2.9. PROBLEMAS AL REALIZAR LA WEB.

En primer lugar tuvimos que realizar un estudio de cómo crear páginas WEB, además de familiarizarnos con todos los nuevos conceptos que la programación tanto HTML, CSS y javascript incorpora, puesto que no teníamos conocimientos previos sobre estos tipos de lenguajes de programación WEB.

Una vez adquiridos los conocimientos anteriormente descritos no pusimos a fondo con el manejo de Dreamweaver, herramienta que nos ha facilitado enormemente el trabajo que hay que realizar en este tipo de proyectos, puesto que se puede trabajar en el modo de diseñador WEB sin tener que entrar tan a fondo en la programación puramente dicha.

Como hemos comentado en apartados anteriores, hemos realizado una WEB bastante simple y de fácil manejo, puesto que solo la hemos usado de soporte para realizar un estudio completo y detallado de los microcontroladores dsPIC, que en realidad es la parte más importante de nuestro proyecto.

Por lo demás no hemos tenidos excesivos problemas a la hora de realizar la WEB, simplemente algunos pequeños errores en el código HTML que han sido solventados a base de paciencia. Otro pequeño inconveniente fue la visualización de nuestra página en los distintos navegadores, puesto que algunos elementos no se posicionan igual en un navegador que en otro, esto no tiene una solución directa, sino que hemos intentado mantener un equilibrio para su ejecución en los diferentes navegadores (Internet Explorer, Google Chrome, Firefox, etc.).

Como punto final cabe destacar que en un primer instante pensamos en hacer el apartado del cuestionario aparte en otro enlace, pero nos volvimos atrás y creímos que era mejor añadir un mini cuestionario al final de cada tema o bloque para que el usuario, después de leer y comprender la teoría de la página pudiera hacer una evaluación de sus conocimientos. Al igual que nos pasó con el cuestionario, a la hora de colocar las aplicaciones vimos que era mejor después de cada tema o bloque para que se viera el ejemplo en el mismo instante de haber leído la teoría.

2.10. PROGRAMAS ÚTILIZADOS.

Para la creación de la página Web hemos utilizado varios programas, los cuales nos ha mantenido ocupados un tiempo hasta que hemos conseguido saber cómo funcionan. Hemos tenido que estudiar diferentes manuales y ver ejemplos de otras web para poder completar la página web.

2.10.1. DREAMWEAVER.

Adobe® Dreamweaver® CS5.5 es una aplicación en forma de estudio (basada en la forma de estudio de Adobe Flash) que está destinada a la construcción, diseño y edición de sitios, videos y aplicaciones Web basados en estándares. Creado inicialmente por Macromedia (actualmente producido por Adobe Systems) es el programa más utilizado en el sector del diseño y la programación web, por sus funcionalidades, su integración con otras herramientas como Adobe Flash y, recientemente, por su soporte de los estándares del World Wide Web Consortium.

Sus principales competidores son Microsoft Expression Web y BlueGriffon (que es de código abierto) y tiene soporte tanto para edición de imágenes como para animación a través de su integración con otras. Hasta la versión MX, fue duramente criticado por su escaso soporte de los estándares de la web, ya que el código que generaba era con frecuencia sólo válido para Internet Explorer y no validaba como HTML estándar. Esto se ha ido corrigiendo en las versiones recientes.

Se vende como parte de la suite Adobe Creative Suite. A partir de la compra de Macromedia por parte de Adobe. Las letras CS significan Creative Suite.

La gran ventaja de este editor sobre otros es su gran poder de ampliación y personalización del mismo, puesto que en este programa, sus rutinas (como la de insertar un hipervínculo, una imagen o añadir un comportamiento) están hechas en Javascript-C, lo que le ofrece una gran flexibilidad en estas materias. Esto hace que los archivos del programa no sean instrucciones de C++ sino rutinas de Javascript que hace que sea un programa muy fluido, que todo ello hace, que programadores y editores web hagan extensiones para su programa y lo ponga a su gusto.

Las versiones originales de la aplicación se utilizaban como simples editores WYSIWYG. Sin embargo, versiones más recientes soportan otras tecnologías web como CSS, JavaScript y algunos frameworks del lado servidor.

Dreamweaver ha tenido un gran éxito desde finales de los años 1990 y actualmente mantiene el 90% del mercado de editores HTML. Esta aplicación está disponible tanto para la plataforma MAC como para Windows, aunque también se puede ejecutar en plataformas basadas en UNIX utilizando programas que implementan las API's de Windows, tipo Wine.

Como editor WYSIWYG que es, Dreamweaver permite ocultar el código HTML de cara al usuario, haciendo posible que alguien no entendido pueda crear páginas y sitios web fácilmente sin necesidad de escribir código.

Algunos desarrolladores web criticaban esta propuesta ya que crean páginas HTML más largas de lo que solían ser al incluir mucho código inútil, lo cual va en detrimento de la ejecución de las páginas en el navegador web. Esto puede ser especialmente cierto ya que la aplicación facilita en exceso el diseño de las páginas mediante tablas. Además, algunos desarrolladores web han criticado Dreamweaver en el pasado porque creaba código que no cumplía con los estándares del consorcio Web (W3C).

No obstante, Adobe ha aumentado el soporte CSS y otras maneras de diseñar páginas sin tablas en versiones posteriores de la aplicación, haciendo que se reduzca el exceso de código.

Dreamweaver permite al usuario utilizar la mayoría de los navegadores Web instalados en su ordenador para previsualizar las páginas web. También dispone de herramientas de administración de sitios dirigidas a principiantes como, por ejemplo, la habilidad de encontrar y reemplazar líneas de texto y código por cualquier tipo de parámetro especificado, hasta el sitio web completo. El panel de comportamientos también permite crear JavaScript básico sin conocimientos de código.

Con la llegada de la versión MX, Macromedia incorporó herramientas de creación de contenido dinámico en Dreamweaver. En lo fundamental de las herramientas HTML WYSIWYG, también permite la conexión a Bases de Datos como MySQL y Microsoft Access, para filtrar y mostrar el contenido utilizando tecnología de script como, por ejemplo, ASP, ASP.NET, ColdFusion, JSP (JavaServer Pages) y PHP sin necesidad de tener experiencia previa en programación.

Un aspecto de alta consideración de Dreamweaver es su funcionalidad con extensiones. Es decir, permite el uso de "Extensiones". Las extensiones, tal y como se conocen, son pequeños programas, que cualquier desarrollador web puede escribir (normalmente en HTML y Javascript) y que cualquiera puede descargar e instalar, ofreciendo así funcionalidades añadidas a la aplicación. Dreamweaver goza del apoyo de una gran comunidad de desarrolladores de extensiones que hacen posible la disponibilidad de extensiones gratuitas y de pago para la mayoría de las tareas de desarrollo web, que van desde simple efectos rollover hasta completas cartas de compra.

2.10.2. ADOBE PHOTOSHOP.

Adobe® Photoshop® CS5 es el nombre o marca comercial oficial que recibe uno de los programas más populares de la casa, Adobe Systems, junto con sus programas hermanos Adobe Illustrator y Adobe Flash, y que se trata esencialmente de una aplicación informática en forma de taller de pintura y fotografía que trabaja sobre un *"lienzo"* y que está destinado para la edición, retoque fotográfico y pintura a base de imágenes de mapa de bits. Su nombre en español significa literalmente "tienda de Fotos" pero puede interpretarse como "taller de foto". Su capacidad de retoque y modificación de fotografías le ha dado el rubro de ser el programa de edición de imágenes más famoso del mundo.

Actualmente forma parte de la familia Adobe Creative Suite y es desarrollado y comercializado por Adobe Systems Incorporated inicialmente para computadores Apple pero posteriormente también para plataformas PC con sistema operativo Windows. Su distribución viene en diferentes presentaciones, que van desde su forma individual hasta como parte de un paquete siendo estos: Adobe Creative Suite Design Premium y Versión Standard, Adobe

Creative Suite Web Premium, Adobe Creative Suite Production Studio Premium y Adobe Creative Suite Master Collection.

2.10.3. FILEZILLA.

FileZilla es un cliente FTP multiplataforma de código abierto y software libre, licenciado bajo la Licencia Pública General de GNU. Soporta los protocolos FTP, SFTP y FTP sobre SSL/TLS (FTPS).

Inicialmente fue diseñado para funcionar en Microsoft Windows, pero desde la versión 3.0.0, gracias al uso de wxWidgets, es multiplataforma, estando disponible además para otros sistemas operativos, entre ellos GNU/Linux, FreeBSD y Mac OS X.

Características:

- Administrador de sitios: permite a un usuario crear una lista de sitios FTP con sus datos de conexión, como el número de puerto a usar, o si se utiliza inicio de sesión normal o anónima. Para el inicio normal, se guarda el usuario y, opcionalmente, la contraseña.
- Registro de mensajes: se muestra en la parte superior de la ventana. Muestra en forma de consola los comandos enviados por FileZilla y las respuestas del servidor remoto.
- Vista de archivo y carpeta: situada en la parte central de la ventana, proporciona una interfaz gráfica para FTP. Los usuarios pueden navegar por las carpetas, ver y alterar sus contenidos tanto en la máquina local como en la remota, utilizando una interfaz de tipo árbol de exploración. Los usuarios pueden arrastrar y soltar archivos entre los ordenadores local y remoto.
- Cola de transferencia: situada en la parte inferior de la ventana, muestra en tiempo real el estado de cada transferencia activa o en cola.

2.10.4. XAMPP CONTROL PANEL.

XAMPP es un servidor independiente de plataforma, software libre, que consiste principalmente en la base de datos MySQL, el servidor web Apache y los intérpretes para lenguajes de script: PHP y Perl. El nombre proviene del acrónimo de X (para cualquiera de los diferentes sistemas operativos), Apache, MySQL, PHP, Perl.

El programa está liberado bajo la licencia GNU y actúa como un servidor web libre, fácil de usar y capaz de interpretar páginas dinámicas. Actualmente XAMPP está disponible para Microsoft Windows, GNU/Linux, Solaris y MacOS X.

Características y requisitos:

XAMPP solamente requiere descargar y ejecutar un archivo zip, tar o exe, con unas pequeñas configuraciones en alguno de sus componentes que el servidor Web necesitará. XAMPP se actualiza regularmente para incorporar las últimas versiones de Apache/MySQL/PHP y Perl. También incluye otros módulos como OpenSSL y phpMyAdmin. Para instalar XAMPP se requiere solamente una pequeña fracción del tiempo necesario para descargar y configurar los programas por separado.

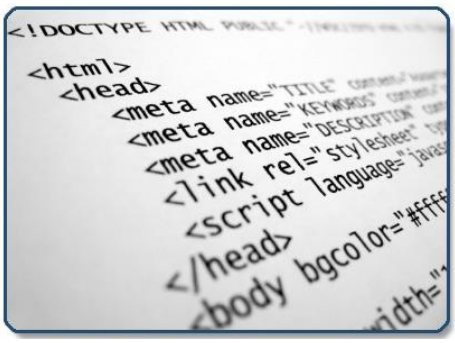
2.11. TECNOLOGÍAS UTILIZADAS.

A continuación serán descritas con mayor nivel de profundidad, algunas de las tecnologías citadas anteriormente.

2.11.1. HTML.

HTML, siglas de HyperText Markup Language (Lenguaje de Marcado de Hipertexto), es el lenguaje que predomina en el desarrollo de páginas web. Gracias a él, es posible no sólo describir el contenido de texto de una página web sino también su forma o estructura, permitiendo incrustar imágenes y otros objetos en la misma.

La historia del estándar se remonta a principio de 1990, cuando Tim Berners-Lee define el HTML como un subconjunto de un conocido lenguaje de etiquetas denominado SGML y además crea algo más valioso aún, el World Wide Web.



Este lenguaje se escribe a base de etiquetas delimitadas por corchetes angulares (< , >). Cuando se utiliza una etiqueta, habitualmente debe utilizarse otra de cierre, la cual antepondrá el símbolo de barra (/). Sirva un ejemplo:

```
<p>Esto está escrito con una fuente normal mientras que <strong>ésto está escrito con una fuente más gruesa o negrita</strong></p>
```

El párrafo anterior será mostrado como:

Esto está escrito con una fuente normal mientras que **ésto está escrito con una fuente más gruesa o negrita**

El lenguaje HTML puede ser escrito y editado con cualquier editor de textos sencillo, aunque habitualmente se emplean editores potentes que permiten visualizar el resultado de lo que se está haciendo así como identificar errores de programación (en los que se encuentra el Dreamwearver).

Una página web HTML debe contener una estructura mínima de etiquetas como sigue:

```
<html>
    <head>
        <title>Aquí iría el título de la página</title>
    </head>
    <body>
        Aquí debe ir el contenido de la página
    </body>
</html>
```

El diseño en HTML debe respetar unos criterios de accesibilidad web, siguiendo unas pautas o normativas. Se encuentra disponible y desarrollado por el W3C a través de las Pautas de Accesibilidad al Contenido Web 1.0 WCAG (actualizadas recientemente con la especificación 2.0).

2.11.2. CSS.

El lenguaje HTML está algo limitado si queremos emplearlo para definir la forma de un documento, ya que no fue concebido precisamente para este fin. Aun así, durante mucho tiempo los diseñadores han debido recurrir a “trucos” para salvar esta dificultad, causando a menudo problemas en las páginas a la hora de su visualización en distintas plataformas.

CSS u hojas de estilo en cascada (Cascading Style Sheets), es un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML. Lo que se pretende con esta tecnología es separar la estructura de una página web de su presentación, esto es, el contenido en sí a mostrar de la forma con que se desea que aparezca.

CSS puede incorporarse en el mismo documento HTML o bien adjuntarse como un documento por separado (obviamente el documento HTML deberá entonces referenciarle).

Enumeraremos algunas de las ventajas de CSS:

- Podemos definir la forma de presentación de un website entero, de sólo una página o una porción de la misma. Se agilizan por tanto enormemente, las tareas de modificación: por ejemplo, bastaría con editar una hoja de estilo CSS para que tuviera una repercusión inmediata en todas las páginas del website.
- El documento HTML es más fácil de entender, pues conseguimos con CSS separar la forma del contenido (el código que hace referencia a la forma, ya no estará entremezclado con el código restante).

- Una misma página Web puede ser mostrada de forma correcta en diversos dispositivos de salida: un móvil, una impresora, una PDA. Para ello bastará con sustituir la hoja de estilo empleada.
- Aumento considerable de la accesibilidad, ya que un navegador web permite a un usuario especificar su propia hoja de estilo local. Idóneo por tanto para usuarios con deficiencias visuales (hojas que aumentan el tamaño de letra o la combinación de colores).

2.11.3. JAVASCRIPT.

JavaScript es un lenguaje de scripting, esto es, un archivo de órdenes, generalmente almacenado en formato de texto plano, que representa a un programa habitualmente de complejidad sencilla que puede efectuar diversas tareas como combinar componentes o bien interactuar con el sistema operativo o con el usuario, motivo por el que es frecuentemente recurrido para diseñar interfaces de usuario.

JavaScript es interpretado por la amplia totalidad de navegadores web modernos. Puede incluirse en cualquier documento y es compatible con HTML en el navegador del cliente. La W3C define que el método óptimo para utilizar JavaScript es incluirlo en un archivo externo, por razones de accesibilidad y rendimiento.

Fue desarrollado por Brendan Eich de Netscape con el nombre de Mocha, el cual fue renombrado posteriormente a LiveScript, para finalmente quedar como JavaScript coincidiendo con el anuncio de que Netscape soportaría oficialmente Java en su, por aquel tiempo, popular navegador web, estrategia que no estuvo exenta de algunas polémicas. Para el portal objeto de estudio, JavaScript será utilizado principalmente para hacer emerger una sub-ventana o ventana-hija, por ejemplo, para mostrar la calificación obtenida en los cuestionarios.

3. ESTUDIO DEL MICROCONTROLADOR dsPIC.

Aunque el estudio del microcontrolador dsPIC pueda parecer tedioso y extenso, es la parte que “menos nos ha costado”, puesto que es un trabajo de redacción y estudio de una serie de libros y documentación propia de la página web de microchip y su datasheet. Es un temario muy extenso, y en este estudio se profundiza bastante en el microcontrolador, por lo que, aunque a nuestro parecer es la parte menos compleja, aun así es un pilar fundamental del proyecto y nos ha llevado un espacio de tiempo considerable.

La mayor parte de la información reflejada en este proyecto, está basada en los libros: *MICROCONTROLADORES AVANZADOS DSPIC* y *DISEÑO PRACTICO DE APLICACIONES DSPIC*, de estos libros hemos sacado prácticamente toda la teoría necesaria para desarrollar cada apartado del estudio, mientras que para las especificaciones, programación, registros y

otros datos más técnicos nos hemos orientado del propio *DATASHEET* que nos proporciona microchip.

El estudio teórico va enfocado al dsPIC30F, que es el microcontrolador más completo y que tiene el mayor número de periféricos y posibilidades de la familia 30F, mientras que las prácticas en PROTEUS las hemos enfocado en todo momento hacía la familia dsPIC33F, que aunque es un microcontrolador menos completo, cumple con casi todas las características del dsPIC30F, y es el que podíamos simular correctamente con PROTEUS.

3.1. ESTUDIO DEL dsPIC30F6010.

3.1.1. Memoria de Datos.

Es de tipo SRAM y se estructura de la siguiente manera: Un *Espacio SFR de 2KB* destinado a los registros específicos de control. Dos *espacios denominados X e Y* que permiten acceso simultáneo y que pueden alcanzar hasta 8KB de capacidad, siendo 16 bits el tamaño de todas sus posiciones, además funcionan de forma independiente al tener su propia AGU (Unidad de generación de direcciones). Para una instrucción de tipo MCU (no DSP) los espacios X e Y se convierten en un solo espacio lineal. Y por último una *Zona de datos X de 32 KB* opcionalmente mapeable en la memoria de programa mediante la "Visibilidad de Espacio de Programa (PSV)", que permite acceso transparente de constantes desde el espacio de datos X sin necesidad de emplear instrucciones especiales como las de Tabla.

Para el direccionamiento de los datos se pueden utilizar los modos clásicos de todos los procesadores: Inmediato, inherente, directo, relativo, registro directo, registro indirecto y registro de desplazamiento. Además los espacios X e Y permiten el modo de direccionamiento modular o circular, y solo el espacio X soporta el de inversión de acarreo o "bit reverse", ambos muy interesantes para la rápida implementación de los algoritmos DSP.

3.1.2. Memoria de Programa.

Se divide en dos espacios claramente diferenciados: El *Espacio de Memoria del Usuario* que abarca 2M posiciones de 24 bits (mitad inferior del espacio de memoria). En el espacio de memoria del usuario es donde residen el Vector de Reset (interrupción GOTO que apunta a la primera instrucción del programa); la Tabla de Vectores de Interrupción; la Tabla Alternativa de Vectores de Interrupción (imagen de la Tabla de Vectores de Interrupción); la memoria FLASH para el usuario, cuya capacidad es de 144KB (48K instrucciones de 24 bits cada una); y la zona de datos EEPROM no volátil de 4KB.

El *Espacio de Memoria de Configuración que es* la otra mitad de memoria de programa, donde se encuentran los registros para la configuración del dispositivo, las 32 instrucciones de identificación de unidad (UNITID) y las posiciones ID destinadas a la identificación del dispositivo (DEVID).

3.1.3. Camino de Datos.

Es donde se ejecutan las instrucciones y se procesan los datos. Soporta funciones aritméticas, lógicas, de desplazamiento, movimiento, rotación y manipulación de bits entre otras. Se organizan en 4 bloque principales: El *Banco de registros W* que está formado por 16 registros de trabajo (W) de 16 bits de longitud cada uno, pudiendo contener datos *o* direcciones dependiendo de la instrucción que les utilice. Cada uno tiene asignadas funciones diferentes.

La *ALU de 16 bits,* que es una Unidad Aritmético Lógica de 16 bits de ancho capaz de sumar, restar, desplazar un bit y de realizar operaciones lógicas (AND, OR, EOR). Salvo que se mencione lo contrario, las operaciones aritméticas son en complemento a 2. Dependiendo de la operación, la ALU puede afectar los valores de los bits del registro de estado (SR). La ALU recibe los operandos del banco de registros "W" y de la memoria de datos X a través del bus de datos correspondiente.

El *Motor DSP,* que se trata de un bloque hardware que se alimenta de datos desde el banco de registros W, pero contiene sus propios registros especializados para contener los resultados. Está controlado desde el mismo decodificador de instrucciones que dirige la ALU, por otra parte, todas las direcciones efectivas de los operandos (EAs) se generan en el banco de registros W, por lo que no es posible ejecutar de forma concurrente instrucciones MCU e instrucciones DSP, aunque los recursos de la ALU y del motor DSP pueden ser compartidos por todas las instrucciones. Internamente trabaja con 40 bits, pero externamente se relaciona con buses de 16 bits. El motor DSP se compone de los siguientes recursos: Un multiplicador de alta velocidad de 17-bits x 17-bits, un registro de desplazamiento de 40 bits, un sumador/restador de 40-bits, dos registros acumuladores destino de 40 bits (A y B), la lógica de redondeo y la lógica de saturación.

La *Unidad de división,* que es utilizada para dividir números enteros con y sin signo, como así también para la división de números fraccionales. Las instrucciones de división de números enteros también soportan divisiones extensas, donde el numerador es de un ancho 32 bits y se almacena en 2 registros de trabajo adyacentes, pero el denominador es de un ancho de 16 bits.

3.1.4. Puertas de E/S Multifunción.

Posee 7 puertas de E/S (PUERTA A,…PUERTA G) cuyas líneas controlan varias funciones multiplexadas en el tiempo, además de la propia línea de E/S digital, la mayor parte de ellas destinadas a manejar periféricos integrados, quedando algunas dedicadas al control de los voltajes de alimentación, las interrupciones externas, las señales de reloj y las señales del sistema.

3.1.5. Periféricos diversos.

En cuanto a periféricos y recursos auxiliares el dsPIC30F6010 contiene todos los posibles, existentes en la familia 30F, excepto el módulo DCI utilizado en aplicaciones de audio y contenido en los dsPIC de propósito general.

Contiene 5 *temporizadores* de 16 bits denominados TIMER1, TIMER2,…TIMER5 respectivamente. Tanto los temporizadores TIMER2 y TIMER3 como TIMER4 y TIMER5 pueden fusionarse para obtener 2 temporizadores de 32 bits (TIMER2/3 y TIMER4/5). Todos los temporizadores pueden funcionar en los 4 modos siguientes: Temporizador síncrono, contador síncrono y asíncrono, y contador con disparo por acumulación de tiempo.

Posee un *conversor A/D* de 10 bits de resolución, 16 canales y alta velocidad, que sirve para convertir una señal analógica en un valor digital de 10 bits.

También tiene un *Módulo de Captura* de entrada para medir el tiempo que dura un acontecimiento externo producido por una señal que se aplica en uno de los terminales de entrada del módulo y que se mide con el temporizador *TIMER2 o TIMER3.*

Además de un *Módulo de comparación* de salida cuyo funcionamiento se basa en la comparación del valor almacenado en un registro con el del temporizador (TIMER2 o TIMER3). Cuando coinciden dichos valores se genera un pulso o tren de pulsos en un terminal de salida. Es capaz de generar interrupciones.

Incorpora un *Módulo PWM para el control de motores* que sirve para generar un tren de impulsos de anchura variable que permite variar la potencia que se entrega a un motor y regular así su velocidad con precisión.

Tiene múltiples *Módulos de comunicación:* El *Módulo UART,* destinado a soportar la comunicación serie asíncrona. Funciona de forma bidireccional, adaptándose al trabajo de muchos periféricos. La comunicación se realiza con dos líneas, una para la transmisión UTX y otra para la recepción URX, entrando y saliendo los bits a una frecuencia controlada internamente. El *Módulo SPI,* que consiste en una interfaz serie síncrona empleada para establecer la comunicación con microcontroladores, EEPROM, conversores AD, etc. El *Módulo I²C, e*mpleado en las transferencias de información con otros microcontroladores, registros de desplazamiento, conversores AD, etc. Tiene el hardware necesario para trabajar en modo esclavo y en modo maestro con una interfaz de 16 bits. Todos los elementos que se conectan al bus I²C lo hacen a través de dos líneas (terminales), SDA (Datos) y SCL (Reloj). Permite la transferencia de datos bidireccional entre el modo maestro y esclavo. Y el *Módulo CAN,* que consiste en una interfaz serie muy empleada para la comunicación con otros periféricos u otros modelos de microcontroladores, reduciendo el cableado y evitando las interferencias de los ambientes ruidosos.

Por último encontramos el *Módulo QEI,* que proporciona una cómoda adaptación de los codificadores incrementales usados para la detección de la posición y la velocidad en los sistemas rotacionales propios de los ejes de motores.

3.1.6. Gestión del sistema y la energía.

Se utilizan los siguientes sistemas y recursos: El *sistema Oscilador* que es el encargado de proporcionar la señal de reloj principal y todas las auxiliares que alimentan a los recursos del dispositivo. Para realizar esto se dispone de tres osciladores primarios, un oscilador secundario, dos osciladores internos y un oscilador externo.

El *sistema de Reset, que* contempla todas las fuentes capaces de provocar Reset y controla la señal maestra de Reset del dispositivo (SYSRST#).

Un *Temporizador WDT (Perro Guardián),* destinado a vigilar el procesamiento del flujo de control y reinicializar el procesador cuando se produzca un fallo.

El *Módulo LVD,* basado en un comparador que detecta cuando la VDD del dispositivo cae por debajo de un valor umbral *(VLVD)* generado internamente, el cual es programable durante la ejecución de la aplicación.

Y por último los dos *Modos de Bajo Consumo de Energía,* que son el Modo Sleep (suspensión) en el que el reloj de la CPU y los periféricos dejan de funcionar, y el Modo Idle (Inactividad) en el que la CPU se deshabilita y deja de ejecutar instrucciones, pero los periféricos siguen funcionando, aunque pueden ser desactivados opcionalmente. En este último modo sólo se deshabilita el oscilador principal.

3.1.7. Otros recursos a destacar.

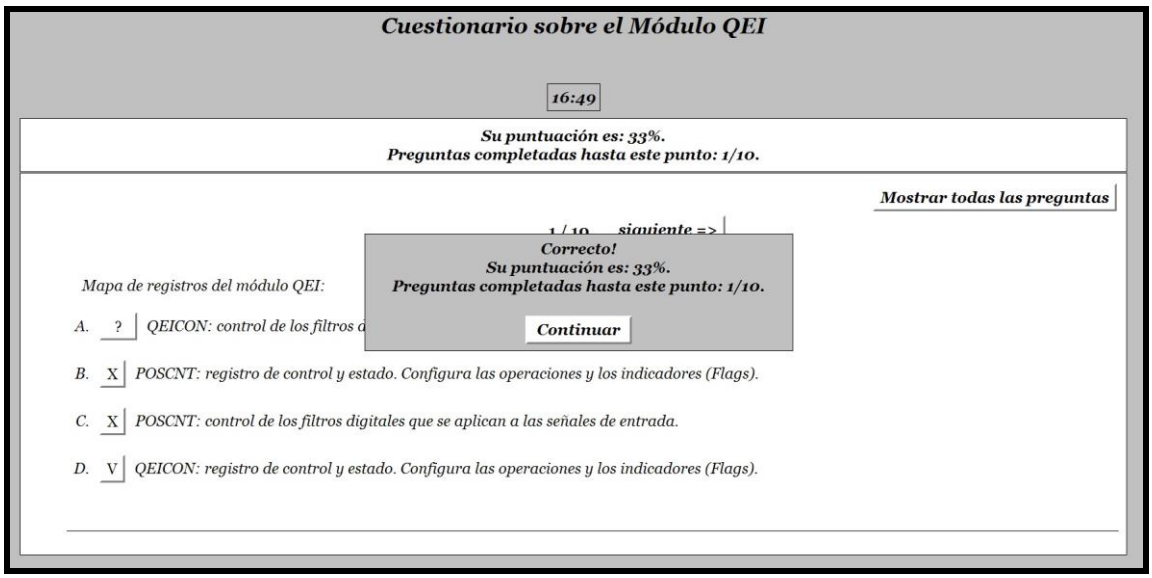
Otros recursos a destacar sobre el *dsPIC30F6010* son las Interrupciones y las Excepciones que son causas que desvían el flujo de control en la ejecución de instrucciones. Las interrupciones son provocadas por acontecimientos externos, como los que originan los periféricos integrados o las señales aplicadas en determinados terminales. Las excepciones se producen automáticamente cuando el procesador detecta algún error o anomalía en la ejecución de una instrucción. Además toda la estructura anteriormente descrita admite operaciones MCU y operaciones DSP con un repertorio de 84 instrucciones, la mayoría de 24 bits de longitud y ejecutables en un ciclo de instrucción.

4. CUESTIONARIO.

Hemos realizado una serie de cuestionarios de cada tema propuesto en el proyecto, a modo de que el alumno/usuario que acceda a nuestra página web pueda comprobar sus conocimientos adquiridos.

El cuestionario esta subido a la página web y esta realizado mediante la herramienta *HOTPOTATOES* (que se explica a continuación). Al final de la visualización del temario de cada parte del estudio podemos acceder al cuestionario, porque entendemos que es el mejor lugar para ponerlo, para evaluar a continuación de cada estudio la comprensión del temario y para mantener un orden dado el tamaño del proyecto.

Dispone de un tiempo específico (2 minutos por cuestión) para realizar el test que ronda entre las 10 y 15 preguntas, todas estas aleatorias y cambiadas el orden de las respuestas y preguntas cada vez que sea ejecutado para poner mayor dificultad a la memorización del test. También dispone de un sistema de corrección por porcentaje de acierto y la visualización de pregunta correcta o incorrecta, además, al finalizar el test te muestra un resumen del mismo con número de aciertos a la 1º. Ver Img.2.



Img.2. Captura ejemplo cuestionario.

4.1. PROGRAMAS UTILIZADOS.

4.1.1. HOTPOTATOES.

Hot Potatoes es un sistema para crear ejercicios educativos que pueden realizar posteriormente a través de la web. Los ejercicios que crea son del tipo respuesta corta, selección múltiple, rellenar los huecos, crucigramas, emparejamiento y variados.

Su licencia no es libre, pero a partir del 1 de septiembre de 2009 se distribuye la versión sin limitaciones a través de la sección descargas de su sitio web.

Hot Potatoes está creado por el centro de humanidades y computación de la Universidad de Victoria, en Canadá. Para asuntos comerciales se ha creado la empresa Half-Baked Software Inc.

Se han observado ciertas incompatibilidades con algunos navegadores como Firefox.

5. APLICACIONES.

Un desarrollo de estas características, el estudio del microcontrolador dsPIC, donde toda la explicación del dispositivo está basada en nombres de registros, modos de funcionamiento y distintas maneras de configurar los terminales, se hace muy complicado de entender y asimilar sin un ejemplo práctico que muestre la manera de programar y porque no, de funcionar.

A través de unos ejercicios sencillos vamos explicar por separado cada periférico y cada módulo del microcontrolador dsPIC, para conocer su funcionamiento, la forma de programarlo e inicializarlo y las posibles aplicaciones reales en las que puede ser utilizado.

5.1 ENTRADAS Y SALIDAS MULTIFUNCIÓN.

En el *dsPIC30f6010* el bloque de E/S multifunción consta de 7 puertas (A, B,…G). Cada una de ellas con un número determinado de terminales para cada función y posibilidad del dispositivo, E/S analógica o digital, entradas de interrupción, de reloj, de alimentación, etc…

5.1.1 DESCRIPCIÓN.

La aplicación consta de un interruptor conectado a una entrada y un LED conectado a una salida del dispositivo. Cada vez que se accione el interruptor, el LED se iluminará (ver Fig. 1).

Este es un ejemplo que aunque sencillo y obvio es necesario para comenzar, puesto que lo primero que se utiliza en un cualquier montaje digital son las entradas y salidas, porque para cualquier programación o configuración del microcontrolador necesitaremos conectar tanto señales analógicas (Ej.: sensor de temperatura) como digitales (Ej.: señal de reloj) a una

o varias entradas del dispositivo para conseguir emitir otra señal analógica o digital por alguno de sus terminales de salida consiguiendo accionar así algún dispositivo externo (Ej.: Un testigo luminoso).

Con este ejemplo veremos la forma de configurar los terminales como entrada o como salida, la estructura de un programa básico, y la forma de programar e introducir las librerías necesarias a la hora de compilar para que el programa recoja la información necesaria sobre el dispositivo.

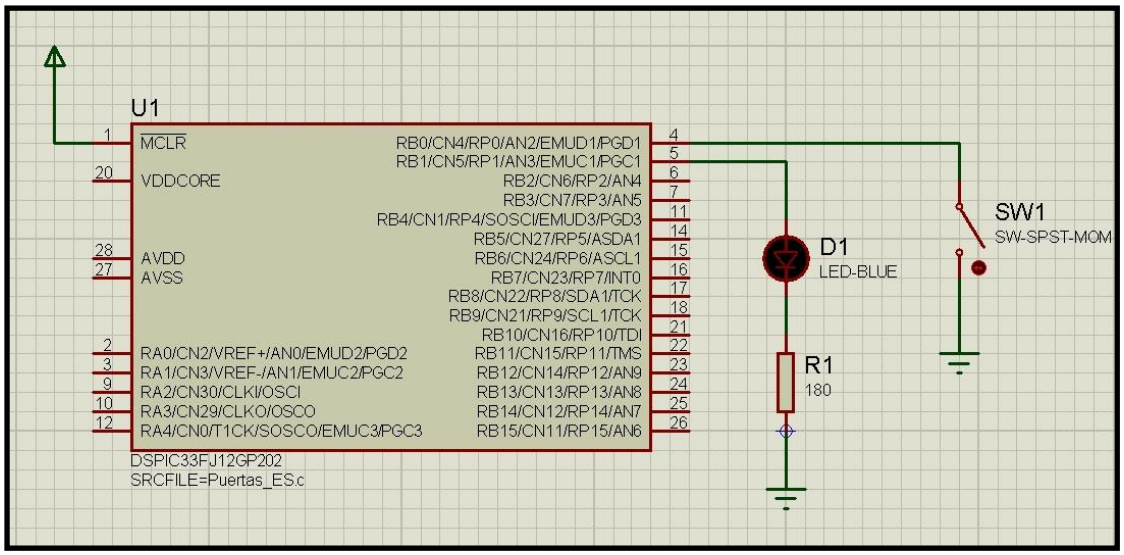


FIG. 1. ESQUEMA ENCENDER Y APAGAR LED. PRACTICA ENTRADA Y SALIDA

5.1.2 DESARROLLO.

La aplicación la hemos desarrollado con el microcontrolador *dsPIC33FJ12MC202*, puesto que en PROTEUS la familia del *dsPIC30F* no existe. En este dispositivo, solo tenemos dos puertas (A y B), pero para la aplicación que vamos a desarrollar nos vale.

Como hemos explicado antes, el programa va a constar de una entrada (con un interruptor) y una salida (con un LED), pues lo primero que necesitamos saber es como se configura el dispositivo para que un determinado terminal de puerta se comporte como entrada o salida de datos. Todo esto se realiza configurando el registro de control TRISx: la “x” determina la puerta con la que estamos trabajando (A, B, …, G) y con los bits asociados al registro elegimos si el terminal de puerta es de entrada o salida. El registro de control TRISx es de 16 bits, cada bit corresponde a un terminal de la puerta “x” respetando el peso. El bit 0 será corresponde con el terminal Rx0 y el bit 15 corresponde con el terminal Rx15 (No todas las puertas están dotadas de 16 terminales). Si el bit es “1” el terminal de puerta estará

configurado como entrada y si es “0” estará configurado como salida. Por defecto todos los terminales de puerta vienen configurados como entrada.

En nuestro caso hemos elegido la puerta B que tiene 16 terminales: vamos a poner “RB0” como entrada (entonces el bit de menos peso de TRISB ira a 1) y “RB1” como salida. Los demás terminales de la puerta B nos dan igual como estén (para este ejemplo), así que les pondremos todos a 0 (como salidas) para que se vea más fácil la transformación a hexadecimal del número correspondiente y que se vea más claramente cuál es la entrada. El registro TRISB entonces quedaría configurado de la siguiente manera: 0000000000000001. Se podría poner directamente ese número en binario: *set_tris_b(0000000000000001),* pero como podéis comprobar resulta muy difícil de leer, entonces lo que se hace es transformarlo a hexadecimal (en grupos de cuatro bits y respetando el peso) *set_tris_b(0x0001).*

Para el funcionamiento del LED solo tenemos que hacer un bucle con unas condiciones, para que el dispositivo detecte el estado de la entrada (RB0) y nos proporcione una salida (RB1). Cuando el interruptor está cerrado (RB0=0) se enciende el LED (RB1=1) y viceversa (si RB0=1 entonces RB1=0).

Código generado:

```
#include<33FJ12MC202.h>

#fuses XT   // Oscilador de cristal para frecuencias entre 4Mhz y 10Mhz.
#fuses NOWDT   // Sin Perro Guardián (WDT).

#use delay( clock = 4000000 )   // Reloj de 4 MHz.

void main()  // Función principal.
{
  set_pullup(TRUE);   // Habilitación resistencias Pull-up.
  set_tris_b(0x01);  // RB0 como entrada y RB1 como salida.

  output_low(PIN_B1);  //Establecer RB1 a nivel bajo (LED apagado).

  while(1)  //Bucle infinito.
  {
    if (input(PIN_B0)== 1)  // Si RB0 está a nivel alto.
      output_low(PIN_B1);  // Ponemos RB1 a nivel bajo (LED apagado).

    else output_high(PIN_B1);  // En caso contrario RB1 a nivel alto (LED encendido).
  }
}
```

5.2 TEMPORIZADORES. TIMER1 DE 16 BITS.

El *dsPIC30F6010* contiene 5 temporizadores de 16 bits denominados TIMER1, TIMER2,…TIMER5. En este ejemplo vamos a utilizar el TIMER1.

5.2.1 DESCRIPCIÓN.

El Timer1 es un temporizador de 16 bits que puede servir como contador de tiempo real, o como un temporizador libre de intervalos/contador. En esta práctica vamos a utilizar el timer1 en modo temporizador de 16bits, el temporizador (registro TMR1) se incrementa en cada ciclo de instrucción hasta el valor precargado en el registro de periodo PR1, luego se pone a ´0´y sigue contando. Se pueden provocar una interrupción cuando hay una coincidencia entre el valor del temporizador (registro TMR1) y el registro de período *PR1* de 16 bits, si además el respectivo bit de indicación de interrupción *T1IF* del registro *IFS0* está activado.

Ahora bien, por defecto el registro de periodo PR1 = FFFFh = 65536d, por lo que debemos precargar un valor en el registro temporizador TMR1 para que podamos temporizar un segundo. Emplearemos el *dsPIC33fj12MC202* para la simulación en PROTEUS y mostraremos como resultado, mediante un reloj/cronometro, el tiempo temporizado para comprobar el resultado (Ver FIG.2).

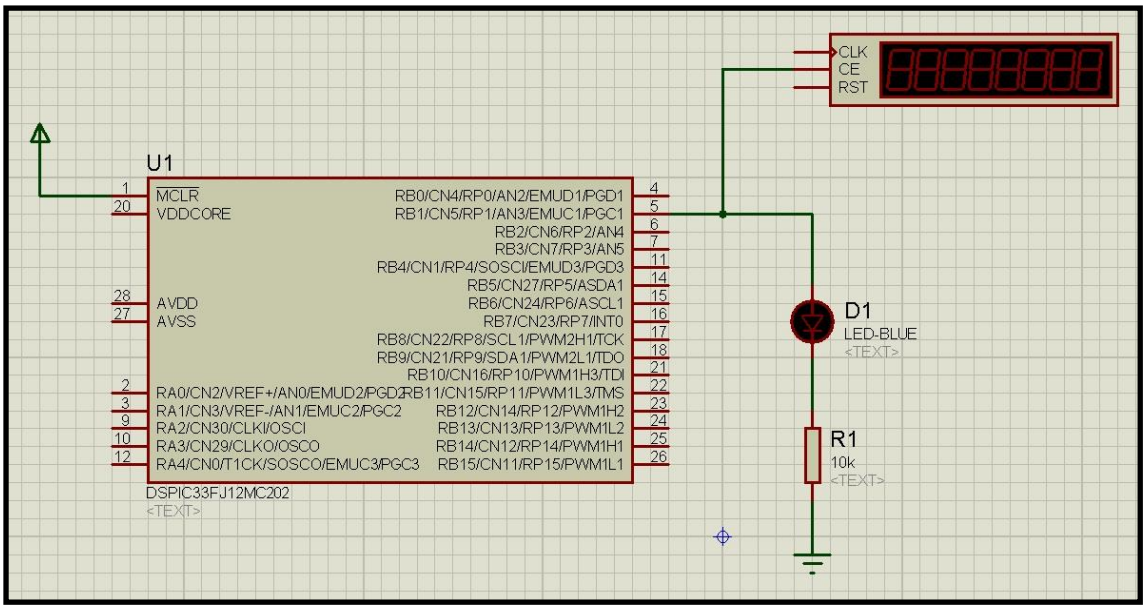


FIG. 2. ESQUEMA EMPLEO DE TEMPORIZADORES.TEMPORIZACIÓN DE 1 SEGUNDO.

5.2.2. DESARROLLO.

Para empezar el desarrollo del código, debemos tener en cuenta las posibilidades del temporizador y a partir de ahí estudiar las posibilidades para llevarlo a cabo. Lo primero que

vamos a hacer es mirar cuales son las máximas temporizaciones de interrupción para el reloj interno seleccionado:

- El timer1 se desborda cuando los valores de los registros *TMR1 y PR1* coinciden, es decir cuando pasamos del valor precargado en *TMR1* a FFFFh=65536d (valor por defecto del registro *PR1*).
- Reloj interno de 4 MHz (elegido por nosotros) → Tcy = 250ns = 1/4 MHz (Tiempo de Ciclo de instrucción).
- El timer1 dispone de los siguientes Prescaler: 1, 8, 64, 256. Con los cuales podemos calcular las máximas temporizaciones de interrupción para reloj interno de 4 MHz, atendiendo a la siguiente ecuación:

$$T_{interrupción} = Tcy\ x\ Prescaler\ x\ (PR1 - TMR1)$$

T_interrupción_máximo = 250ns x Prescaler x (65536-0) → *TMR1=0 y PR1*=65536.

Prescaler 1 → T_interrupción = 16,38ms

Prescaler 8 → T_interrupción=0.131s

Prescaler 64 →T_interrupción=1.048s

Prescaler 256 →T_interrupción=4.194s

- El tiempo que tarda el temporizador (registro TMR1) en incrementar su cuenta en "1" se calcula con la siguiente ecuación:

$$T_{incremento\_TMR1} = Tcy\ x\ Prescaler$$

- El número total de incrementos del temporizador (registro TMR1) necesarios para llegar a la temporización deseada se calcula con la siguiente ecuación:

$$Nº_{incrementos} = PR1 - TMR1$$

- De todo lo anterior deducimos que la temporización de interrupción del timer1 atiende a la siguiente ecuación (la misma que para el cálculo de las temporizaciones máximas):

$$T_{interrupción} = T_{incremento\_TMR1}\ x\ Nº_{incrementos}$$

Ahora que conocemos las posibilidades del temporizador, pasaremos a elegir un prescaler y a calcular el valor de TMR1 para obtener una temporización adecuada. Con el prescaler 1 y 8 no llegamos a un segundo. Vamos a utilizar el prescaler 64 pero en vez de hacer una temporización de un segundo, vamos a emplear un bucle que realice dos veces una temporización de 0,5s. Se podría elegir el prescaler 64 o el 256 y calcular directamente el tiempo para un segundo, pero lo vamos a hacer así para ver que se puede jugar con el código para adaptar las características del dispositivo a nuestras necesidades. Pasemos entonces a calcular el valor del registro TMR1 para un prescaler de 64 y para una temporización de 0,5s.

- Sustituimos en la ecuación anterior:

  $0{,}5s = 250ns\ x\ 64\ (65536 - TMR1) \rightarrow TMR1 = 34286$

Para que el programa repita dos veces la temporización de 0,5s y completar así una temporización de un segundo, solamente necesitaremos un bucle if que repita dos veces la temporización de 0,5s dentro de la rutina de atención a la interrupción (ISR) del Timer1.

Código generado:

```
#include<33FJ12MC202.h>

#fuses XT   // Oscilador de cristal para frecuencias entre 4Mhz y 10Mhz.
#fuses NOWDT   // Sin Perro Guardián (WDT).

#use delay( clock = 4000000 )   // Reloj de 4 MHz.

#use standard_io(b)  // Hace que un terminal de E/S sea entrada o salida cada vez que se utiliza.

int1 cont=0;

#int_TIMER1  // Rutina de atención a la interrupción del Timer1 (ISR).
void TIMER1_isr(void)   // Función de la interrupción por desbordamiento del timer1.
{
  if (cont==1) output_toggle(PIN_B1);   // Cada 2 interrupciones de 0.5s (1s) se conmuta la salida RB1.
  set_timer1 (34286);  // Recarga del TMR1.
  cont++;  // Incremento del contador.
}

void main()  // Función principal.
{
 setup_timer1 (TMR_INTERNAL | TMR_DIV_BY_64); // Reloj interno y preescaler de 64.
 set_timer1 (34286);  // Recarga del TMR1.
 enable_interrupts(INT_TIMER1);  // Habilita la interrupción del timer1.
 enable_interrupts (INTR_GLOBAL);  // Habilita la interrupción general.

 while(1); // Bucle infinito.
}
```

5.3. TEMPORIZADORES. TIMER 2/3 DE 32 BITS.

El módulo Timer 2/3 es un temporizador de 32 bits, que puede ser configurado como dos temporizadores de 16 bits, con modos de funcionamiento seleccionables.

5.3.1. DESCRIPCIÓN:

En el modo de temporizador de 32 bits, que es el modo que vamos a utilizar en esta práctica, el temporizador se incrementa en cada ciclo de instrucción hasta el valor cargado previamente en el registro de periodo combinado PR3/PR2 de 32 bits, posteriormente se reinicia a ´0´y continua contando.

Vamos a realizar una temporización de 2,5s, el circuito dispone de un diodo LED que se iluminara durante dicha temporización, mediante la ISR (rutina de atención a la interrupción) de la interrupción por desbordamiento del timer3, en la que cambiamos el estado del terminal RB8 que está conectado al LED. Emplearemos el *dsPIC33FJ12MC202* para la simulación en PROTEUS y mostraremos como resultado, mediante un reloj/cronometro, el tiempo temporizado para comprobar el resultado (Ver FIG.3).

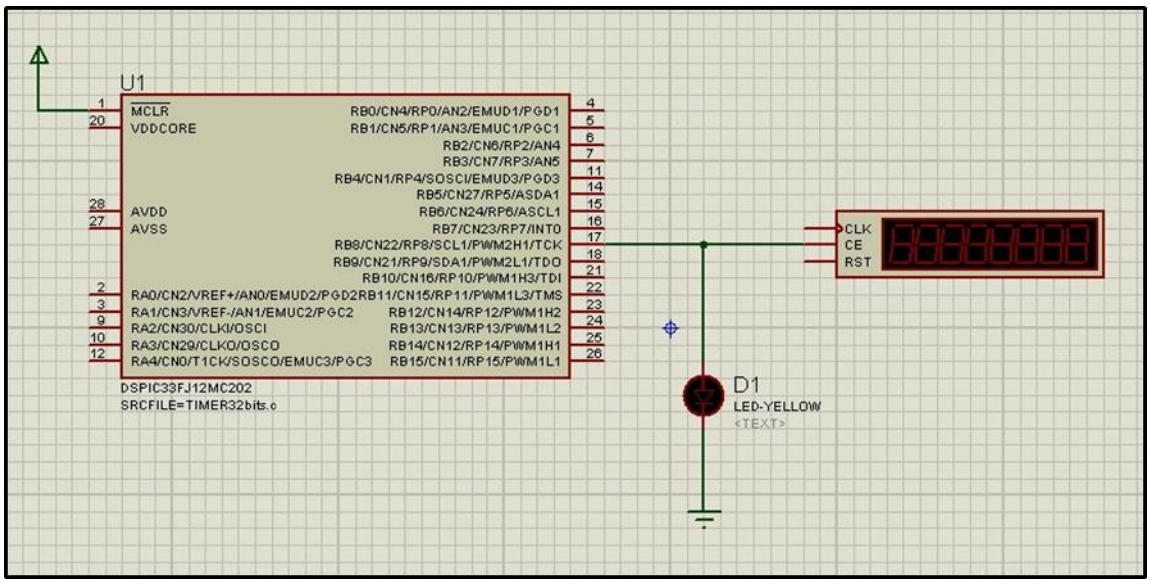


FIG. 3. ESQUEMA EMPLEO DEL TEMPORIZADOR 2/3.TEMPORIZACIÓN DE 2,5 SEGUNDOS.

5.3.2 DESARROLLO:

Como en el ejemplo anterior de la temporización de un segundo, para afrontar el código, necesitaremos estudiar las formulas y posibilidades del temporizador 32 para hacer una selección del valor precarga que nosotros en este ejemplo llamaremos ¨period¨.

El registro del temporizador ¨timer¨ aumenta su cuenta en uno cada Tcy (tiempo de ciclo de instrucción) desde ´0´hasta el valor precargado en el registro “period” de 32 bits.

Por lo tanto lo primero que deberemos hacer es inicializar el registro "timer" a ´0´. La interrupción por desbordamiento del timer 3 se produce cuando: timer = period, una vez alcanzada la cuenta el registro "timer", se pone a ´0´y continua contando.

- La temporización viene dada por:

$$Temporizacion = period\ x\ Tcy$$

- La frecuencia del oscilador elegida es de 4Mhz, el ciclo de instrucción Tcy se calcula:

$$Fcy = \frac{Fcosc}{2} = \frac{4Mhz}{2} = 2$$

$$Tcy = \frac{1}{Fcy} = \frac{1}{2Mhz} = 500ns$$

- El valor que tendremos que meter en la variable "period" será:

$$Temporización = 2{,}5s = period\ x\ 500ns$$

$$period = 5.000.000\ dec = 0x4C4B40\ hex$$

Código generado:

```
#include <p33FJ12MC202.h>
#include <timer.h> // Librería para poder usar los temporizadores.

_FGS(GCP_OFF & GWRP_OFF); // Configuración para protección de RAM.
//GCP_OFF --> Memoria de usuario de programa no está como código protegido.
//GWRP_OFF --> Memoria de usuario de programa no está protegida contra escritura.
_FWDT(FWDTEN_OFF);  // Configuración para selección de Watchdog.
//FWDTEN_OFF --> Temporizador Watchdog activado/desactivado por el usuario mediante software.
_FOSCSEL(FNOSC_PRI); // Configuración para selección de oscilador.
//FNOSC_PRI --> Oscilador Primario (XT, HS, EC).
_FOSC(FCKSM_CSDCMD & POSCMD_XT); // Configuración para selección de oscilador.
//FCKSM_CSDCMD --> Conmutación de fuente de reloj y monitorización de fallo de reloj deshabilitadas.
//POSCMD_HS --> Modo de Oscilador XT.

/****************************************************************** */
void __attribute__((interrupt,auto_psv)) _T3Interrupt(void)
```

```
{

LATBbits.LATB8=~LATBbits.LATB8; // Cambiamos de estado el terminal RB8.
_T3IF = 0; // Borramos flag de interrupción.

}

int main ()
{
// Definición de variables locales.
unsigned int config1, config2;
unsigned long period;
unsigned long timer;

TRISB = 0xFEFF; // Configuro el terminal RB8 como salida.

config1 = (T2_32BIT_MODE_ON & T2_IDLE_STOP & T2_GATE_OFF & T2_PS_1_1 & T2_SOURCE_INT);
//T2_32BIT_MODE_ON -->  Modo temporizador de 32 bits activado.
//T2_IDLE_STOP --> Modo Idle desactivado.
//T2_GATE_OFF --> Acumulación de tiempo en puerta deshabilitada.
//T2_PS_1_1 --> Valor del preescaler del temporizador de 32 bits.
//T2_SOURCE_INT --> Fuente de reloj interna.

config2 = (T3_INT_PRIOR_1 &  T3_INT_ON);
//T3_INT_PRIOR_1 --> Prioridad '1' de la interrupción del temporizador 3.
//T3_INT_ON --> Interrupción del temporizador 3 habilitada.

period = 0x004C4B40; // Metemos en el registro de periodo de 32 bits un valor para conseguir los 2.5 segundos.
timer = 0x00; // Inicializamos a 0 el temporizador.

OpenTimer23(config1,period);    // Esta función configura el temporizador de 32 bits.

ConfigIntTimer23(config2);  // Esta función configura la interrupción del temporizador de 32 bits.

WriteTimer23(timer);     // Esta función escribe un valor de 32 bits dentro del registro "Timer" del temporizador
de 32 bits.

while(1){};

}
```

5.4. CONVERSOR A/D DE 10 BITS.

El dsPIC30F6010 posee un conversor A/D de 10 bits de resolución, 16 canales y alta velocidad, que sirve para convertir una señal analógica de entrada en un valor digital de 10 bits.

Posee las siguientes características: Velocidad máxima de conversión de 1Msps con una impedancia máxima de 500Ω, tiempo de muestreo de 154ns, capacidad de hasta 4 terminales de entrada para muestreo simultáneo, exploración automática del canal, buffer de 16 palabras para el almacenamiento de los resultados de la conversión, 4 amplificadores de captura y mantenimiento (“Sample & Hold”) y +/- 1 bit de exactitud.

5.4.1 DESCRIPCIÓN.

En esta práctica vamos a implementar el conversor A/D de una manera muy gráfica, vamos a transformar la tensión de un potenciómetro (señal analógica) a una emulación de VUmeter en la que por medio de 8 diodos LED representara la tensión del potenciómetro (señal digital).

El funcionamiento es muy sencillo, en el potenciómetro se puede subir o bajar la tensión de salida, la tensión va desde 0v (terminal AVSS) en la que todos los LED´s estarán apagados, hasta 5v (terminal AVDD) en la que los 8 LED´s estarán encendidos, y entre medias de esa tensión 0-5v se irán encendiendo o apagando los LED´s si subimos o bajamos la tensión del potenciómetro respectivamente.

El terminal encargado de transportar la tensión del potenciómetro (entrada analógica) será RA0/AN0 y los terminales que sacan la transformación de esa entrada serán los ocho primeros terminales del puerto B, es decir, desde RB0 hasta RB7 (salida digital). Para la simulación del código en PROTEUS, utilizamos el *dsPIC33FJ12MC202* en el montaje del siguiente circuito. Fig.4

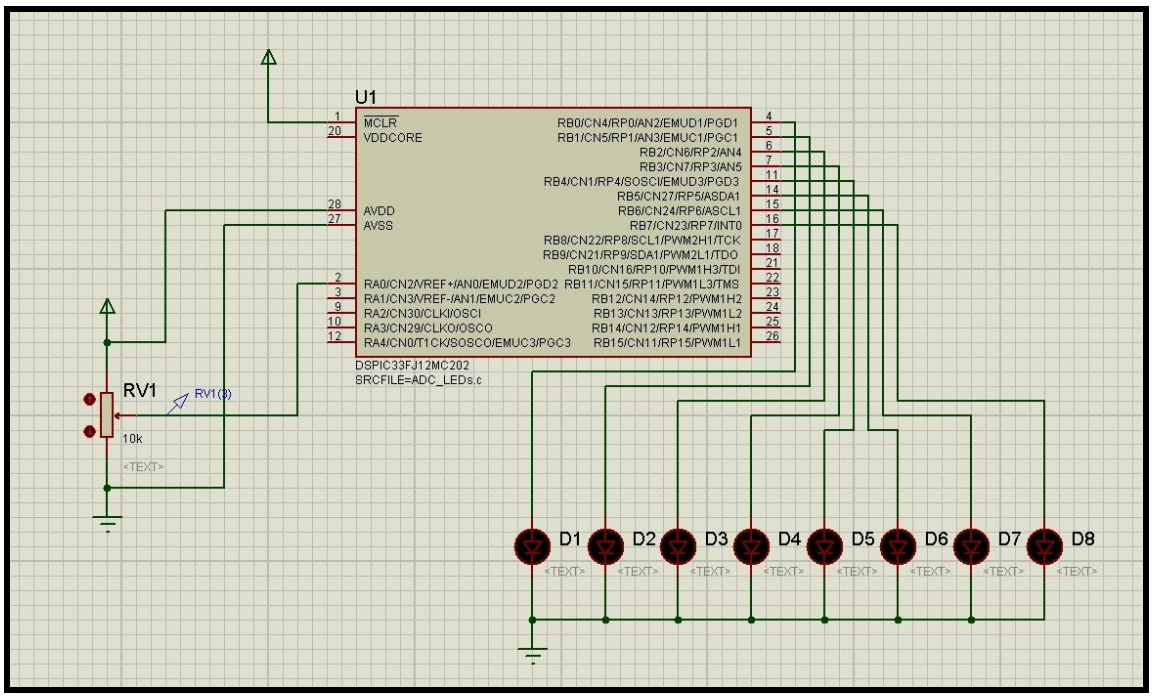


FIG. 4. ESQUEMA CONVERSOR ANALOGICO DIGITAL.

5.4.2 DESARROLLO.

Para el desarrollo del código, lo primero es establecer la configuración del reloj del sistema (mediante la función "init_clock()") y la de los puertos. Establecemos el PUERTO B para la salida de los LED´s y mediante la función "init_led(void)" encendemos y apagamos los LED's 3 veces para comprobar su correcto funcionamiento, ayudándonos de unos retardos de 500ms para poderlo apreciar visualmente. Después con la función "init_ADC(void)" establecemos la configuración del conversor A/D, es decir, establecemos el canal AN0 como canal analógico de adquisición de datos (la señal del potenciómetro analógico), para adquirir una muestra por interrupción, con una resolución de 12 bits, además, configuramos el muestreo y la conversión en modo automático con un TAD = 47 TCY (con el modo por defecto de formato de lectura).

Posteriormente creamos una función "volts_value_to_leds(void)" para transformar los datos adquiridos del potenciómetro (en la ejecución de la interrupción actualizamos la bandera de la interrupción del ADC para poder seguir muestreando y convirtiendo datos de manera infinita). Dicha función por medio de una tabla de valores predefinidos y de la siguiente fórmula de conversión, transforma el valor de los voltios a la escala de LED´s.

$$scala = (unsigned\ int)\ volts\_value * 0x0007 / 0x0C00$$

Para finalizar, dentro de la función main desactivamos el perro guardián e iniciamos todas las funciones antes descritas para el funcionamiento del programa junto con un bucle infinito para que nunca pare de ejecutarse la conversión.

Código generado:

```
#include <p33FJ12MC202.h>

#define FCY 4000000UL  // Definimos el ciclo de trabajo para poder utilizar los Delays (retardos).
#include <libpic30.h> // Librería para los Delays (retardos).

_FOSCSEL(FNOSC_PRIPLL); // Oscilador Primario (XT, HS, EC) con circuito PLL.
_FOSC(FCKSM_CSDCMD & OSCIOFNC_OFF & POSCMD_XT);
// FCKSM_CSDCMD --> Cambio de reloj y monitorización de reloj deshabilitados.
// OSCIOFNC_OFF --> La función del terminal OC2 es la de reloj.
// POSCMD_XT --> Oscilador XT.
_FWDT(FWDTEN_OFF);  // Perro Guardián deshabilitado.


volatile unsigned int volts_value = 0;
volatile unsigned int flag_volts_value = 0;


void init_clock(void)
{
```

```
    // Configuro el oscilador y el circuito PLL para 16 MIPS
    PLLFBD = 0x00A0;    //PLLDIV<8:0>=0xA0 --> M=160 en Decimal.
    CLKDIV = 0x0048;    //01001000 =0x48.
    //NOS VAMOS AL DATASHEET...
    //PLLPRE<4:0>(CLKDIV<4:0>)=01000 --> N1=ENTRADA/10.
    //PLLPOST<1:0>(CLKDIV<7:6>)=01 --> N2=SALIDA/4.

    //FRECUENCIA = (PLLDIV/PLLPOST*PLLPRE)*FOSC ---> EN ESTE CASO FOSC=4MHz.
    //Por lo consiguiente tendríamos una frecuencia:
    //Frecuencia= (160/10*4)*4 Mhz = 16 Mhz
}


void init_leds(void)
{
    unsigned int i;

    TRISB = 0xFF00; // Terminales RB0...RB7 configurados como salidas.
    LATB = 0xFF00; // Terminales RB0...RB7 puestos a '0'.


    for(i=0; i<3; i++) // Bucle que enciende y apaga 3 veces todos los LEDs para comprobar que funcionan.
    {
        LATB = 0xFFFF;  // Encendemos todos los LEDs.
        __delay_ms(500); //Retardo de 0.5 segundos.
        LATB = 0xFF00;  // Apagamos todos los LEDs.
        __delay_ms(500); //Retardo de 0.5 segundos.
    }

}


void init_ADC( void )
{
    // Establecemos la configuración del puerto analógico (PUERTO A).
    AD1PCFGLbits.PCFG0 = 0; // Asegurar que el terminal AN0 es analógico (Potenciómetro analógico).
    // Establecer aquí el canal de exploración, auto muestreo y conversor con el modo por defecto de formato de
lectura.
    AD1CON1 = 0x00E4; // Inicio y conversión automáticos.
    // Selección de 12 bits y 1 canal de operación ADC.
    AD1CON1bits.AD12B = 1;
    // No canal de escaneo para CH0+, Usamos MUX A, SMPI = 1 por interrupción, Vref = AVdd/AVss.
    AD1CON2 = 0x0000;
    // Establecer el número de muestras y el tiempo de conversión.
    AD1CON3 = 0x032F; // 1 TAD y TAD = 47 TCY
    // Establecemos aquí el canal de escaneo para AN0.
    AD1CSSLbits.CSS0 = 1;
    //Seleccionamos el canal AN0.
    AD1CHS0bits.CH0SA = 0000;
    // Restablecer la bandera de interrupción ADC.
    IFS0bits.AD1IF = 0;
    // Habilitar las interrupciones ADC.
    IEC0bits.AD1IE = 1;
    // Encender el módulo ADC.
    AD1CON1bits.ADON = 1;
}
```

```
void __attribute__((interrupt,no_auto_psv)) _ADC1Interrupt(void){

   if(flag_volts_value==0)
   {
      volts_value = ADC1BUF0; // Guardar el valor del potenciómetro conectado a AN0 en el buffer de datos.
      flag_volts_value = 1; // Poner la bandera para actualizar los LEDs.
   }
   IFS0bits.AD1IF = 0; // Restablecer bandera de interrupción ADC.
}


void volts_value_to_leds(void)
{
   unsigned int scala,leds;

   scala = (unsigned int) volts_value * 0x0007 / 0x0C00;   // Conversión del valor muestreado a un valor decimal
comprendido entre 0 y 8.
   leds = scala;   // Actualización del valor de la variable 'leds'.

   LATB = 0xFF00;  // Apagamos todos los LEDs.
   // Dependiendo del valor de la variable 'leds' encendemos mayor o menor número de LEDs.
   if(leds>0) LATB += 0b00000001;  // RB0 =1;
   if(leds>1) LATB += 0b00000010;  // RB1 =1;
   if(leds>2) LATB += 0b00000100;  // RB2 =1;
   if(leds>3) LATB += 0b00001000;  // RB3 =1;
   if(leds>4) LATB += 0b00010000;  // RB4 =1;
   if(leds>5) LATB += 0b00100000;  // RB5 =1;
   if(leds>6) LATB += 0b01000000;  // RB6 =1;
   if(leds>7) LATB += 0b10000000;  // RB7 =1;
}


int main(void)
{
   RCONbits.SWDTEN = 0; // Desactivar el temporizador del perro guardián WDT.

   init_clock();   // Función que inicializa el reloj del sistema.
   init_leds();    // Función que inicializa el PUERTO B conectado a los LEDs.
   init_ADC();     // Función que inicializa el Módulo ADC.


   // Bucle infinito
   while(1)
   {
      if(flag_volts_value == 1)
      {
         flag_volts_value = 0;
         volts_value_to_leds();   // Función que enciende los LEDs dependiendo del valor digital obtenido del
conversor A/D.
      }
   }
}
```

5.5. MÓDULO DE CAPTURA DE ENTRADA.

El dispositivo dsPIC30F6010 dispone de 8 canales de captura. Su misión es la de medir el tiempo que dura un acontecimiento externo producido por una señal que se aplica a uno de los terminales de entrada del módulo y que se mide con el temporizador TIMER2 o TIMER3.

Es muy útil en aplicaciones que hay que controlar la frecuencia o el periodo de los impulsos a analizar (Captura simple) y también como fuente de interrupción externa.

5.5.1 DESCRIPCIÓN.

Para esta práctica vamos a utilizar el módulo de captura en modo de captura simple en cada flanco de bajada de la señal introducida por el terminal IC1.

Se generara una interrupción cada vez que se produce un flanco de bajada, y se activara o desactivara un LED que está conectado al terminal de salida RB3 (alarma) si el valor acumulado en el registro de temporización del timer3 es mayor al que nosotros programemos.

Para la señal de entrada por el terminal IC1, hemos elegido un generador de señales digitales de un único flanco, en el que podemos elegir el periodo del flanco y elegir si queremos flanco descendente o ascendente, a parte de otras muchas funcionalidades de este dispositivo que para esta práctica no usaremos. El montaje del circuito en PROTEUS sería: Ver fig.5

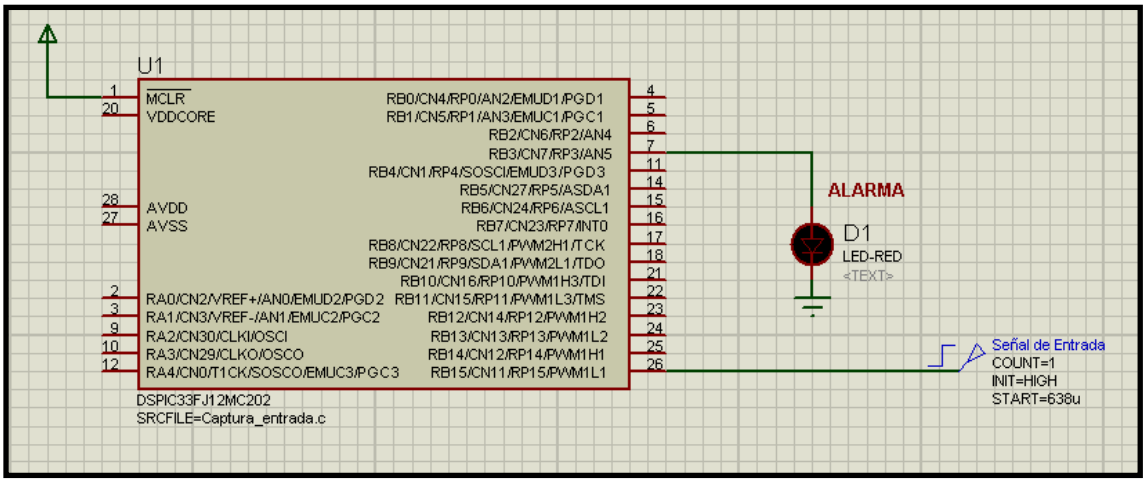


FIG. 5. ESQUEMA MODULO DE CAPTURA.

5.5.2 DESARROLLO:

Como en el resto de programas empezamos configurando los puertos: con el bit ICTMR que está dentro del registro IC1CONbits elegiremos el TIMER a usar para medir la temporización, en este caso elegimos el TIMER3. Dentro de ese mismo registro, configuramos

el módulo de captura para que este capturando siempre y que funcione en el modo de flanco de bajada (descendente).

La entrada de los flancos hay que introducirlos por el terminal ICx, el dspic33f no dispone de terminales específicos de entrada al módulo de captura como el dspic30f, pero tiene una serie de terminales remapeables (se pueden elegir las características del terminal), así que configuramos el terminal RB15 como IC1, configuramos todos los terminales de salida menos el RB15, y el LED de alarma lo podremos conectar a cualquier terminal, en este caso hemos elegido el RB3.

Ahora pasamos a configurar el TIMER3 y su temporización para que se genere una alarma (encienda el diodo LED) si la temporización es mayor que la que nosotros configuremos o no genere alarma. Hemos ajustado una frecuencia centrada en el oscilador de 8,2Mhz.

Pasamos a los cálculos para introducir la temporización al TIMER3, teniendo en cuenta que el registro de temporización TMR3 aumenta su cuenta en una unidad cada Tcy (tiempo de ciclo de instrucción).

- Queremos meter una temporización de 550us, con una frecuencia de oscilador de 8,2Mhz. Calculamos el tiempo de ciclo de instrucción (Tcy).

$$Fcy = \frac{Fcosc}{2} = \frac{8{,}2Mhz}{2} = 4{,}1\ Mhz$$

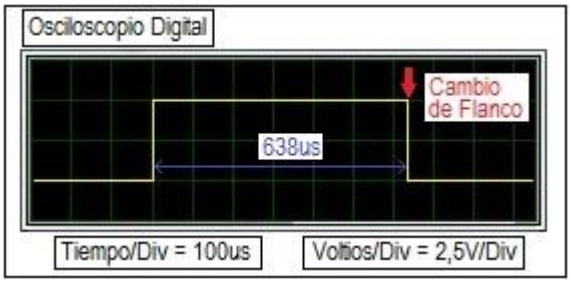
$$Tcy = \frac{1}{Fcy} = 244ns$$

- El valor a meter en el registro de temporización TMR3 es el resultado de dividir la temporización que queremos entre el valor del Tcy.

$$Temporización = TMR3 \times Tcy = 550us$$

$$TMR3 = \frac{Temporización}{Tcy} = \frac{550us}{244ns} = 2255\ d = \ 0X08CF\ h$$

Hemos comprobado que esos 550us a la hora de simular se nos convierten en 638us debido a retardos de inicialización tanto del TIMER3 como del módulo de captura, los tiempos de retardo producidos en la simulación y en la propia ejecución de las secuencias del programa. La señal resultante sería la siguiente:



Código generado:

```
#include <p33FJ12MC202.h>

/****************PROTOTIPADO DE FUNCIONES*****************
*********************************************************************************/
void ConfigPort (void);
void ConfigTMR3 (void);
void ConfigOSC (void);


void __attribute__((interrupt,auto_psv)) _IC1Interrupt(void)
{
   IFS0bits.IC1IF=0;
   if(PORTBbits.RB15==1)
   {
      TMR3 = 0x0000;         // Inicializo el registro temporizador.
   }
   if(PORTBbits.RB15==0)
   {
      T3CONbits.TON=0;      // Apagamos el Timer3.
      if(TMR3>0X08CF)        // Registro temporizador del Timer3; Temporización= TMR3*Tcy.
                             // Temporización = 2255*244 ns = 550 us. (NOTA: 0X08CF h = 2255 d).
      {
         PORTBbits.RB3=1;    //Si TMR3 > 550us, se enciende el LED (conectado al terminal RB3).
      }
      if(TMR3<0X08CF)
      {
         PORTBbits.RB3=0;    //Si TMR3 < 550us, se apaga el LED (conectado al terminal RB3).
      }
   }
}

int main (void)
{

   ConfigPort();
   ConfigOSC();
   ConfigTMR3();

   // Configuración del módulo de captura de entrada 1.
   IC1CONbits.ICM = 0b000;         // Se deshabilita el módulo de captura de entrada.
   IC1CONbits.ICTMR = 0;           // Se usara el timer 3.
```

```
   IC1CONbits.ICI = 0b00;          // Interrupción siempre capturando.
   IC1CONbits.ICM = 0b010;         // Modulo Captura de entrada cada flanco de bajada.

   // Configuración de la interrupción del IC1.
   IPC0bits.IC1IP = 1;         // Prioridad de la Interrupción del IC1.
   IFS0bits.IC1IF = 0;         // Limpio el Flag de la interrupción del IC1.
   IEC0bits.IC1IE = 1;         // Activar la interrupción del IC1.

  while(1);
}

/*////////////////CONFIGURACION DE PUERTOS//////////////////
////////////////////////////////////////////////////////////////////////////////////////*/

void ConfigPort (void)
{

   TRISB = 0X8000;// Todos los terminales como salidas menos el terminal RB15 que es de entrada.

   // Mapeo (PPS) del terminal IC1.
   __builtin_write_OSCCONL(OSCCON & 0xbf);
   RPINR7bits.IC1R = 0b01111;  // Terminal IC1 en el terminal RB15.
   __builtin_write_OSCCONL(OSCCON | 0x40);

}


/*/////////////////CONFIGURACION DEL OSCILADOR ////////////////
////////////////////////////////////////////////////////////////////////////////////////*/

void ConfigOSC (void)
{
   OSCCONbits.COSC=0b000;      // Oscilador FRC.
   OSCCONbits.NOSC=0b000;      // Cambio de oscilador a FRC.
   OSCCONbits.CLKLOCK=0;       // Cambio de fuente de reloj habilitado.
   OSCCONbits.IOLOCK=0;        // Remapeo de terminales no bloqueado (PPS).
   OSCCONbits.LOCK=0;          // Estado del PPL (bit de solo lectura).
   OSCCONbits.CF=0;            // Detección de fallo de reloj deshabilitada.
   OSCCONbits.LPOSCEN=0;       // Oscilador secundario deshabilitado.
   OSCCONbits.OSWEN=0;         // Cambio de fuente de reloj completado.
   OSCTUNbits.TUN=0b011110;    // Frecuencia centrada + 11.25% (8.20 MHz).

// Tcy = 1/(FOSC/2) = 1/ 4.10 MHz = 244 ns --> Tiempo de ciclo de instrucción
}

/*/////////////////CONFIGURACION DEL TIMER3//////////////////
////////////////////////////////////////////////////////////////////////////////////////*/

void ConfigTMR3 (void)
{
   T3CONbits.TON=0;            // Apagamos el Timer3.
   T3CONbits.TCS=0;            // Fuente interna de reloj (FOSC/2)--> 8.20 MHz/2= 4.10 MHz.
   T3CONbits.TGATE=0;          // Acumulación de tiempo en puerta deshabilitado.
   T3CONbits.TCKPS=0b00;       // Valor del preescaler igual a 0.
   T3CONbits.TON=1;            // Encendemos el Timer3.
}
```

5.6. MÓDULO DE COMPARACIÓN DE SALIDA.

Este recurso es muy eficaz en aquellas aplicaciones que precisan generar impulsos de anchura variable y los que realizan operaciones simples PWM (para la corrección del factor de potencia).

El funcionamiento simplificado de este dispositivo se basa en la comparación del valor almacenado en un registro con el temporizador (TIMER2 o TIMER3). Cuando coinciden dichos valores se genera un pulso o tren de pulsos en un terminal de salida. Es capaz de generar interrupciones. Dispone de los siguientes modos de operación:

- Modo de comparación simple.
- Doble modo de comparación de salida.
    - Pulso simple de salida.
    - Pulso continúo de salida.
- Modulación simple de anchura de pulso (PWM)
    - Con entradas de protección de fallo.
    - Sin entrada de protección de fallo.
- Comparación durante los modos Idle y Sleep de la CPU.
- Interrupción mediante la salida de comparación/PWM.

5.6.1 DESCRIPCIÓN:

Vamos a desarrollar en esta práctica el módulo de comparación de salida en Modo de Pulso Continuo para generar una onda cuadrada del 50% del ciclo de trabajo, 5 Voltios de pico a pico y con una frecuencia de 40Khz → f = 1/(PR3*TCY) = 1/(200*125ns) = 1/25µs= 40Khz. Para configurar el módulo de comparación para la generación de una corriente continua de pulsos en su salida, es necesario seguir los siguientes pasos:

- Determinar el tiempo de ciclo de instrucción TCY = 1/(Fosc/2) = 1/(16Mhz/2) = 125ns.
- Calcular el valor deseado del ancho de pulso basado en TCY (Ancho = 100 TCY).
- Calcular el tiempo de comienzo del pulso para que el temporizador comience en 0x0000.
- Asignar el ancho del pulso de inicio y fin en los registros de comparación (OCxR y OCxRS respectivamente, donde x denota el canal 1,2,…, N).
    - OC1R = valor 100 decimal = 0x64.
    - OC1RS = valor 200 decimal = 0xC8.
- Establecer en el registro de periodo del temporizador un valor igual o mayor que el valor asignado en el registro de comparación OCxRS. (PR3 = valor 200 decimal = 0xC8).
- Ajustar OCM<2:00>=101 Modo de pulso continuo.

- Activar el temporizador con el bit TON (TxCON<15>) = 1.

El circuito montado en PROTEUS es el siguiente. Fig.6.

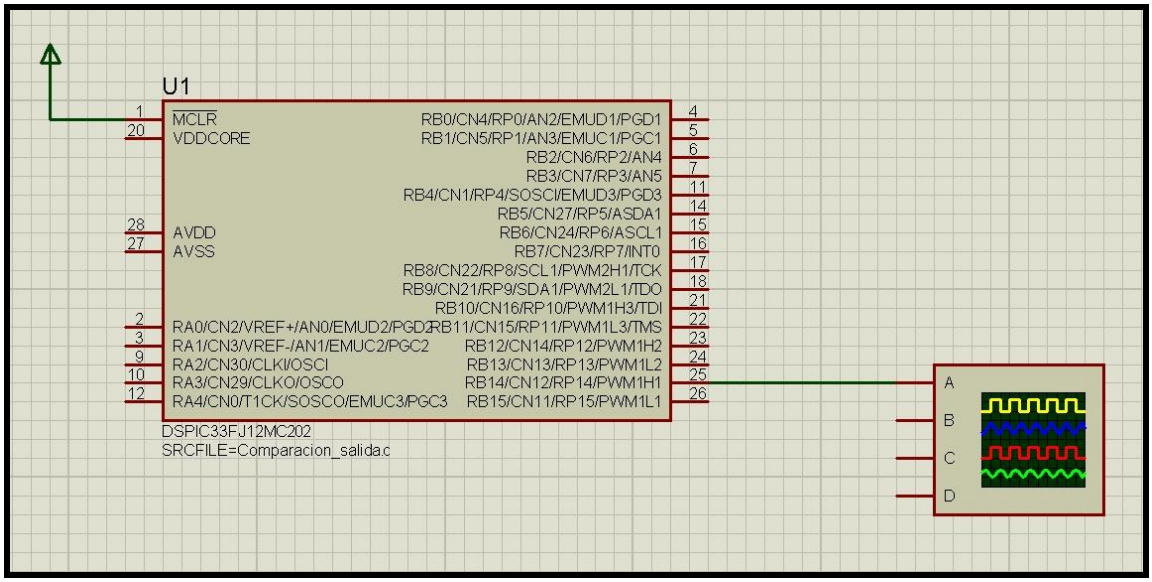


FIG. 6. ESQUEMA MODULO DE COMPARACIÓN.

5.6.2 DESARROLLO:

Dentro del programa principal (main) configuramos el oscilador y el circuito PLL, para 16 MHz. En el datasheet encontramos:

- PLLPRE<4:0>(CLKDIV<4:0>)=01000 → N1=ENTRADA/10
- PLLPOST<1:0>(CLKDIV<7:6>)=01→N2=SALIDA/4
- PLLDIV<8:0>=0X0A0→160 EN DECIMAL

Calculamos la frecuencia con esta fórmula y estos valores:

$$FRECUENCIA = \left(\frac{PLLDIV}{PLLPOST * PLLPRE}\right) * FOSC = \left(\frac{160}{10 * 4}\right) * 4Mhz = 16Mhz$$

Posteriormente mapeamos el terminal de salida OC1 al terminal RP14 y configuramos la interrupción del módulo de comparación de salida 1.

Después configuramos el módulo OC1 para la anchura del pulso requerida:

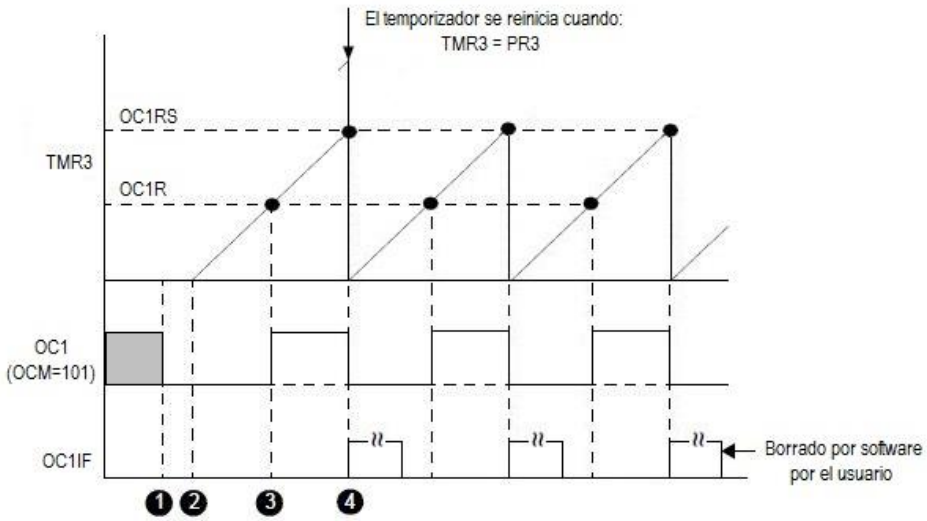
- Pulse_star = 0x64 → valor 100 decimal.
- Pulse_stop = 0xC8 → valor 200 decimal.

Inicializamos el valor del registro de periodo PR3 con el valor en decimal 200 (0xC8 h) y configuramos el valor del temporizador 3 a '0' (TMR3=0x0000). A continuación encendemos el temporizador 3 (TON=1).

Por último realizo la llamada a la función OpenOC1() para configurar el módulo de comparación como se ha descrito anteriormente y creo un bucle infinito para que se estén generando pulsos por el terminal de salida OC1 indefinidamente.

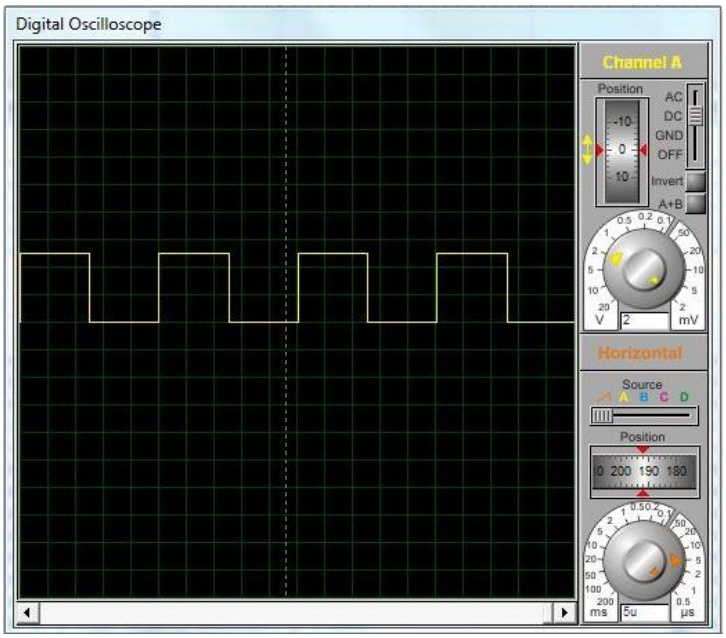
Con todo lo anterior programado, tenemos el módulo de comparación configurado de la siguiente manera:

- *Modo de Pulso Continuo:*



1. Modo de Pulso continuo habilitado, terminal OC1 conduce a nivel bajo inmediatamente.
2. El temporizador está habilitado y comienza a incrementarse.
3. Cuando OC1R = TMR3, el terminal de salida OC1 se pone a nivel alto.
4. Cuando OC1RS = TMR3, el terminal de salida OC1 se pone a nivel bajo.

Señal de salida generada:



Código generado:

```
#include <p33FJ12MC202.h>
#include <outcompare.h>

_FOSCSEL(FNOSC_PRIPLL); // Oscilador Primario con circuito PLL.
_FOSC(FCKSM_CSDCMD & OSCIOFNC_OFF & POSCMD_XT);
// FCKSM_CSDCMD --> Cambio de reloj y monitorización de reloj deshabilitados.
// POSCMD_XT --> Oscilador XT.
_FWDT(FWDTEN_OFF);  // Perro Guardián habilitado/deshabilitado por el usuario mediante software.
// Rutina de atención a la interrupción (ISR) de la interrupción OC1.
void __attribute__((interrupt,auto_psv)) _OC1Interrupt(void)
{
  IFS0bits.OC1IF = 0;
}
int main(void)
{
    // Contiene el valor en el que el terminal OC1 comienza a conducir a nivel alto.
    unsigned int pulse_start;
    // Contiene el valor en el que el terminal OC1 comienza a conducir a nivel bajo.
    unsigned int pulse_stop;
    // Configuro el oscilador y el circuito PLL para 16 Mhz.
     PLLFBD = 0x009E; //PLLDIV<8:0>=0x0A0 --> 160 EN DECIMAL.
     CLKDIV = 0x0048; //01001000 =0x48.
```

```
   // NOS VAMOS AL DATASHEET...
   // PLLPRE<4:0>(CLKDIV<4:0>)=01000 --> N1=ENTRADA/10.
   // PLLPOST<1:0>(CLKDIV<7:6>)=01 --> N2=SALIDA/4.
   // FRECUENCIA = (PLLDIV/PLLPOST*PLLPRE)*FOSC ---> EN ESTE CASO FOSC = 4MHz.
   // Por lo consiguiente tendríamos una frecuencia:
   // Frecuencia= (160/10*4)*4 Mhz = 16 Mhz.
   // Desbloqueo el remapeo de terminales.
   __builtin_write_OSCCONL(OSCCON & 0xbf);
   // Mapeo el terminal de salida OC1 al terminal RP14.
   RPOR7bits.RP14R = 18;
   // Bloqueo el remapeo de terminales.
   __builtin_write_OSCCONL(OSCCON | 0x40);
   // Cierro el módulo OC1.
   CloseOC1();
     // Configuro la interrupción del módulo de comparación de salida 1.
   ConfigIntOC1(OC_INT_OFF & OC_INT_PRIOR_5);
   // OC_INT_OFF --> Interrupción deshabilitada.
   // OC_INT_PRIOR_5 --> Interrupción con prioridad 5.
   // Configuro el módulo OC1 para la anchura de pulso requerida.
   pulse_start = 0x64; // valor 100 decimal = 0x64.
   pulse_stop = 0xC8;  // valor  200 decimal = 0xC8.
   PR3 = 0xC8 ; // Inicio el registro de periodo del temporizador 3 con el valor 300 decimal = 0xC8.
   TMR3  = 0x0000; // Inicio a '0' el registro de temporización del temporizador 3.
   T3CON = 0x8000; //Encendemos el temporizador 3 --> TON=1.
   // Configuro el módulo de comparación de salida.
   OpenOC1(OC_IDLE_CON & OC_TIMER3_SRC & OC_CONTINUE_PULSE, pulse_stop ,pulse_start);
   // OC_IDLE_CON --> Configuración del modo Idle.
   // OC_TIMER3_SRC --> Timer3 como fuente de reloj.
   // OC_CONTINUE_PULSE --> Modo de operación de pulso continuo.
   // pulse_stop --> Valor cargado en el registro OC1RS (Fin del Pulso).
   // pulse_start --> Valor cargado en el registro OC1R (Inicio del Pulso).

while(1){   // Bucle infinito.
}
   return 0;   // En caso de fallo se cierra el programa.
}
```

5.7 MÓDULO PWM. CONTROL DE MOTORES.

El Módulo PWM para el Control de Motores sirve para generar un tren de impulsos de anchura variable que permite variar la potencia que se entrega a un motor y regular su velocidad con precisión. En particular soporta las siguientes aplicaciones de potencia y control del movimiento:

- Motores de CA de inducción trifásica.
- Motores de reluctancia conmutada (SR).
- Motores de CC sin escobillas (BLDC).
- Sistemas de alimentación ininterrumpida (UPS).

5.7.1 DESCRIPCIÓN:

Vamos a configurar el módulo PWM para generar una onda cuadrada del 50% del ciclo de trabajo, 1,22 Khz de frecuencia, 5 voltios de pico a pico (5 Vpp) y 13 bits de resolución.

La salida complementaria PWM1H y PWM1L va a ir conectada a un motor de corriente continua y a su vez conectaremos el terminal PWM1H a un osciloscopio digital para comprobar la forma de onda generada. El circuito en PROTEUS es el siguiente. Ver Fig.7.

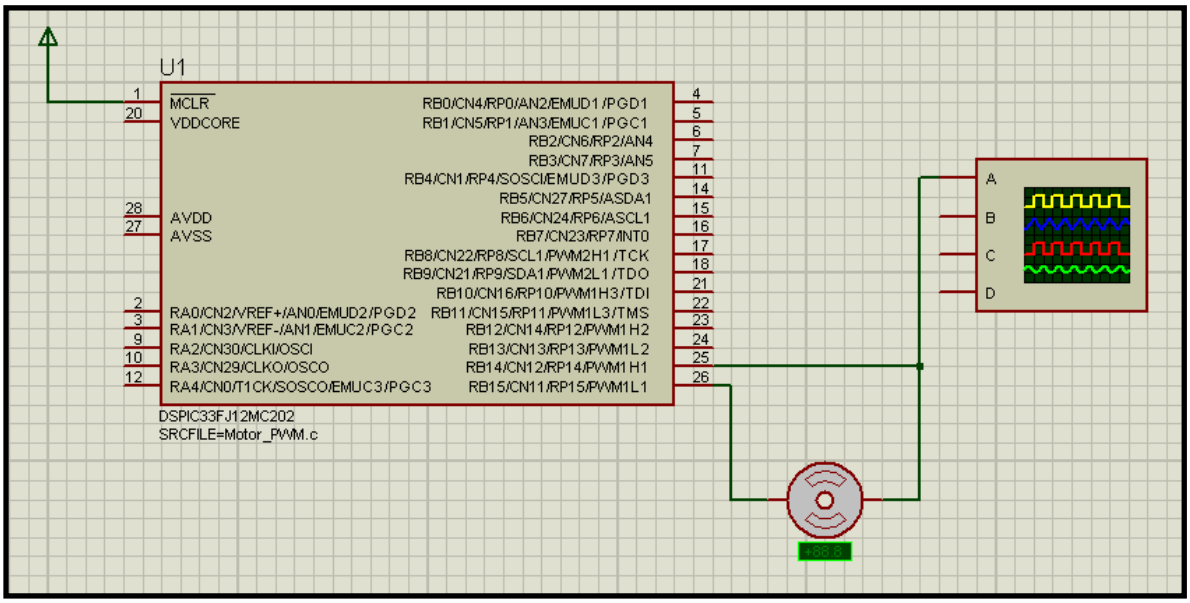


FIG. 7. ESQUEMA MODULO PWM DE CONTROL DE MOTORES.

5.7.2 DESARROLLO:

En esta práctica incluimos la librería <pwm12.h> en la que están todas las funciones configuradas del módulo PWM tanto para el dspic30fxxx como para el dspic33fxxx.

Incluimos oscilador primario sin PLL, conmutación de reloj y monitorización de fallo de reloj deshabilitados y deshabilitamos el perro guardián (WDT).

Lo primero es declarar y configurar las variables que vamos a utilizar dentro de las funciones del módulo PWM (<pwm12.h>):

- *config*: Contiene la configuración de la interrupción PWM.
- *config1*: Contiene la configuración del módulo PWM.
- *config2*: Contiene la configuración a cargar en el registro PWMCON1.

- *config3*: Contiene la configuración del postscaler del disparador de eventos especiales y del ciclo de trabajo.
- *period:* Contiene el valor del tiempo base del periodo de la señal PWM (Registro PTPER).
- *sptime:* Contiene el valor a cargar en el registro especial de comparación de eventos.
- *dutycyclereg:* Puntero al Registro de ciclo de trabajo.
- *dutycycle:* Contiene el valor del ciclo de trabajo PWM (Registro PDC1).
- *updatedisable:* Contiene el valor del bit de actualización del registro PWMCON2.

Posteriormente realizamos la llamada a las funciones por el siguiente orden de ejecución:

- *ConfigIntMCPWM1(config):* Esta función configura las interrupciones y el nivel de prioridad para el módulo PWM.
- *SetDCMCPWM1(dutycyclereg,dutycycle,updatedisable):* Esta función configura los registros del ciclo de trabajo y el bit de actualización "PWM Update" del registro PWMCON2.
- *OpenMCPWM1(period,sptime,config1,config2,config3):* Esta función configura los registros PTPER, SEVTCMP, PTCON, PWMCON1 y PWMCON2.

Finalmente realizamos un bucle infinito para que el programa se esté ejecutando indefinidamente con las configuraciones anteriormente mencionadas.

Con todo lo anterior programado, tenemos el módulo PWM configurado de la siguiente manera:

- S*alida complementaria* en los terminales PWM1H y PWM1L (usa un generador de impulsos de anchura variable con hasta 16 bits de resolución).
- *PDC1=0x7FE* → Registro para el ciclo de trabajo que determina la cantidad de tiempo que la salida del módulo PWM está en estado activo (valor de 2046 d).
- *PTPER=0x7FE* → Registro para el tiempo base del periodo de la señal PWM (valor de 2046 d).
- *PTMR sin prescaler* → Registro para la generación de la base de tiempo PWM (contador de tiempo).
- *Periodo de la señal PWM:*

$$PxTPER = \frac{F_{CY}}{F_{PWM} \times PxTMR\ Prescaler \times 2} - 1$$

Donde:

$$Fcy = \frac{Fosc}{2} = \frac{10Mhz}{2} = 5Mhz \rightarrow Tcy = 200ns$$

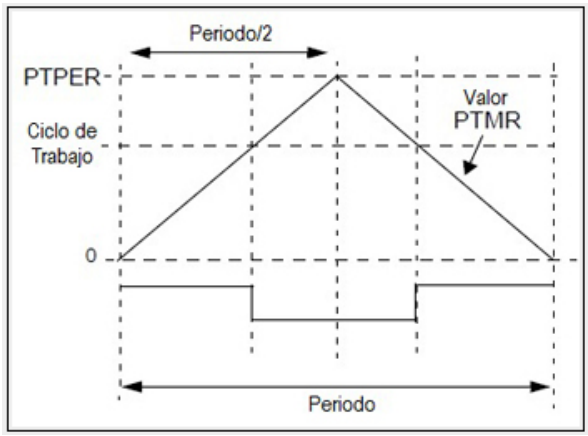
$$PTMR\ preescaler = 1:1$$

$$PTPER = 2046$$

Luego:

$$Fpwm = 1{,}22\ Khz \rightarrow Tpwm = 820\ \mu s$$

- *Modo de operación continuo ascendente/descendente (PTMOD<1:0>=10):* En este modo la base de tiempo PWM cuenta hacia arriba hasta que el temporizador *PTMR* alcanza el valor de *PTPER*. El temporizador *PTMR* comenzará a contar en sentido descendente en el siguiente flanco de reloj y cuando llega a '0' se produce una interrupción y pasa a contar de nuevo en modo ascendente.
- *PWM con centro alineado:*



Resolución:

$$PWM\ Resolution = log_2\left(\frac{FCY}{FPWM}\right)$$

Donde:

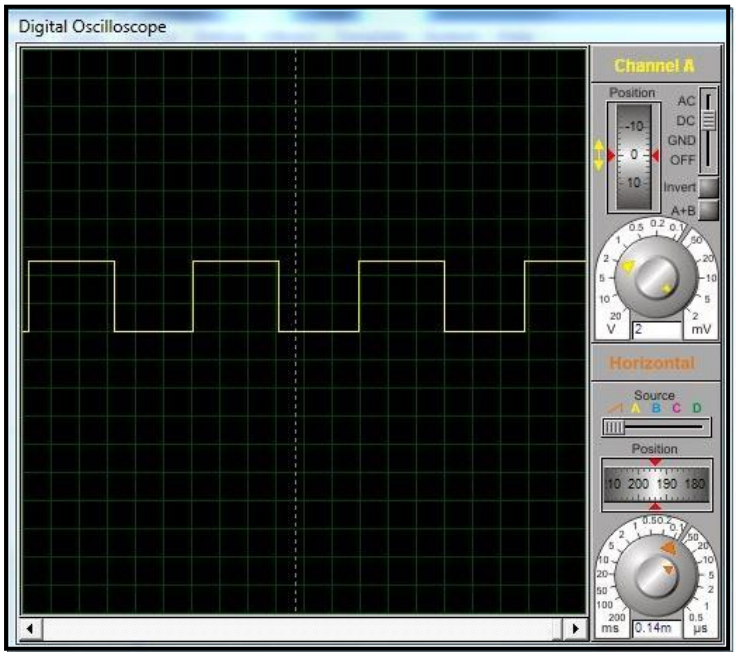
$$Fcy = \frac{Fosc}{2} = \frac{10Mhz}{2} = 5Mhz \rightarrow Tcy = 200ns$$

$$Fpwm = 1{,}22\ Khz$$

Luego:

$$PWMResolution = 12\ bits$$

Señal de salida generada:



Código generado:

```
#include <p33FJ12MC202.h>
#include <pwm12.h>

_FOSCSEL(FNOSC_PRI); // Oscilador Primario (XT, HS, EC) sin PLL.
_FOSC(FCKSM_CSDCMD & POSCMD_XT);
// FCKSM_CSDCMD --> Conmutación de reloj y Monitorización de fallo de reloj deshabilitados.
// POSCMD_XT --> Oscilador XT (10MHz).
_FWDT(FWDTEN_OFF); // Watchdog deshabilitado.
```

```
int main()
{
   unsigned int config; // Contiene la configuración de la interrupción PWM.
   unsigned int period; // Contiene el valor del periodo de la señal PWM (Registro PTPER).
   unsigned int sptime; // Contiene el valor a cargar en el registro especial de comparación de eventos.
   unsigned int config1; // Contiene la configuración del módulo PWM.
   unsigned int config2; // Contiene la configuración a cargar en el registro PWMCON1.
   unsigned int config3; // Contiene la configuración del postscaler del disparador de eventos especiales y del
ciclo de trabajo.

   // El valor de 'dutycyclereg' determina el ciclo de trabajo que se escribe en el registro PDC1.
   unsigned int dutycyclereg; // Puntero al Registro de ciclo de trabajo.
   unsigned int dutycycle; // Contiene el valor del ciclo de trabajo PWM (Registro PDC1).
   unsigned char updatedisable; // Contiene el valor del bit de actualización del registro PWMCON2.

   // Configura las interrupciones y su nivel de prioridad para el módulo PWM.
   config = (PWM1_INT_EN & PWM1_FLTA_DIS_INT & PWM1_INT_PR1 & PWM1_FLTA_INT_PR0 &
PWM1_FLTB_DIS_INT & PWM1_FLTB_INT_PR0);

   ConfigIntMCPWM1(config); // Esta función configura las interrupciones y el nivel de prioridad para el módulo
PWM.
   // PWM1_INT_EN --> Interrupciones PWM habilitadas.
   // PWM1_FLTA_DIS_INT --> Interrupción por fallo en el terminal A deshabilitada.
   // PWM1_INT_PR1 --> Prioridad '0' de interrupción PWM.
   // PWM1_FLTA_INT_PR0 --> Prioridad '0' de interrupción por fallo en el terminal A.
   // PWM1_FLTB_DIS_INT --> Interrupción por fallo en el terminal B deshabilitada.
   // PWM1_FLTB_INT_PR0 --> Prioridad '0' de interrupción por fallo en el terminal B.

   //Configuración del módulo PWM para generar una onda cuadrada del 50% del ciclo de trabajo.
   dutycyclereg = 1;   // Puntero al Registro de ciclo de trabajo.

   dutycycle = 0x7FE; // Valor del ciclo de trabajo PWM (Registro PDC1)--> 2046 d.

   updatedisable = 1; // Actualización del ciclo de trabajo habilitada.

   SetDCMCPWM1(dutycyclereg,dutycycle,updatedisable); //Esta función configura los registros del ciclo de
trabajo y el bit de actualización "PWM Update" del registro PWMCON2.

   period = 0x7FE; // Valor del tiempo base del periodo de la señal PWM (Registro PTPER) --> 2046 d.

   sptime = 0x0; // Valor de los bits de la comparación de eventos especiales (Registro SEVTCMP).

   config1 = (PWM1_EN & PWM1_IDLE_STOP & PWM1_OP_SCALE1 & PWM1_IPCLK_SCALE1 &
PWM1_MOD_UPDN); //Registro PTCON
   // PWM1_EN --> Habilitación del Módulo PWM.
   // PWM1_IDLE_STOP --> Modo Idle deshabilitado.
   // PWM1_OP_SCALE1 --> Postscaler de salida igual a 1:1.
   // PWM1_IPCLK_SCALE1--> Preescaler de entrada igual a 1:1.
   // PWM1_MOD_UPDN --> Modo de operación PWM: Modo continuo Ascendente/Descendente.

   config2 = (PWM1_MOD1_COMP & PWM1_PDIS3H & PWM1_PDIS2H & PWM1_PEN1H & PWM1_PDIS3L &
PWM1_PDIS2L & PWM1_PEN1L);// Registro PWMCON1.
   // PWM1_MOD1_COMP --> Primer canal en modo de salida complementario.
   // PWM1_PDIS3H y PWM1_PDIS2H --> Terminales que trabajan como E/S de propósito general.
   // PWM1_PDIS3L y PWM1_PDIS2L --> Terminales que trabajan como E/S de propósito general.
   // PWM1_PEN1H --> Terminal que trabaja como salida a nivel alto del módulo PWM.
   // PWM1_PEN1L --> Terminal que trabaja como salida a nivel bajo del módulo PWM.
```

```
    config3 = (PWM1_SEVOPS1 & PWM1_OSYNC_PWM & PWM1_UEN); // Registro PWMCON2.
    // PWM1_SEVOPS1 --> Postscaler para el disparador de eventos especiales.
    // PWM1_OSYNC_PWM --> Las salidas del registro P1OVDCON se sincronizan con la base de tiempo PWM.
    // PWM1_UEN --> Actualización del PWM habilitada.

    // Esta función configura los registros PTPER, SEVTCMP, PTCON, PWMCON1 y PWMCON2.
    OpenMCPWM1(period,sptime,config1,config2,config3);

    while(1); // Bucle infinito.
}
```

5.8 MÓDULO DE COMUNICACIÓN. UART.

El UART es un elemento destinado a soportar la comunicación serie asíncrona y funciona de forma bidireccional, adaptándose al trabajo de muchos periféricos. La comunicación se realiza con dos líneas, una para la transmisión UTX y otra para la recepción URX, entrando y saliendo los bits a una frecuencia controlada internamente. El dspic30F6010 tiene 2 módulos UART.

Las características principales del módulo UART son:

- Funciona en modo “Full-duplex” con datos de 8 o 9 bits.
- Tiene 1 o 2 bits de STOP (parada).
- Incluye un bit de paridad par o impar (para datos de 8 bits).
- Dispone de un generador de baudios con un predivisor de frecuencia de 16 bits (encargado de generar la frecuencia de trabajo del módulo).
- Puede generar una interrupción tanto el bloque transmisor como el receptor.
- Las velocidades de transmisión varían desde 38 bps hasta 2.5Mbps a una velocidad de instrucción de 30 MHz.
- Buffers de transmisión y recepción de 9 bits de ancho y con una profundidad de 4 caracteres.
- Modo de bucle cerrado para dar soporte de diagnóstico.
- Detección de ERROR por paridad, trama y saturación del buffer.
- Soporte de interrupción únicamente en la dirección de detección (9ºbit=1).

5.8.1 DESCRIPCIÓN:

En esta práctica utilizamos el módulo UART para leer y escribir por el canal serie, incorporamos al *dsPIC33FJ12MC202* un terminal virtual que nos mostrara los siguientes mensajes:

- “Transmisión UART vía dsPIC33FJ12MC202”.
- "Pulsa un carácter para provocar la interrupción UART: ".

Al pulsar cualquier carácter del teclado, el dsPIC devolverá ese carácter (con el que hemos provocado la interrupción) al terminal virtual con el mensaje:

- "INTERRUPCION >> CARÁCTER x", donde x es el carácter pulsado.

El funcionamiento más detallado es el que sigue: Primero se presenta a sí mismo, como todo dsPIC de buena familia debe hacer, y después se pone a la escucha del canal serie (por el UART1) y devuelve por ese mismo canal el carácter recibido (hace el eco).

El montaje en PROTEUS es el siguiente. Ver Fig.8.

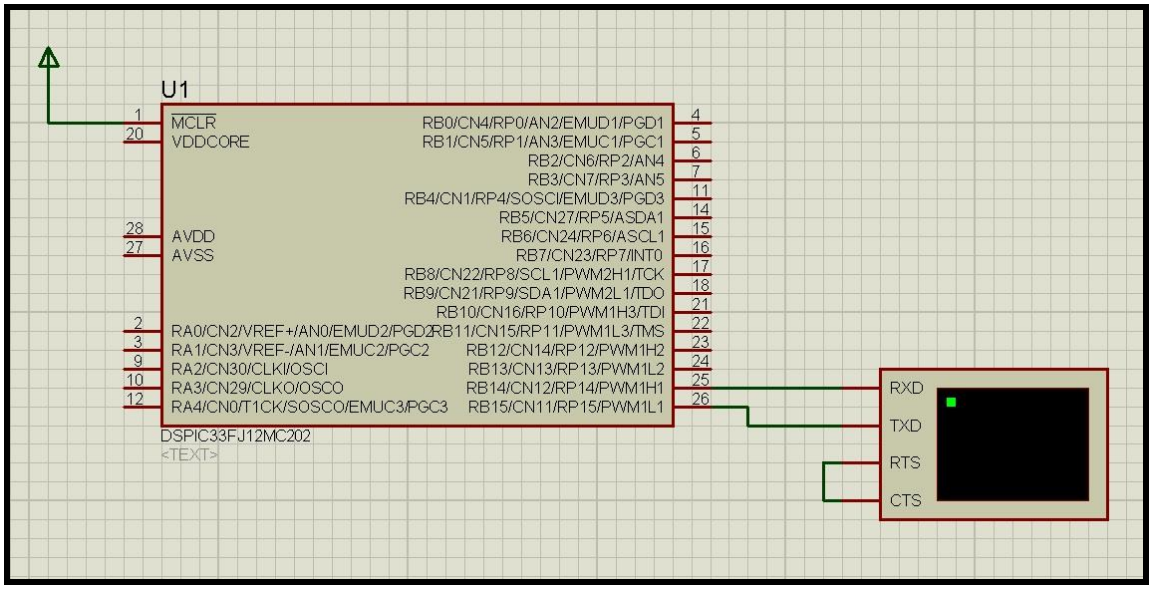


FIG. 8. ESQUEMA MODULO DE COMUNICACIÓN UART.

5.8.2 DESARROLLO:

Se añade al fichero la librería "uart.h" en la que se incluyen unas estructuras de funciones predefinidas en memoria para la configuración y manejo del módulo UART, además de la manera de programar dichas funciones.

Declaramos el oscilador FRC interno con PLL, deshabilitamos el Oscilador primario y el cambio de reloj, y configuramos la velocidad de la transmisión serie (baudios).

Declaramos tres funciones incluidas en la librería "uart.h", para la escritura y configuración del módulo UART:

- *void putUART1(char c)* → Para escribir un único carácter en el registro U1TXREG.
- *void putsUART1(char *s)* → Para escribir una cadena de caracteres en el registro U1TXREG.
- *void InitUART1()* → Para configurar el módulo UART de la siguiente manera: 1 bit de parada (stop), sin paridad, 8 bits de datos, interrupción después de transmitir un carácter, además de la propia configuración de los terminales de entrada y salida así como la configuración de las interrupciones.

Dentro del programa principal (main), configuramos el PLL y el "tuning" del oscilador FRC, y como en otras prácticas, tenemos que hacer el remapeo de los terminales: Mapeo el terminal de entrada U1RX al terminal RP15 y el terminal de salida U1TX al terminal RP14.

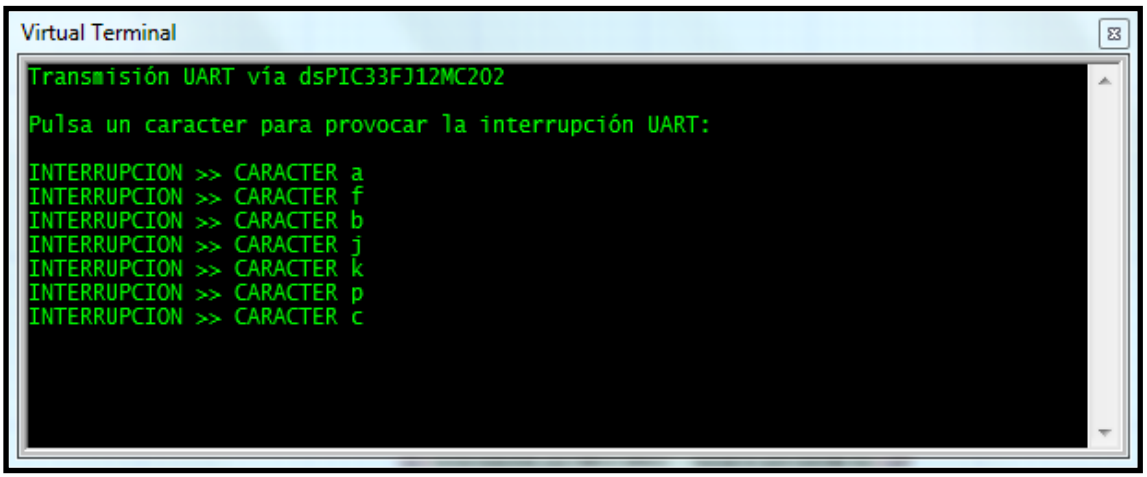
A continuación se envían al terminal virtual los mensajes de presentación del programa (cabecera) y esperamos dentro de un bucle infinito a que se produzca la interrupción por la recepción de un carácter, y es dentro de la interrupción donde se transmite el carácter recibido mediante un ECO (el módulo UART transmite el mismo carácter que recibe).

Código generado:

```
#include <p33FJ12MC202.h>
#include <uart.h>

_FOSCSEL(1);      // Oscilador FRC interno con PLL.
_FOSC (195);      // Oscilador primario deshabilitado, Cambio reloj deshabilitado.

#define FCY              40000000   // Frecuencia del ciclo de instrucción .
#define BAUDRATE         9600       // Baudios  .
#define BRGVAL           ((FCY/BAUDRATE)/16)-1

unsigned char CHAR_RECUART; // Variable para guarda los caracteres recibidos por el terminal U1TX.

/*************************Funciones para la transmisión de Caracteres*****************************/

void putUART1(char c){
        while ( U1STAbits.UTXBF); // Espera mientras el buffer de transmisión está lleno.
        U1TXREG = c;
}

 void putsUART1(char *s){
        unsigned int i=0;
        while(s[i]) // Se repite hasta el final de la cadena (*s == '\0').
        putUART1(s[i++]); // Envía un carácter y se posiciona para enviar el siguiente.
}

/*********************Interrupción Por Recepción (Rx)*****************************/
void __attribute__ ((interrupt, no_auto_psv)) _U1RXInterrupt(void){
        CHAR_RECUART = U1RXREG;
        char frase[] = {'I','N','T','E','R','R','U','P','C','I','O','N',' ','>','>',' ','C','A','R','A','C','T','E','R',' ','\0'};
        char salto[] = {'\n','\r','\0'};
        putsUART1(frase); // Escribimos la frase.
        putUART1(CHAR_RECUART); // Devuelvo como eco el valor pulsado.
        putsUART1(salto); //Salto de línea.

        IFS0bits.U1RXIF = 0; // Borramos el flag de interrupción.
        if(U1STAbits.OERR) U1STAbits.OERR=0; // Para que nunca haya desbordamiento.

}

/***************Inicializar UART*****************************/
void InitUART1(){
```

```
        U1MODEbits.STSEL = 0;           // 1 bit de parada .
        U1MODEbits.PDSEL = 0;           // Sin paridad, 8 bits de datos .
        U1MODEbits.ABAUD = 0;           // Auto-Baud Deshabilitado .
        U1MODEbits.BRGH = 0;            // Modo de baja velocidad .
        U1BRG = BRGVAL;                 // Configuración del BAUD Rate .
        U1STAbits.UTXISEL0 = 0;         // Interrupción después de que un carácter es transmitido.
        U1STAbits.UTXISEL1 = 0;
        IEC0bits.U1TXIE = 0;            // Interrupción en la transmisión deshabilitada.
        IFS0bits.U1RXIF = 0;            // Flag de Interrupción en la recepción borrado.
        IEC0bits.U1RXIE = 1;            // Interrupción en la recepción habilitada.
        U1MODEbits.UARTEN = 1;          // Módulo UART habilitado.
        U1STAbits.UTXEN = 1;            // Terminal Tx habilitado.
        U1STAbits.URXISEL1 = 0;         // Terminal Rx habilitado.
        IPC2bits.U1RXIP=7;              // Interrupt Rx Mayor prioridad.
}

/*****************************Main*******************************/
int main (void){
        CLKDIVbits.PLLPRE = 4;          // Divisor PLL_1 por 6.
        CLKDIVbits.PLLPOST = 0;         // Divisor PLL_2 por 2.
        PLLFBD = 128;                   // Multiplicador PLL por 130.
        OSCTUN = 0;                     // Frecuencia del FRC 7.37 MHz.

        // Desbloqueo el remapeo de terminales.
        __builtin_write_OSCCONL(OSCCON & 0xbf);
        // Mapeo el terminal de salida U1TX al terminal RP14.
        RPOR7bits.RP14R = 3;
        // Mapeo el terminal de entrada U1RX al terminal RP15.
        RPINR18bits.U1RXR = 15;
        // Bloqueo el mapeo de terminales.
        __builtin_write_OSCCONL(OSCCON | 0x40);

        InitUART1();
        unsigned char mytext1[] = "Transmisión UART vía dsPIC33FJ12MC202";
        unsigned char mytext2[] = "Pulsa un caracter para provocar la interrupción UART: ";
        // Cabecera que se muestra con el encendido del terminal virtual.
        putsUART1(mytext1);
        putsUART1("\r\n");
        putsUART1("\r\n");
        putsUART1(mytext2);
        putsUART1("\r\n");
        putsUART1("\r\n");
        while(1){
        // Esperando la interrupción de Rx…
        }
}
```

Recepción generada en el Terminal Virtual:



5.9 MÓDULO DE COMUNICACIÓN. SPI.

Consiste en una interfaz serie síncrono que es muy empleado para establecer la comunicación con microcontroladores, memorias EEPROM, conversores AD, etc. Es compatible con el SPI de Motorola y las interfaces SIOP. El dsPIC30F6010 dispone de 2 módulos SPI.

5.9.1 DESCRIPCIÓN:

En este ejemplo vamos a realizar una comunicación serie mediante el módulo SPI (interfaz de periféricos serie) del dsPIC33FJ12MC202, y el depurador SPI de PROTEUS.

El programa envía por el terminal SDO1 del módulo SPI los números del 0 al 255 (00 al FE) repetidamente y el depurador SPI les recibe (mediante una transmisión de 8 bits). En esta comunicación el dispositivo maestro es el propio dsPIC y el dispositivo esclavo es el depurador SPI de PROTEUS.

El circuito en PROTEUS es el siguiente. Ver Fig.9.

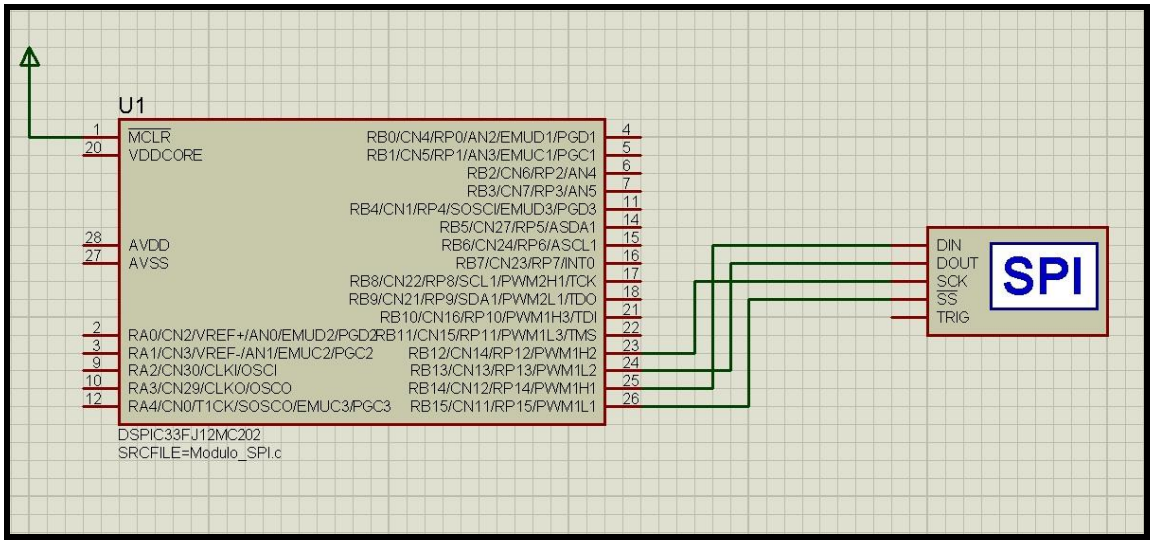


FIG. 9. ESQUEMA MODULO DE COMUNICACIÓN SPI.

5.9.2 DESARROLLO:

Añadimos al proyecto el oscilador primario FRC, conmutación de reloj habilitada, monitorización de fallo de reloj deshabilitada, modo del oscilador primario con Cristal XT y perro guardián habilitado/deshabilitado por el usuario mediante software.

Creamos tres funciones auxiliares para el control del módulo SPI:

- *void write_SPI(short command)* → Función que escribe en el registro SPI1BUF del módulo SPI.
- *void delay(void)* → Función retardo de 12,75us.
- *void Init_SPI(void)* → Función que configura el módulo SPI.

Dentro del programa principal (main), configuramos el PLL del oscilador y el PUERTO B a nuestras necesidades, además de hacer un remapeo de terminales como en prácticas anteriores:

  - Mapeo el terminal de salida SDO1 al terminal RP14.
  - Mapeo el terminal de salida SCK1OUT al terminal RP12.
  - Mapeo el terminal de entrada SS1# al terminal RP15.

Posteriormente realizamos una llamada a la función *Init_SPI()* para configurar el módulo SPI y dentro de un bucle infinito creamos la estructura para la transmisión de los datos. Dicha estructura contiene un bucle de repetición (for) para hacer llamadas a la función *write_SPI(i)* para que escriba en el registro SPI1BUF del módulo SPI los números del 0 hasta el 255, de la siguiente manera:

- Activa la línea de selección esclavo (terminal SS1#) a nivel bajo.
- Lee el registro SPI1BUF para borrar la bandera SPIRBF.

- Escribe el dato fuera del módulo SPI (A través del terminal SDO1).
- Espera a que el dato sea enviado.
- Desactiva la línea de selección esclavo (terminal SS1#) a nivel alto.

Después de cada llamada a la función *write_SPI(i),* realizamos una llamada a la función *void delay(),* para para poder escribir un dato en el registro SPI1BUF y continuar con la transmisión de los siguientes datos. A continuación se muestra el cálculo del retardo que produce la llamada a dicha función:

- El valor 255 se lo programamos manualmente dentro del bucle "for", para que cuente hasta ese valor.

$$retardo = 255 * Tcy$$

- Calculamos Tcy, donde Fosc=40MHz→M=40, N1=2, N2=2 y Fin=4MHz.

$$Fosc = Fin * M/(N1 * N2)$$

$$Tcy = \frac{1}{\frac{Fosc}{2}} = 50ns$$

- El retado es de 12,75us, un valor suficiente para permitir al módulo SPI poder escribir un dato en el registro SPI1BUF sin tener problemas con la transmisión del siguiente dato.

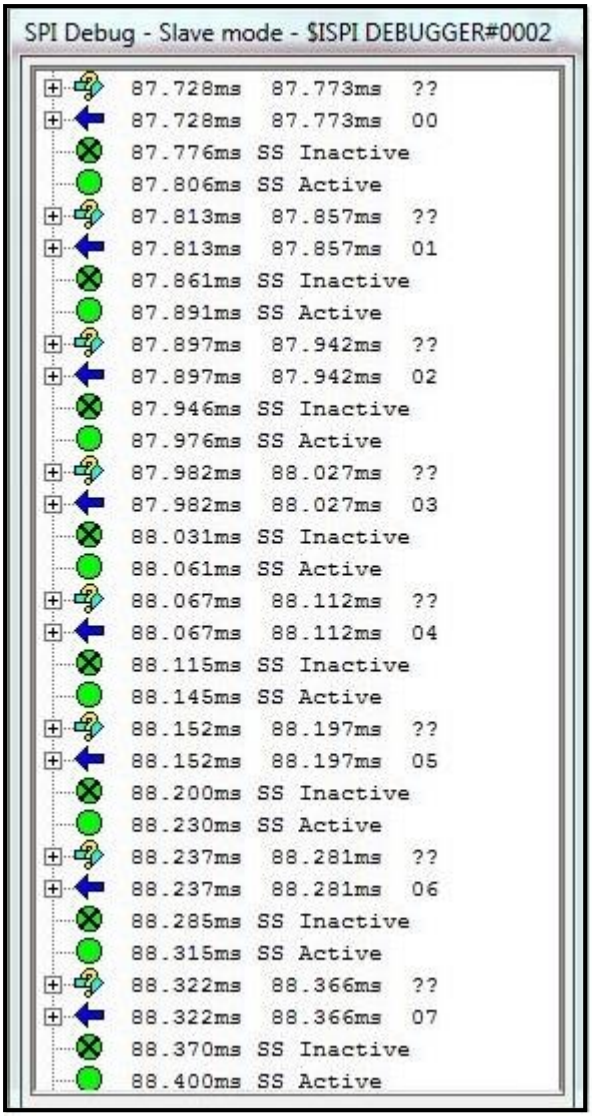
Código generado:

```
#include <p33FJ12MC202.h>

_FOSCSEL(FNOSC_FRC);    // Oscilador Primario FRC.
_FOSC(FCKSM_CSECMD & POSCMD_XT);
// FCKSM_CSECMD --> Conmutación de reloj habilitada, y monitorización de fallo de reloj deshabilitada.
// POSCMD_XT -->  Modo del oscilador primario: Cristal XT.
_FWDT(FWDTEN_OFF);  // Perro Guardián habilitado/deshabilitado por el usuario mediante software.

void write_SPI(short command) // Función que escribe en el registro SPI1BUF del módulo SPI.
{
  short temp;
  PORTBbits.RB15 = 0;      // Línea de selección esclavo (terminal SS1#) a nivel bajo (activada).
  temp = SPI1BUF;         // Lectura del registro SPI1BUF para borrar la bandera SPIRBF.
  SPI1BUF = command;      // Escribe el dato fuera del módulo SPI (A través del terminal SDO1).
   while (!SPI1STATbits.SPIRBF);   // Espera a que el dato sea enviado.
  PORTBbits.RB15 = 1;      // Línea de selección esclavo (terminal SS1#) a nivel alto (desactivada).
}
void delay(void) // Retardo equivalente a 255*Tcy = 12,75 us; donde Tcy = 1/(Fosc/2) = 1/ 20MHz = 50 ns.
```

```
{
  short temp;
  for (temp=0;temp<255;temp++);
}
void Init_SPI(void) // Función que configura el módulo SPI.
{
  // Configuramos el módulo SPI
  SPI1STAT = 0x0;  // Deshabilitamos el módulo SPI.
  SPI1CON1 = 0x0161;
  // DISSCK = 0 --> Reloj interno SPI habilitado.
  // DISSDO = 0 --> Terminal SDO1 controlado por el módulo SPI.
  // MODE16 = 0 --> Comunicación de 8 bits.
  // SMP = 0 --> Datos de entrada muestreados en la mitad del tiempo del dato de salida.
  // CKE = 1 --> La salida de datos serie cambia en la transición del estado activo de reloj al estado inactivo de
reloj.
  // SSEN = 0 --> Terminal SS1# controlado por la función del puerto.
  // CKP = 1 -->  Estado de inactividad del reloj a nivel alto; Estado activo a nivel bajo.
  // MSTEN = 1 --> Modo Maestro.
  // SPRE = 0b000 --> Preescaler Secundario 8:1 (en modo maestro).
  // PPRE = 0b01 --> Preescaler Primario 16:1 (en modo maestro).
  SPI1STAT = 0x8000; // Habilitamos el módulo SPI.
}
int main(void)
{
  short i,j;
   // Configuramos el oscilador para operar a 40MHz.
   // Fosc= Fin*M/(N1*N2), Fcy=Fosc/2.
   // Fosc= 4M*40/(2*2)=40Mhz para una frecuencia de entrada de 4MHz.
  PLLFBD = 38;              // M=40
  CLKDIVbits.PLLPOST = 0;     // N1=2
  CLKDIVbits.PLLPRE = 0;      // N2=2
  // Deshabilitamos el perro guardián.
  RCONbits.SWDTEN=0;
  // Cambio de fuente de reloj para incorporar el circuito PLL.
  __builtin_write_OSCCONH(0x03);      // Iniciamos la fuente de reloj primaria.
                                // Oscilador con PLL (NOSC=0b011).
  __builtin_write_OSCCONL(0x01);      // Comienzo del cambio de reloj.
  while (OSCCONbits.COSC != 0b011);   // Esperamos a que se produzca el cambio de reloj.

  // Esperamos al circuito PLL para bloquear el registro OSCCON.
  while(OSCCONbits.LOCK!=1) {};

  // Desbloqueo el remapeo de terminales
  __builtin_write_OSCCONL(OSCCON & 0xbf);
  // Mapeo el terminal de salida SDO1 al terminal RP14.
  RPOR7bits.RP14R = 7;
  // Mapeo el terminal de salida SCK1OUT al terminal RP12.
  RPOR6bits.RP12R = 8;
  // Mapeo el terminal de entrada SDI1 al terminal RP13.
  RPINR20bits.SDI1R = 13;
  // Mapeo el terminal de entrada SS1# al terminal RP15.
  RPINR21bits.SS1R = 15;
  // Bloqueo el remapeo de terminales.
  __builtin_write_OSCCONL(OSCCON | 0x40);

  PORTB = 0xFFFF;  // Ponemos el valor de todos los terminales del PUERTO B a '1'.
  TRISB = 0x5FFF;  // Terminales SDI1 y SS1# como terminales de entrada, los demás como salidas.
```

```
  Init_SPI();    // Llamada a la función que configura el módulo SPI.

  while(1) // Bucle infinito
  {
    for (i=0;i<255;i++)
      {
        write_SPI(i);   // Llamada a la función que escribe en el registro SPI1BUF del módulo SPI.
        delay();    // Llamada a la función retardo.
      }
  }
}
```

Recepción generada en el Depurador SPI:



NOTA: Las interrogaciones (??) se deben a que el depurador SPI de PROTEUS únicamente está recibiendo datos (no está enviando datos al dsPIC).

5.10. MODULO DE COMUNICACIÓN. I^2C

El módulo I^2C (Inter-Integrated-Circuit) tiene el hardware necesario para trabajar en modo esclavo y en modo maestro con una interfaz de 16 bits. Todos los elementos que se conectan al bus I^2C lo hacen a través de dos líneas (terminales), SDA (Datos) y SCL (Reloj). Permite la transferencia de datos bidireccional entre el modo maestro y esclavo.

En este protocolo las direcciones de los elementos de la red pueden especificarse con direcciones de 7 o 10 bits. Iniciada la transferencia de información se envía la dirección del elemento y al comprobar que le corresponde genera una señal de reconocimiento ACK que permite el progreso de la transferencia.

5.10.1. DESCRIPCION:

Vamos a emplear el módulo de comunicación I^2C en modo maestro para grabar datos en una memoria EEPROM.

En el modo transmisor Maestro, los datos serie son emitidos a través del terminal SDA1, mientras que el terminal SCL1 emite el reloj serie. El primer byte recibido es el byte de comienzo de la transmisión (en este caso `A0´) y nos sirve para comprobar que se ha iniciado la transmisión. El segundo byte recibido es el byte de dirección ID de selección de dispositivo (en este caso `11´). Este contiene la dirección de esclavo del dispositivo de recepción (7 bits) y el bit de dirección de datos. En este caso el bit de dirección de datos (R_W) es de lógica negativa.

Los datos serie son transmitidos en grupos de 8 bits (1 byte). Después de la transmisión de cada byte, un bit ACK es recibido. Las condiciones de Inicio y Parada se emiten para indicar el principio y el fin de la transferencia serie.

La transmisión de un byte de datos o la transmisión de la dirección de 7 bits, se logra simplemente escribiendo un valor en el registro de transmisión I2CTRN (en este caso escribe la secuencia de datos: 00 01 02 03 04 05 06 07 08 09 en la memoria EEPROM 24LC256).

El usuario solo debe escribir en el registro de transmisión I2CTRN cuando el módulo se encuentra en un estado de inactividad (Idle). Esta acción activa el indicador de búfer completo (TBF), y permite que el Generador de Baudios empiece a contar de forma decreciente (en este caso desde 300 hasta 0) y que comience la siguiente transmisión.

Cada bit de dirección/datos se desplaza hacia afuera a través del terminal SDA1 después de que se haya producido el flanco de bajada de reloj (terminal SCL1). El resultado de la transmisión lo podemos visualizar a través del depurador I2C que PROTEUS nos ofrece. El montaje del circuito en PROTEUS sería el siguiente. Fig.10.

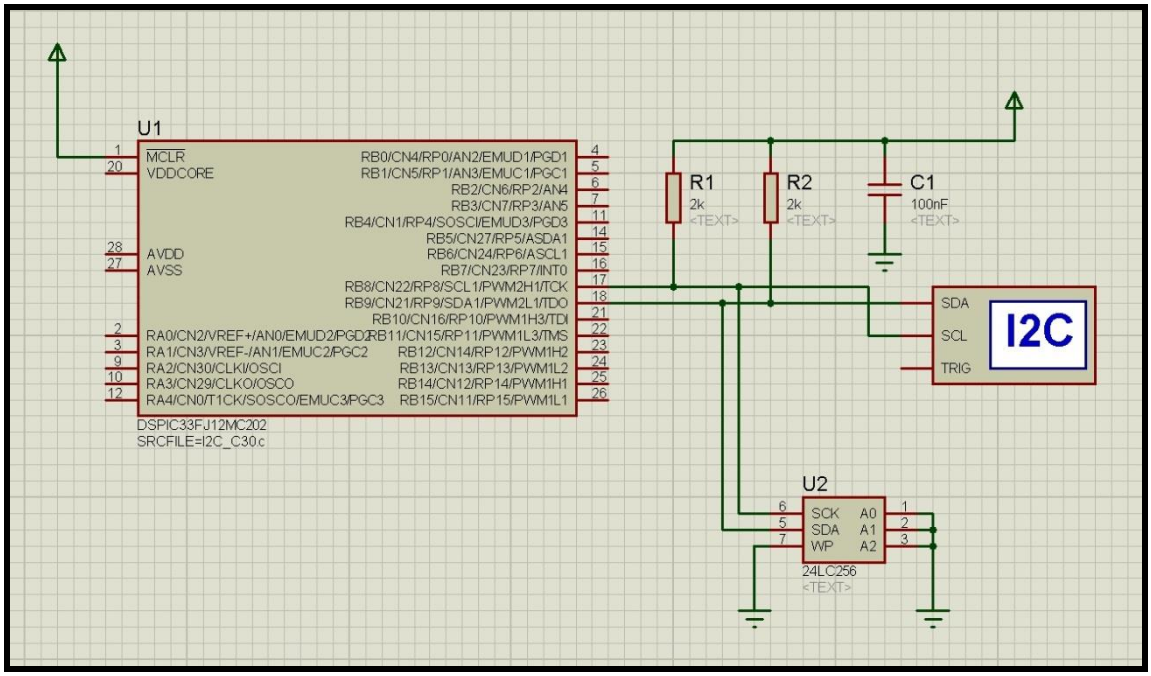


FIG. 10. ESQUEMA MODULO DE COMUNICACIÓN I^2C.

5.10.2. DESARROLLO

Incluimos un fichero de cabecera llamado “i2cEmem.h” en el que están incluidas las estructuras: I2CEMEM_DRV y I2CEMEM_DATA.

Como es necesario en cualquier práctica, configuramos el dispositivo con los “FUSES” de configuración de la siguiente manera:

- Oscilador primario: Oscilador Rápido RC.
- Cambio de fuente de reloj habilitado a Oscilador XT.
- Perro Guardián habilitado/deshabilitado por el usuario mediante software.
- Temporizadores de encendido apagados.
- Protección de código deshabilitada.

Posteriormente declaramos dos variables del tipo de las dos estructuras anteriores, para trabajar con ellas en nuestro programa:

- *i2cmem* → Variable tipo estructura I2CEMEM_DRV (variable para el accionamiento).
- *wData* → Variable tipo estructura I2CEMEM_DATA (variable para los datos).

Declaramos las rutinas de atención a la interrupción (ISR) del módulo I^2C Maestro y Esclavo dentro de las cuales borramos sus respectivas banderas de interrupción para que se puedan seguir produciendo interrupciones cuando sea necesario.

Además declaramos y programamos las dos funciones siguientes:

- *void I2CEMEMinit(I2CEMEM_DRV *i2cMem)* → Para inicializar el Módulo I^2C con las especificaciones descritas en el código.
- *void I2CEMEMdrv(I2CEMEM_DRV *i2cMem)* → Para realizar la comunicación serie entre el Módulo I2C y la memoria EEPROM 24LC256.

Dentro de la función *I2CEMEMdrv(I2CEMEM_DRV *i2cMem)* creamos una máquina de estados para el proceso de comunicación serie del módulo I^2C con la EEPROM mediante una estructura *"switch(case)"* y la variable de control *"state"* para saltar entre los diferentes estados como se muestra a continuación:

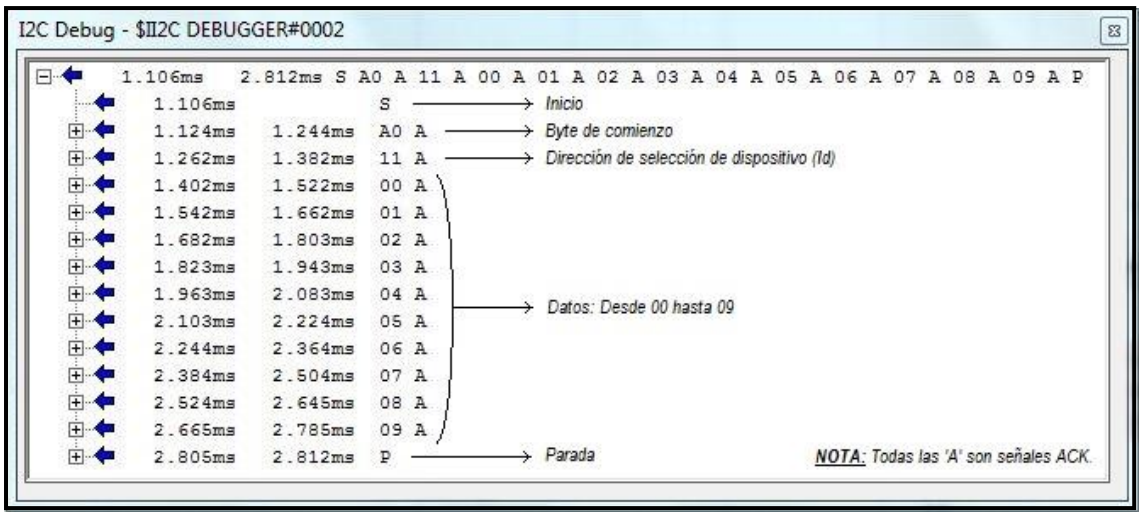
- Inicialización de comando:
  - Case 0: Activamos el comando de escritura I^2C.
- Control y dirección de fase:
  - Case 1: Condición de inicio (Start) en los terminales SDA1 y SCL1.
  - Case 2: Byte de comienzo con selector Id del dispositivo.
  - Case 3: Enviar dirección Id de 1 byte, si se recibe Ack. Sino reintentar
  - Case 4: Escritura de la EEPROM, Si Ack no recibido ir a bandera de error y salir
- Escritura de la fase de datos:
  - Case 5: Enviamos datos y aumentamos el contador.
  - Case 6: Buscar el final de los datos o la falta de la señal Ack, Si Ack no recibido ir a bandera de error y salir. Si el contador llega al número de datos (7), parar secuencia.
- Parar secuencia:
  - Case7: Condición de parada en los terminales SDA1 y SCL1.
  - Case 8: Reiniciar contador e iniciar etapas.
- Bandera de error y salir:
  - Case 9: Condición de parada en los terminales SDA1 y SCL1.
  - Case 10: Reiniciar contador e iniciar etapas.
- Reintentar:
  - Case 11: Condición de parada en los terminales SDA1 y SCL1.
  - Case 12: Reiniciar contador e iniciar etapas.

Por último, dentro del programa principal, seguimos la siguiente secuencia de comandos:

- Configuramos el oscilador para que el dispositivo opere a 40MHz.
- Deshabilitamos el perro guardián.
- Realizamos el cambio de reloj para incorporar el circuito PLL.
- Inicializamos el módulo I^2C y el accionamiento (Driver).

- Inicializamos el buffer de datos para poder escribirlos en la memoria EEPROM.
- Inicializamos la estructura de datos I^2C para operación de escritura.
- Habilitamos la escritura de datos serie desde el módulo I^2C hacia la memoria EEPROM.
- Dentro del bucle infinito "while" escribimos los datos en la EEPROM y mientras el dispositivo no se encuentre en el modo de inactividad (el dispositivo sale del modo de inactividad una vez concluido el envío de un dato) enviamos la señal ACK para que se pueda continuar con la transmisión de los datos.

Secuencia de datos generada:



Código generado:

```
#include "p33FJ12MC202.h"
#include "i2cEmem.h"

_FOSCSEL(FNOSC_FRC);       // Oscilador Primario: Oscilador Rápido RC.
_FOSC(FCKSM_CSECMD & POSCMD_XT);
// FCKSM_CSECMD --> Cambio de reloj habilitado.
// POSCMD_XT --> Oscilador XT.
_FWDT(FWDTEN_OFF);      // Perro Guardián habilitado/deshabilitado por el usuario mediante software.
_FPOR(FPWRT_PWR1);      // Temporizadores de encendido apagados.
_FGS(GCP_OFF);      // Protección de código deshabilitada.

// Instancias de estructuras de accionamiento y Datos.
I2CEMEM_DRV i2cmem = I2CSEMEM_DRV_DEFAULTS;   // Accionamiento --> Variable i2cmem de tipo estructura I2CEMEM_DRV.
// I2CSEMEM_DRV_DEFAULTS { 0,\(I2CEMEM_DATA *)0,\(void (*)(void *))I2CEMEMinit,\(void (*)(void *))I2CEMEMdrv}
I2CEMEM_DATA wData; // Datos --> Variable wData de tipo estructura I2CEMEM_DATA.
// Las dos estructuras de datos anteriormente mencionadas están incluidas dentro del fichero de cabecera "i2cEmem.h"
```

```
// Definición de variables globales.
unsigned int wBuff[10];
unsigned int enable;
unsigned int jDone;


/*=============================================================================
                    Rutina de atención a la interrupción del Módulo I2C Maestro
=============================================================================*/
void __attribute__((interrupt, no_auto_psv)) _MI2C1Interrupt(void)
{
      jDone=1;
      IFS1bits.MI2C1IF = 0;      //  Bandera de la Interrupción del módulo I2C Maestro borrada.

}


/*=============================================================================
                  Rutina de atención a la interrupción del Módulo I2C Esclavo
=============================================================================*/
void __attribute__((interrupt, no_auto_psv)) _SI2C1Interrupt(void)
{

      IFS1bits.SI2C1IF = 0;   //  Bandera de la Interrupción del módulo I2C Esclavo borrada.
}


/*=============================================================================
                                  Inicialización del módulo I2C
=============================================================================*/
void I2CEMEMinit(I2CEMEM_DRV *i2cMem)
{
  i2cMem->cmd = 0;     // Inicializo la variable cmd (comando).
  i2cMem->oData = 0;   // Inicializo la variable oData (datos de salida).

  I2C1CONbits.A10M = 0; // El registro I2C1ADD contiene una dirección esclavo de 7 bits.
  I2C1CONbits.SCLREL = 1; // Actualizar (Release) el reloj SCL1.
  I2C1BRG = 300; // Registro del generador de Baudios -->  BRG cuenta hacia atrás y se detiene cuando llega a
0, hasta que se produzca una nueva recarga.

  I2C1ADD = 0; // Registro de dirección.
  I2C1MSK = 0; // Registro de la máscara de la dirección.

  I2C1CONbits.I2CEN = 1;    // Habilitación del módulo I2C y configuración de los terminales SDA1 y SCL1 como
terminales de puerto serie.
  IEC1bits.MI2C1IE = 1;     // Interrupción del módulo I2C Maestro habilitada.
  IFS1bits.MI2C1IF = 0;     // Bandera de la Interrupción del módulo I2C Maestro borrada.
}


/*=============================================================================
     Comunicación serie Módulo I2C --> EEPROM. Máquina de estados base del accionamiento.
=============================================================================*/
void I2CEMEMdrv(I2CEMEM_DRV *i2cMem)
{

static int state = 0, cntr = 0, rtrycntr = 0;
```

```
 switch(state)
 {
/*=================================*/
/*       Inicialización de comando          */
/*=================================*/
 case 0:
    if( (i2cMem->cmd == I2C_WRITE))     // Comando de escritura I2C.
       state=1;  // Valor de state para pasar al siguiente case (case 1).
    break;

/*=================================*/
/*       Fase de control y dirección          */
/*=================================*/
 case 1:

   I2C1CONbits.SEN = 1;  // Condición de inicio (Start) en los terminales SDA1 y SCL1.
   state = state+1;      // Valor de state para pasar al siguiente case (case 2).
   break;


 case 2:    // Byte de comienzo con selector Id del dispositivo.

    if(jDone == 1)
    {
       jDone = 0;
       state = state+1;    // Valor de state para pasar al siguiente case (case 3).
       I2C1TRN  =  (0xA0)|(((i2cMem->oData->csel)&0x7)<<1); // Byte de comienzo igual a 'A0'(Cargado en el
Registro de transmisión).
    }
    break;

 case 3:    // Enviar dirección Id de 1 byte, si se recibe Ack. Si no reintentar.

    if(jDone == 1)
    {
       jDone = 0;

       if(I2C1STATbits.ACKSTAT == 1)    // Ack no recibido, Reintentar.
       {

          if(rtrycntr < MAX_RETRY)    // MAX_RETRY = 1000.
             state = 11;    // Valor de state para pasar al case 11(Reintentar).
          else
             state = 9;    // Valor de state para pasar al case 9(Bandera de error y salir).

       }
       else
       {

       rtrycntr = 0;     // Ponemos el contador de intentos de envío a '0'.

       I2C1TRN =((i2cMem->oData->addr));  // Enviamos dirección de selección de dispositivo (Id).
       state = state+1;  // Valor de state para pasar al siguiente case (case 4).
       }
    }
    break;
```

```
case 4:
   // Escritura de la EEPROM.
   if(jDone == 1)
   {
      jDone = 0;

      if(I2C1STATbits.ACKSTAT == 1)  // Si Ack no recibido...
      {
         state = 9;      // Valor de state para pasar al case 9(Bandera de error y salir).
      }
      else
      {
        if(i2cMem->cmd == I2C_WRITE) // Comando de escritura I2C.
            state = state+1;   // Valor de state para pasar al siguiente case (case 5).
      }
   }
   break;

/*==================================*/
/*       Fase de escritura de datos          */
/*==================================*/

case 5:
   I2C1TRN = *(i2cMem->oData->buff + cntr); // Enviar datos.
   state = state+1;  // Valor de state para pasar al siguiente case (case 6).
   cntr = cntr+1;    // Aumento del contador "cntr"
   break;

case 6:
   // Buscar el final de los datos o la falta de la señal Ack.
   if(jDone == 1)
   {
      jDone = 0;
      state = state-1;    // Valor de state para pasar al anterior case (case 5).

      if(I2C1STATbits.ACKSTAT == 1)  // Si Ack no recibido...
      {
         state = 9;      // Valor de state para pasar al case 9(Bandera de error y salir).
      }
      else
      {
         if(cntr == i2cMem->oData->n) // Si el contador "cntr" llega al número de datos a transmitir...
            state = 7;     // Valor de state para pasar al case 7 (Parar secuencia).
      }
   }
   break;

/*==================================*/
/*              Parar secuencia                */
/*==================================*/
case 7:
   I2C1CONbits.PEN = 1;  // Condición de parada en los terminales SDA1 y SCL1.
   state++;
   break;

case 8:
   if(jDone == 1)
   {
```

```
            jDone = 0;
            state = 0;
            cntr = 0;   // Reinicio contador.
            i2cMem->cmd = 0;
         }
         break;

      /*=================================*/
      /*         Bandera de error y salir              */
      /*=================================*/
       case 9:
         I2C1CONbits.PEN = 1;   // Condición de parada en los terminales SDA1 y SCL1.
         state++;
         break;

       case 10:
         if(jDone == 1)
         {
            jDone = 0;
            state = 0;
            rtrycntr = 0;
            cntr = 0;
            i2cMem->cmd = 0xFF;
         }
         break;

      /*=================================*/
      /*                  Reintentar                      */
      /*=================================*/
       case 11:
         I2C1CONbits.PEN = 1;   // Condición de parada en los terminales SDA1 y SCL1.
         state++;
         rtrycntr++;
         break;

       case 12:
         if(jDone == 1)
         {
            jDone = 0;
            state = 0;
            cntr = 0;
         }
         break;
      }
}

int main(void)
{
   int i=0;

   // Configuración del oscilador para que el dispositivo opere a 40MHz.
   // Fosc= Fin*M/(N1*N2), Fcy=Fosc/2.
   // Fosc= 4M*40/(2*2)=40Mhz para una frecuencia de entrada de 4MHz.
   PLLFBD = 38;              // M=40.
   CLKDIVbits.PLLPOST = 0;      // N1=2.
   CLKDIVbits.PLLPRE = 0;      // N2=2.
   OSCTUN = 0x0000;           // Frecuencia centrada 7.37 MHz nominales), cuando usamos el oscilador FRC.
```

```
  // Deshabilito el perro guardián.
  RCONbits.SWDTEN = 0;


  // Cambio de reloj para incorporar el circuito PLL.
  __builtin_write_OSCCONH(0x03);      // Inicio el cambio de reloj con el oscilador primario.
                                      // Oscilador con PLL (NOSC=0b011).
  __builtin_write_OSCCONL(0x01);      // Inicio del cambio de reloj.
  while (OSCCONbits.COSC != 0b011);   // Espero a que se produzca el cambio de reloj.

  // Espero al circuito PLL para bloquear el registro OSCCON.
  while(OSCCONbits.LOCK!=1) {};


  // Inicializo el módulo I2C y el accionamiento (Driver).
  i2cmem.init(&i2cmem);

  // Inicializo el buffer de datos para poder escribirlos en la memoria EEPROM .
  for(i=0;i<10;i++)
    wBuff[i] = i;

  // Inicializo la estructura de datos I2C para operación de escritura.
  wData.buff = wBuff; // Buffer de datos.
  wData.n = 10;   // Nº de datos a transmitir.
  wData.addr = 0x11;  // Dirección de selección de dispositivo(Id).
  wData.csel = 0x00;  // Bits de selección de dispositivo (terminales A0, A1 Y A2 de la EEPROM a masa, es
decir, a '0').


  // Habilitación de la escritura de datos serie desde el módulo I2C hacia la memoria EEPROM.
  enable = 1;

  while(1)
  {

    if(enable == 1)
    {
      enable = 0;

      // Escritura de datos
      i2cmem.oData = &wData;    // La salida de datos es cargada con el buffer de datos.
      i2cmem.cmd = I2C_WRITE;   // Comando de escritura I2C.

      while(i2cmem.cmd!=I2C_IDLE )   // Mientras el dispositivo no se encuentre en el modo de inactividad...
      {
        i2cmem.tick(&i2cmem);   // Envío de la señal ACK.
      }
    }
  }
}
```

5.11. INTERRUPCIONES Y EXCEPCIONES

Las interrupciones y las excepciones son causas que desvían el flujo de control en la ejecución de instrucciones. Son provocadas por acontecimientos externos, como los que originan los periféricos integrados o las señales aplicadas en determinados terminales. Dependiendo del modelo dsPIC atienden a una cantidad diferente de interrupciones y excepciones, cuya prioridad a la hora de actuar se puede programar.

5.11.1. DESCRIPCIÓN:

Para explicar este mecanismo que tiene el dsPIC para desviar el flujo de control en la ejecución de instrucciones, mediante las interrupciones, vamos a hacer un ejemplo muy sencillo y didáctico. Vamos a provocar una interrupción externa por medio de un interruptor (la señal externa que provoca la interrupción puede venir de cualquier dispositivo físico o una señal realimentada del propio microcontrolador) conectado al terminal RB0/INT dispuesto para este tipo de interrupciones, que controle el apagado/encendido de un LED conectado al terminal RB0. El montaje del circuito en PROTEUS sería el siguiente. Fig.x.

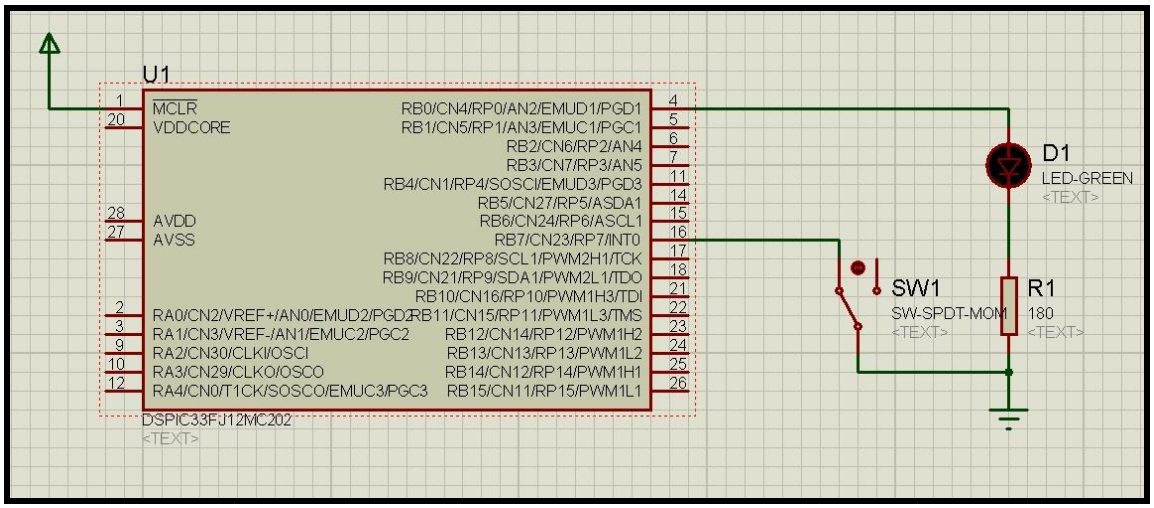


FIG. x. ESQUEMA DE LA INTERRUPCIÓN EXTERNA.

5.11.2 DESARROLLO:

El código de este programa es muy sencillo, como todos los demás. En este caso es suficiente con activar la señal que entra en el terminal RB7/INT0 mediante un conmutador de estado, para que la rutina de atención a la interrupción (ISR) envíe una llamada a la función de interrupción → "void EXT0_isr(void)" , la cual realiza el cambio de estado en el terminal RB0 (conectado al LED), cada vez que se produzca una interrupción externa, para apagar y encender un LED. Tendremos también que habilitar la interrupción externa por el terminal INT0, declarar la interrupción externa por flanco de subida (de '0' a '1' lógico) y las interrupciones

generales y además configurar el terminal RB7/INT0 como entrada para que se habilite la interrupción externa a través de él (damos por hecho que el terminal RB0 tiene que estar configurado como salida).

Al accionar el interruptor, este cambia de estado generándose una señal (flanco de subida) que entra por el terminal RB7/INT0 activando la interrupción externa mediante la cual se cambia el estado del terminal RB0 encendiendo un LED.

Código generado:

```
#include<33FJ12MC202.h>

#fuses XT   // Oscilador de cristal para frecuencias entre 4Mhz y 10Mhz.
#fuses NOWDT   // Sin Perro Guardián (WDT).

#use delay( clock = 4000000 )   // Reloj de 4 MHz.
#use fast_io(B)  // Ocasiona que el compilador realice E/S sin programar el registro de dirección.

#INT_EXT0  // Rutina de atención a la interrupción por cambio en RB0 (ISR)
void EXT0_isr(void)    // Función de la interrupción externa.
{
  output_toggle(pin_B0);  // Se conmuta la salida RB0.
}
void main()  // Función principal.
{
  set_tris_B(0x80);  // RB7 como entrada, RB0 como salida.
  output_low(PIN_B0);   // Apaga LED (RB0 a nivel bajo)
  set_pullup(PIN_B7);   // Pull-up para el terminal RB7
  enable_interrupts(INT_EXT0);  // Habilita la interrupción externa
  ext_int_edge(L_TO_H);  // Interrupción externa por flanco de subida
  enable_interrupts (INTR_GLOBAL);  // Habilita la interrupción general.

while(1); // Bucle infinito.
}
```

5.12. PROGRAMAS UTILIZADOS.

En la creación de aplicaciones hemos utilizado muchos programas, unos con acierto y otros sólo nos han llevado a problemas de incompatibilidades y quebraderos de cabeza. La premisa del estudio es que las aplicaciones fueran realizadas para PROTEUS, a partir de ahí, nosotros hemos tenido que averiguar qué programas eran más compatibles con PROTEUS y con nuestro dsPIC para poder realizar la programación de las prácticas y su compilación para que funcionara correctamente la simulación bajo PROTEUS.

Para compilar los programas escritos en lenguaje C, hemos utilizado las herramientas de compilación C COMPILER de CCS y C30 de MPLAB. Puesto que son compiladores, y solo los hemos utilizado de puente para realizar la compilación necesaria para que funcione el

código escrito en lenguaje c en el simulador de PROTEUS, no los vamos a desarrollar. Se instalan los compiladores y luego al trabajar con el entorno VSM se encarga de elegir un compilador de entre una lista de ellos instalados, unos vienen por defecto ya instalados en el propio entorno VSM.

5.12.1. PROTEUS 7 Y ENTORNO VSM.

Proteus es un software de diseño electrónico desarrollado por Labcenter Electronics que consta de dos módulos: Ares e Isis y que incluye un tercer módulo opcional denominado Electra.

ISIS: Mediante este programa podemos diseñar el circuito que deseemos con componentes muy variados, desde una simple resistencia hasta algún que otro microprocesador o microcontrolador, incluyendo fuentes de alimentación, generadores de señales y muchas otras prestaciones. Los diseños realizados en Isis pueden ser simulados en tiempo real. Una de estas prestaciones es VSM, una extensión de la aplicación con la cual podremos simular, en tiempo real, todas las características de varias familias de microcontroladores, introduciendo nosotros mismos el programa que queramos que lleven a cabo.

El software de diseño y simulación Proteus VSM es una herramienta útil para estudiantes y profesionales que desean acelerar y mejorar sus habilidades para el desarrollo de aplicaciones analógicas y digitales.

La simulación puede incluir instrumentos de medición y la inclusión de gráficas que representan las señales obtenidas en la simulación.

También tiene la capacidad de pasar el diseño a un programa integrado llamado ARES (la otra aplicación de PROTEUS) en el cual se puede llevar acabo el desarrollo de placas de circuitos impresos.

VSM es un completo simulador para esquemas electrónicos que contienen microprocesador. El corazón de VSM es ProSPICE, un producto que combina un núcleo de simulación analógica usando el estándar SPICE3f5, con modelos animados de los componentes electrónicos y los microprocesadores que comprenden el circuito, tanto si el programa se ha escrito en ensamblador como si se ha utilizado un lenguaje de alto nivel, permitiendo interactuar con nuestro diseño, utilizando elementos gráficos animados realizando operaciones de indicadores de entrada y salida.

En PROTEUS8 viene VSM viene incorporado en el programa como un conjunto, nosotros hemos utilizado la versión 7 de PROTEUS por lo que hemos instalado a parte la herramienta VSM que nos ayuda a juntar la programación con el diseño realizado con ISIS.

La simulación se realiza en tiempo casi real, los efectos se pueden considerar prácticamente como en tiempo real. Incorpora prácticos controles de depuración paso a paso y

visualización del estado de las variables. La característica más sorprendente e importante de VSM es su capacidad de simular el software que se ejecuta en el microcontrolador y su interacción con cualquier componente electrónico digital o analógico conectado a él.

6. MANUAL DE USUARIO WEB.

Este manual de Usuario, tiene como finalidad dar a conocer de una manera detallada y sencilla la estructura de la Web de nuestro estudio del microcontrolador dsPIC, para que cualquier usuario pueda sacar el máximo partido de la misma, el sitio fue diseñado para que el usuario pueda, de una forma intuitiva, clara y sencilla, sin mayor capacitación, realizar búsquedas eficientes, y encontrar rápidamente dentro de ella la información que necesite.

6.1. REQUERIMIENTOS.

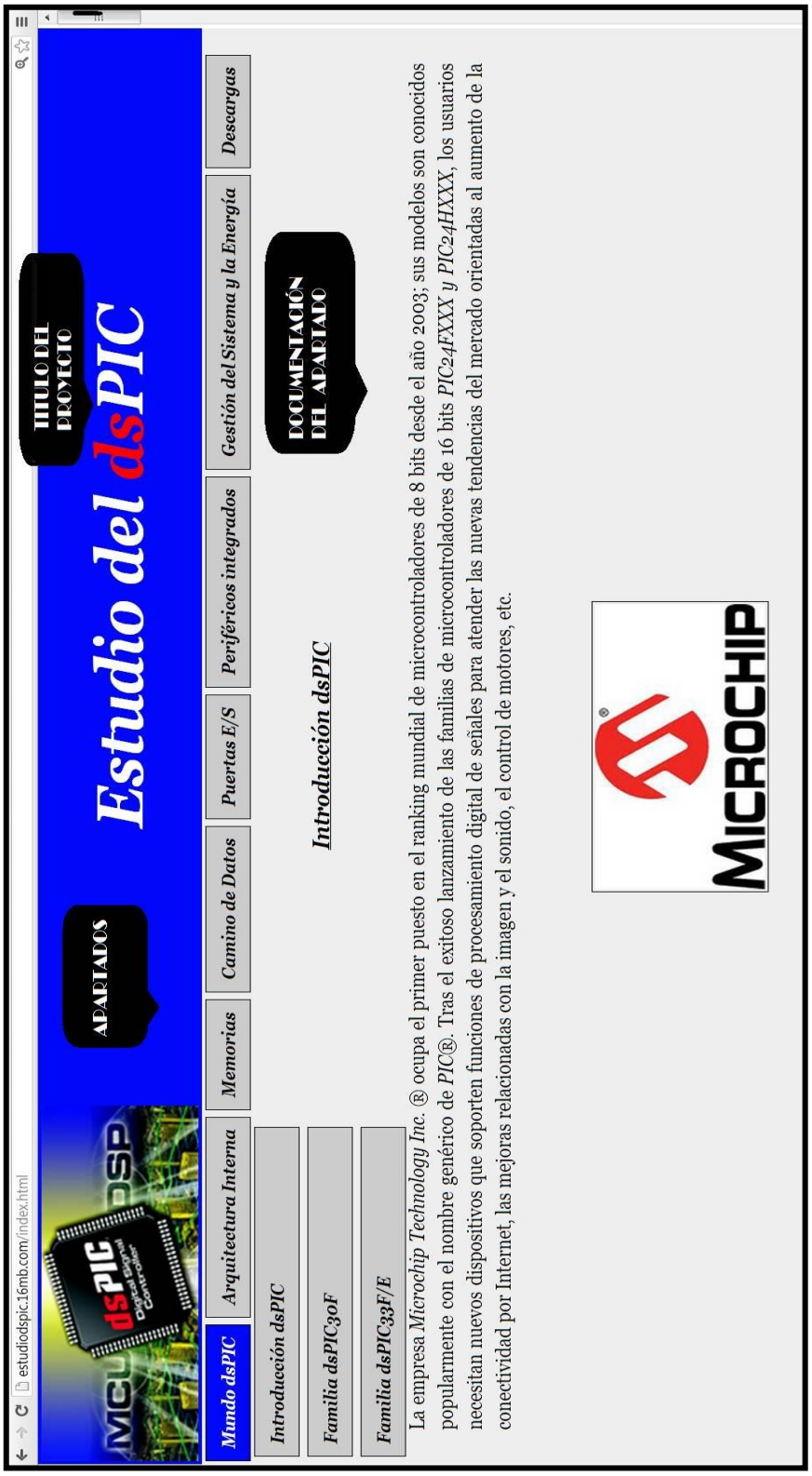
Requerimientos de Software:

1) El sitio está diseñado para poderse visualizar en cualquiera de los navegadores existentes en el mercado.
2) La resolución de la WEB esta optimizada para resoluciones de pantalla de 1280 x 800 ppp o superiores, en la que hemos tenido en cuenta, los diferentes modelos de portátiles, notebooks y ordenadores de sobremesa, fijándonos en el ranking mundial de utilización de resoluciones.

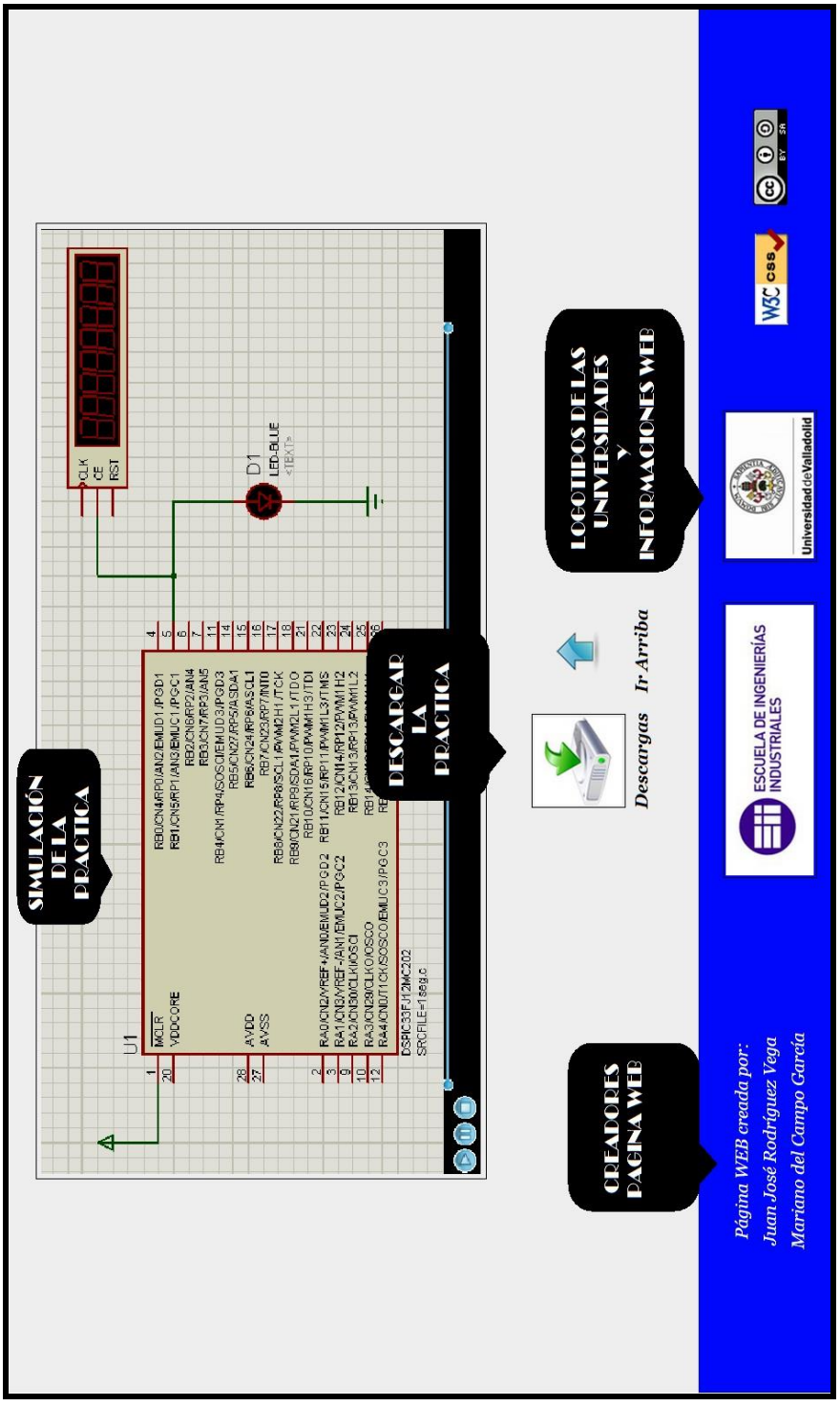
6.2. PÁGINA PRINCIPAL.

Para acceder a nuestra página web el usuario deberá ingresar la dirección electrónica http://estudiodspic.16mb.com/. Esta es la dirección en la que hemos trabajado en el desarrollo del proyecto, pero al ser una web con finales didácticos y de dominio de la Universidad de Valladolid, esta dirección y sus requerimientos pueden ser cambiados en cualquier momento.

Una vez que el usuario ingresa en el sitio Web, se encontrara con la página principal del sitio, tanto desde esa página principal como desde cualquier apartado se puede acceder a todos los apartados. Por lo que es una Web en la que no te pierdes en submenús y tener que retroceder para poder acceder a otro apartado. La describimos gráficamente a continuación: Ver Cap.1 Y Cap.2



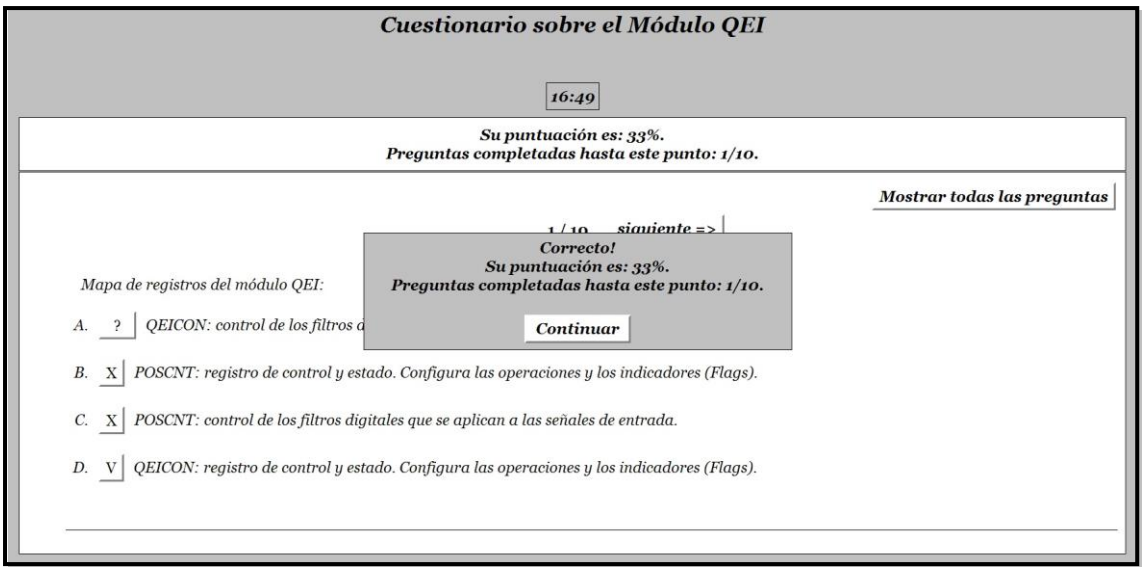
Cap.1. Captura parte superior de la página Web.



Cap.2. Captura de la parte inferior de la página Web.

Al final de cada apartado, en los apartados que corresponde, hay una simulación explicativa del funcionamiento, y un botón de descargas, en el que el usuario se puede descargar la aplicación programada y su correspondiente esquema, para si quiere realizar cambios en la programación u otros cambios en el programa.

También al final de cada apartado, existe un botón llamado cuestionario, en el que el usuario al pinchar, accede a otro sitio web en el que realiza una prueba de sus conocimientos teóricos adquiridos en el estudio de cada apartado. Este cuestionario pone las respuestas y las preguntas de forma aleatoria cada vez que se entra. Y te devuelve un porcentaje de acierto para comprobar los conocimientos. Ver Cap.3.



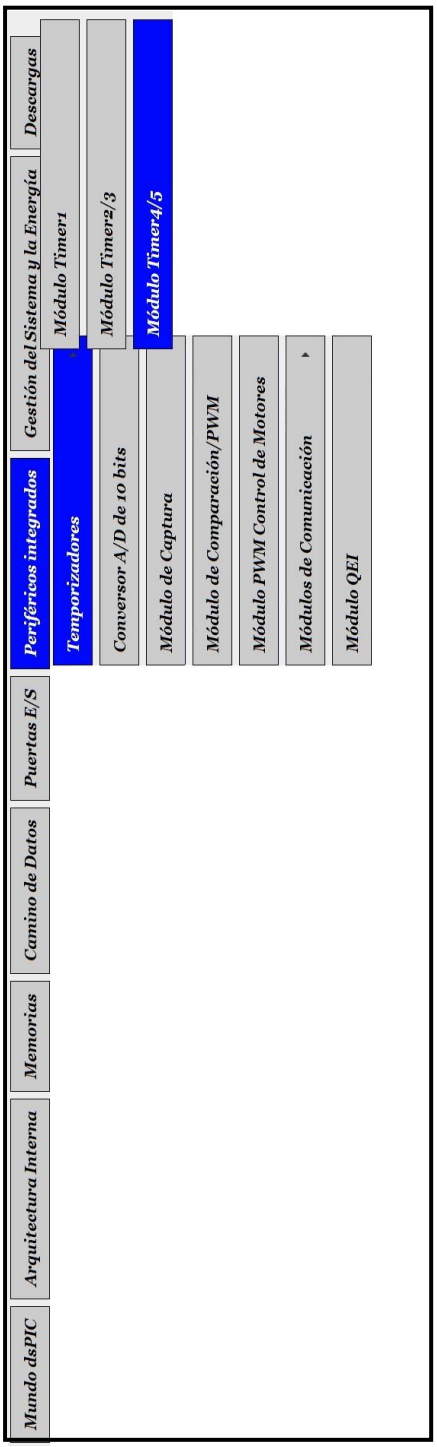
Cap.3. Captura de cuestionario.

6.3. ACCESO.

En nuestro diseño principal, la página es de acceso libre para cualquier usuario, tan solo hace falta introducir la dirección web para poder disfrutar de toda la información en ella contenida sin ninguna restricción. Cuando el proyecto este presentado y este pase a ser propiedad de la Universidad, puede que las condiciones cambien y se necesite ser alumno para poder entrar en la página web.

6.4. MENU DE NAVEGACIÓN.

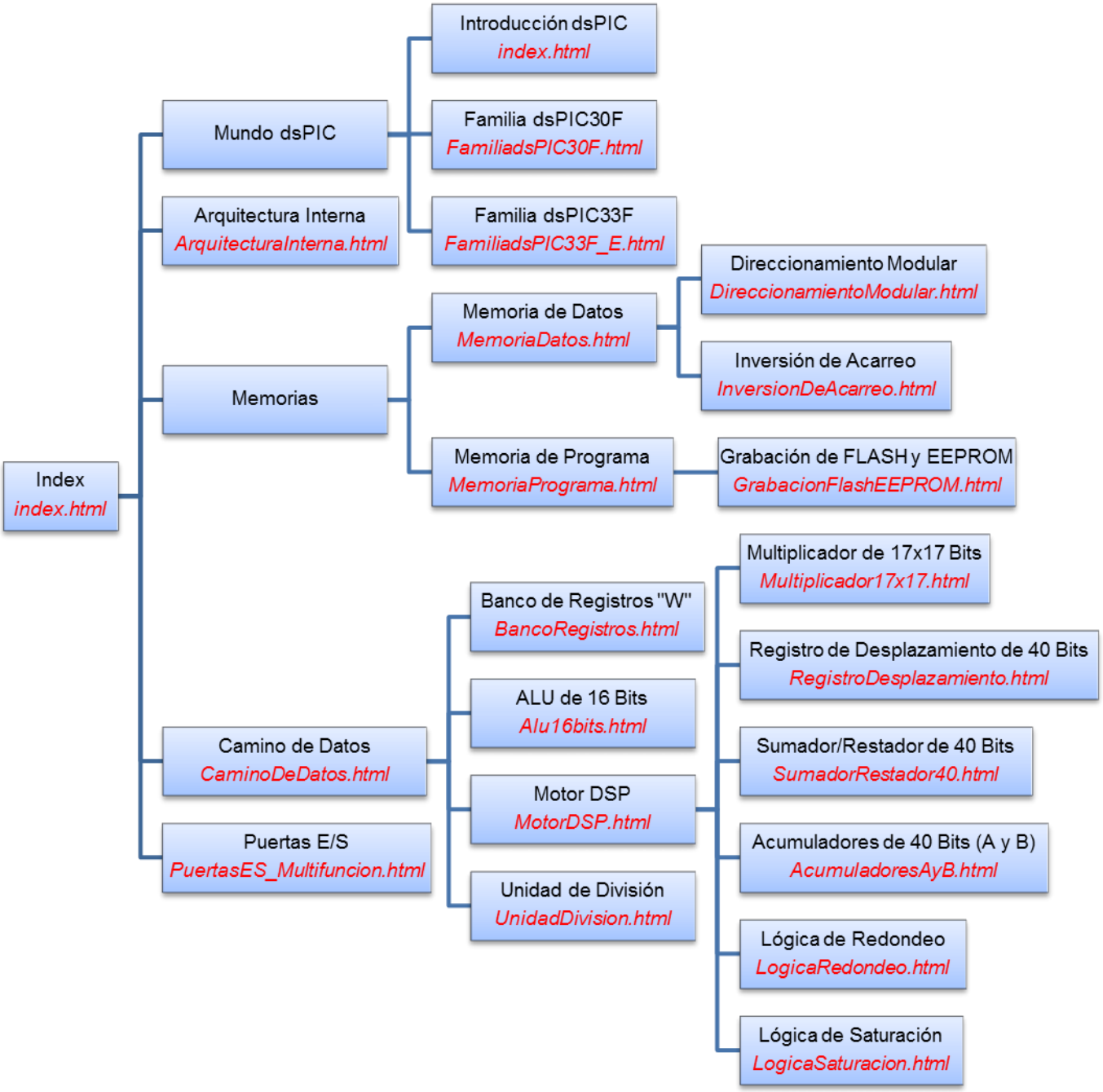
El menú de navegación de nuestra página Web es muy fácil y muy intuitivo, y se puede acceder a cualquier apartado desde cualquier sitio en que estemos navegando, cosa que hace más sencilla la navegación. Ver Cap.4.



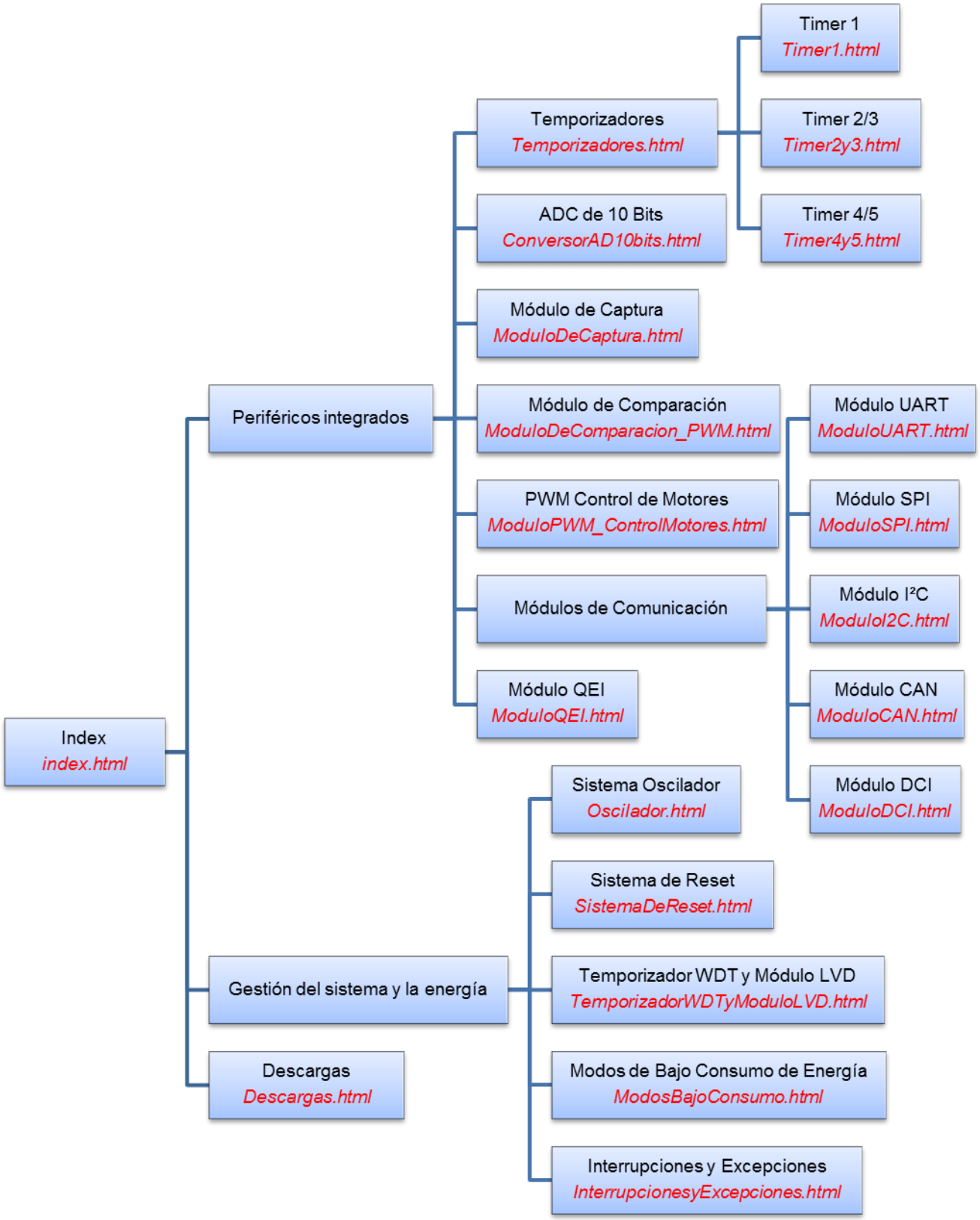
Cap.4. Captura de la parte de debajo de la página Web.

Los submenús de cada parte del estudio se despliegan del menú de navegación, con títulos claros e intuitivos.

PLANOS



UNIVERSIDAD DE VALLADOLID ***ESCUELA DE INGENIERÍAS INDUSTRIALES***	
PROYECTO FIN DE CARRERA ***ESTUDIO DEL MICROCONTROLADOR AVANZADO dsPIC***	
MAPA WEB	***PLANO Nº:*** *01-A*
LOS INGENIEROS TÉCNICOS: *RODRÍGUEZ VEGA, JUAN JOSÉ* *DEL CAMPO GARCÍA, MARIANO*	***ESCALA:*** *S/E* ***FECHA:*** *23/05/2013*



UNIVERSIDAD DE VALLADOLID ***ESCUELA DE INGENIERÍAS INDUSTRIALES***	
PROYECTO FIN DE CARRERA ***ESTUDIO DEL MICROCONTROLADOR AVANZADO dsPIC***	
MAPA WEB	***PLANO Nº:*** *01-B*
LOS INGENIEROS TÉCNICOS: *RODRÍGUEZ VEGA, JUAN JOSÉ* *DEL CAMPO GARCÍA, MARIANO*	***ESCALA:*** *S/E* ***FECHA:*** *23/05/2013*

PLIEGO DE CONDICIONES

PLIEGO DE CONDICIONES

ÍNDICE:

1. **LICENCIA PARA LAS APLICACIONES** **04**
1.1. PREAMBULO 04
1.2. TERMINOS Y CONDICIONES 06
1.2.1. Definiciones 06
1.2.2. Código fuente 07
1.2.3. Permisos básicos 08
1.2.4. Protección de los derechos legales de los usuarios frente a la ley anti-evasión 08
1.2.5. Transmisión de copias exactas 09
1.2.6. Transmisión de versiones modificadas del código fuente 09
1.2.7. Transmisión de códigos que no son códigos fuente 10
1.2.8. Términos adicionales 12
1.2.9. Cancelación 14
1.2.10. Aceptación innecesaria para la posesión de copias 14
1.2.11. Traspaso automático de licencia a destinatarios subsiguientes 15
1.2.12. Patentes 15
1.2.13. Protección de la libertad de terceros 17
1.2.14. Uso conjunto de la licencia Pública General Affero de GNU 18
1.2.15. Revisiones de esta licencia 18
1.2.16. Ausencia de garantías 19
1.2.17. Limitación de la responsabilidad 19
1.2.18. Interpretación de los apartados 1.2.16 y 1.2.17 19
1.3. APLICACIÓN 19
2. LICENCIA PARA LA WEB **21**
2.1. LO QUE HACE LA LICENCIA 21
2.2. DISEÑO Y JUSTIFICACIÓN DE LA LICENCIA 21
2.3. TRES “CAPAS” DE LAS LICENCIAS 22
2.4. LA LICENCIA 23
2.5. RESUMEN DE LA LICENCIA 24
2.6. CODIGO LEGAL 25
2.7. LICENCIA CCPL 25
2.7.1. Definiciones 25
2.7.2. Límites de los derechos 28
2.7.3. Concesión de la licencia 28
2.7.4. Restricciones 29

2.7.5. Exoneración de responsabilidad 31
2.7.6. Limitación de responsabilidad 32
2.7.7. Finalización de la licencia 32
2.7.8. Miscelánea 32
2.7.9. Aviso de Creative Commons 33

PLIEGO DE CONDICIONES

1. LICENCIA PARA LAS APLICACIONES

LICENCIA PÚBLICA GENERAL DE GNU

Versión 3, 29 de junio de 2007

1.1. PREÁMBULO

La Licencia Pública General de GNU (GNU GPL, por sus siglas en inglés) es una licencia libre y gratuita con derecho de copia para software y otros tipos de obras.

Las licencias para la mayoría del software y otras obras de índole práctica están diseñadas para privarle de la libertad para distribuir y modificar las obras. Por el contrario, la Licencia Pública General de GNU garantiza la libre distribución y modificación de todas las versiones de un programa, a fin de asegurarle dicha libertad a todos los usuarios. En la Fundación para el Software Libre utilizamos la Licencia Pública General de GNU para la mayoría de nuestro software; también se aplica a cualquier otra obra publicada de esta manera por sus autores. Usted también puede aplicarla a sus programas.

Cuando hablamos de software libre, nos referimos a la libertad, no al precio. Nuestras Licencias Públicas Generales están diseñadas para garantizarle a usted la libertad de distribuir copias de software libre (y cobrar por ellas, si así lo desea), obtener el código fuente, o tener la posibilidad de obtenerlo, modificar el software o utilizar partes del mismo en nuevos programas libres, y saber que puede hacer estas cosas.

Para proteger sus derechos, necesitamos evitar que otros le nieguen estos derechos o le pidan que renuncie a los mismos. Por lo tanto, en el caso de que usted distribuya o modifique este software, tendrá ciertas responsabilidades a fin de garantizar la libertad de los demás.

Por ejemplo, si usted distribuye copias de un programa de esta naturaleza, ya sea en forma gratuita o a cambio de dinero, debe extender a los destinatarios del software las mismas libertades que le fueron otorgadas a usted. Debe asegurarse de que ellos también reciban o tengan la posibilidad de obtener el código fuente. Y debe mostrarles los presentes términos a fin de que conozcan sus derechos.

Los desarrolladores que utilizan la GNU GPL siguen dos pasos para proteger los derechos que usted recibe: (1) declarar los derechos de autor del software, y (2) ofrecerle esta Licencia para que usted pueda copiar, distribuir y/o modificar el software legalmente.

A fin de proteger a los desarrolladores y autores, la GPL explica claramente que no se ofrecen garantías por este software libre. Por el bien de los usuarios y de los autores, la GPL exige que las versiones modificadas se identifiquen como tales, de modo que los problemas que puedan contener estas versiones no se atribuyan erróneamente a los autores de versiones anteriores.

Existen algunos dispositivos diseñados para negarles a los usuarios el acceso para instalar o ejecutar versiones modificadas del software que contienen, aunque el fabricante pueda hacerlo. Esto es esencialmente incompatible con el objetivo de proteger la libertad de los usuarios para modificar el software. El patrón sistemático de tal abuso se da en el área de productos para el uso por parte de individuos, precisamente un área en la cual se vuelve más inaceptable. Por lo tanto, hemos diseñado esta versión de la GPL a fin de prohibir la práctica para dichos productos. En el caso de que dichos problemas surgieran en otras esferas, hemos tomado los recaudos necesarios para extender esta disposición a dichas esferas en futuras versiones de la GPL, según se requiera para proteger la libertad de los usuarios.

Por último, todos los programas se ven amenazados constantemente por patentes de software. Los Estados no deberían permitirles a las patentes restringir el desarrollo y el uso de software en computadoras para fines generales, pero, en el caso de que esto suceda, deseamos evitar el riesgo especial de que las patentes que se apliquen a un programa libre efectivamente otorguen tal exclusividad. Para lograrlo, la GPL garantiza la imposibilidad del uso de las patentes para apropiarse de un programa y restringir dicha libertad.

A continuación, se incluyen los términos y condiciones particulares para la reproducción, distribución y modificación del software.

1.2. TÉRMINOS Y CONDICIONES

1.2.1. DEFINICIONES.

Por “esta Licencia” se entiende la versión 3 de la Licencia Pública General de GNU.

El término “copyright” también se extiende a las leyes que protegen los derechos de autor para otros tipos de obras, tales como diseños de circuitos integrados sobre substrato semiconductor.

Por “el Programa” se entiende cualquier obra incluida en esta Licencia sobre la que se puedan ejercer derechos de autor. Para referirnos a cada licenciatario, utilizamos el término “usted”. Los “licenciatarios” y “destinatarios” pueden ser individuos u organizaciones.

Por “modificar” una obra se entiende el proceso de copiar o adaptar una obra en forma parcial o total de un modo que requiera autorización de copyright y que no sea la reproducción de una copia exacta. La obra resultante es una “versión modificada” de la obra anterior o una obra “basada en” la obra anterior.

Por “obra amparada” se entiende el Programa sin modificaciones o una obra basada en el Programa.

Por “propagar” una obra se entiende cualquier acción sobre la misma que, en el caso de no tener autorización, pudiera hacerlo responsable, ya sea en forma directa o indirecta, de infringir las leyes de derechos de autor aplicables, salvo que dicha acción se realice en una computadora o se modifique una copia privada. La propagación incluye la reproducción, distribución (con o sin modificaciones), divulgación y, en algunos países, otras actividades también.

Por “transmitir” una obra se entiende cualquier tipo de propagación que le permita a un tercero hacer o recibir copias. La mera interacción con un usuario a través de una red informática, cuando no se transfiere una copia, no se considera una transmisión.

Una interfaz de usuario interactiva muestra “avisos legales apropiados” en la medida en que, de un modo práctico y bien visible, (1) muestre un aviso de copyright apropiado y (2) le informe al usuario que no se ofrecen garantías por la obra (salvo que efectivamente se ofrezcan garantías) y que los licenciatarios pueden transmitir la obra conforme a las disposiciones de esta Licencia, además de mostrar la forma en que se puede consultar una

copia de esta Licencia. Si la interfaz presenta una lista de comandos del usuario u opciones, tales como un menú, dicha lista debe incluir un ítem visible que cumpla con este criterio.

1.2.2. CÓDIGO FUENTE.

El "código fuente" de una obra es el formato preferido de la obra para realizar modificaciones en la misma. Por "código objeto" se entiende cualquier formato de una obra que no sea código fuente.

Una "interfaz estándar" es una interfaz que puede ser una norma oficial, según lo defina un organismo de normas reconocido, o bien, en el caso de interfaces específicas para un lenguaje de programación particular, una interfaz de uso generalizado entre los desarrolladores que trabajan con dicho lenguaje.

Las "bibliotecas de sistemas" de una obra ejecutable comprenden cualquier cosa, salvo la obra en su totalidad, que (a) se incluya en la forma normal de empaquetamiento de un Componente Importante, pero que no forme parte del Componente Importante, y (b) sirva únicamente para permitir el uso de la obra con dicho Componente Importante o para implementar una Interfaz Estándar para la cual haya a disposición del público una implementación en forma de código fuente. Un "Componente Importante", en este contexto, es un componente fundamental (núcleo, sistema de ventanas, etc.) del sistema operativo específico (si hubiera) en el que funcione la obra ejecutable, o un compilador utilizado para producir la obra, o un intérprete de código objeto utilizado para ejecutarlo.

La "Fuente Correspondiente" para una obra en código objeto refiere a todo el código fuente necesario para generar, instalar y (para una obra ejecutable) ejecutar el código objeto y modificar la obra, incluidas las secuencias de comandos para controlar dichas actividades. Sin embargo, no incluye las Bibliotecas de Sistemas de la obra, así como tampoco herramientas de aplicación general o programas libres generalmente disponibles que se utilicen sin modificaciones para realizar dichas actividades pero que no formen parte de la obra. Por ejemplo, la Fuente Correspondiente incluye los archivos de definición de interfaz asociados a los archivos fuente para la obra, así como el código fuente para las bibliotecas compartidas y los subprogramas vinculados en forma dinámica requeridos específicamente conforme a su diseño, por ejemplo, mediante la comunicación de datos intrínseca o el control de flujo entre esos subprogramas y otras partes de la obra.

La Fuente Correspondiente no necesita incluir nada que los usuarios puedan regenerar automáticamente de otras partes de la Fuente Correspondiente.

La Fuente Correspondiente para una obra en código fuente es esa misma obra.

1.2.3. PERMISOS BÁSICOS.

Todos los derechos que se otorgan conforme a esta Licencia se otorgan por el término del copyright que ampara al Programa y son irrevocables siempre y cuando se cumplan las condiciones establecidas. Esta Licencia lo autoriza en forma expresa e ilimitada a ejecutar el Programa sin modificaciones. El producto obtenido a partir de la ejecución de una obra amparada está cubierto por esta Licencia únicamente si el producto, dado su contenido, constituye una obra amparada. Esta Licencia reconoce sus derechos de uso razonable y otros equivalentes, conforme a las leyes de copyright.

Usted puede crear, ejecutar y propagar obras amparadas que no transmita, sin condiciones en la medida en que su licencia siga vigente de alguna otra manera. Usted puede transmitir obras amparadas a terceros con el único fin de que éstos realicen modificaciones exclusivamente para usted, o que le proporcionen los medios para ejecutar dichas obras, siempre y cuando usted cumpla con los términos de esta Licencia en lo que respecta a la transmisión de cualquier material que exceda su control del copyright. Aquéllos que de esta manera creen o ejecuten las obras aparadas para usted deben hacerlo exclusivamente en su nombre, bajo su dirección y control y sobre la base de términos que les prohíban hacer copias de su material protegido por derechos de autor fuera de la relación que mantienen con usted.

La transmisión bajo cualquier otra circunstancia se permite únicamente conforme a las condiciones que se describen a continuación. Se prohíbe sub-licenciar; la sección 10 hace que sea innecesario.

1.2.4. PROTECCIÓN DE LOS DERECHOS LEGALES DE LOS USUARIOS FRENTE A LA LEY ANTI-EVASIÓN.

Ninguna obra amparada se considerará parte de una medida tecnológica efectiva conforme a cualquier ley aplicable que cumpla las obligaciones del artículo 11 del tratado de copyright WIPO adoptado el 20 de diciembre de 1996 o a leyes similares que prohíban o restrinjan la evasión de dichas medidas.

Cuando usted transmite una obra amparada, renuncia a cualquier facultad legal de prohibir la evasión de medidas tecnológicas en la medida en que dicha evasión se realice al

hacer uso de los derechos que se otorgan conforme a esta Licencia con respecto a la obra amparada, y niega cualquier intención de restringir el uso o la modificación de la obra como una forma de hacer valer, en contra de los usuarios de la obra, sus derechos legales o los derechos legales de terceros para prohibir la evasión de medidas tecnológicas.

1.2.5. TRANSMISIÓN DE COPIAS EXACTAS.

Usted puede transmitir copias exactas del código fuente del Programa tal cual lo reciba, en cualquier medio, siempre y cuando publique de un modo llamativo y adecuado un aviso de copyright apropiado en cada copia; mantenga intactos todos los avisos que establecen que esta Licencia y cualquier término no-permisivo que se agregue conforme a la sección 7 se aplican al código; mantenga intactos todos los avisos mediante los cuales se niega cualquier tipo de garantía; y les proporcione a todos los destinatarios una copia de esta Licencia junto con el Programa.

Usted puede cobrar el precio que usted desee o no cobrar nada por cada copia que transmita, y puede ofrecer soporte o protección de garantía a cambio de una tarifa.

1.2.6. TRANSMISIÓN DE VERSIONES MODIFICADAS DEL CÓDIGO FUENTE.

Usted puede transmitir una obra basada en el Programa, o las modificaciones para producirlo a partir del Programa, en forma de código fuente conforme a los términos de la sección 4, siempre y cuando también cumpla con todas las condiciones que se incluyen a continuación:

- a) La obra debe conservar avisos llamativos que establezcan que usted la ha modificado e incluyan la fecha correspondiente.
- b) La obra debe conservar avisos llamativos que establezcan que la misma se realiza conforme a esta Licencia y a todas las condiciones que se agreguen bajo la sección 7. Este requerimiento modifica el requerimiento de la sección 4 que establece que se deben “mantener intactos todos los avisos”.
- c) Usted debe otorgar una licencia por la obra completa, en forma íntegra, conforme a esta Licencia, a cualquier tercero que adquiera una copia. Por lo tanto, esta Licencia, junto con cualquier término adicional aplicable de la sección 7, se aplica a la obra en su totalidad y a todas sus partes, independientemente del modo en que se las empaquete.

Esta Licencia no lo autoriza a otorgar licencias para la obra de ningún otro modo, pero no invalida dicha autorización si usted la ha recibido por separado.

- d) Si la obra tuviera interfaces de usuario interactivas, cada una de ellas deberá mostrar Avisos Legales Apropiados. No obstante, si el Programa tuviera interfaces interactivas que no mostraran Avisos Legales Apropiados, usted no necesita incluirlos.

Se denomina “conjunto” a la compilación de una obra amparada con otras obras diferentes e independientes que por su naturaleza no sean extensiones de la obra amparada ni se combinen con ella para formar un programa más grande en un volumen de un medio de distribución o almacenamiento, si la compilación y el copyright consiguiente no se utilizan para restringir el acceso o los derechos legales de los usuarios de la compilación más allá de lo que permitan las obras individuales. La inclusión de una obra amparada en un conjunto no implica que esta Licencia se aplique a las otras partes del conjunto.

1.2.7. TRANSMISIÓN DE CÓDIGOS QUE NO SON CÓDIGOS FUENTE.

Usted puede transmitir una obra amparada en código objeto conforme a los términos de las secciones 4 y 5, siempre y cuando también transmita la Fuente Correspondiente legible por máquina conforme a los términos de esta Licencia, de alguna de las siguientes maneras:

- a) Transmisión del código objeto dentro de un producto físico (incluidos medios físicos de distribución) o incorporado a éste, acompañado de la Fuente Correspondiente en un medio físico duradero habitual para el intercambio de software.
- b) Transmisión del código objeto dentro de un producto físico (incluidos medios físicos de distribución) o incorporado a éste, acompañado de una oferta escrita, que sea válida por un plazo mínimo de tres años y por el tiempo que usted ofrezca repuestos o soporte técnico para ese modelo del producto, para proporcionarle a cualquier persona que posea el código objeto (1) una copia de la Fuente Correspondiente para todo el software del producto que esté amparado por esta Licencia, en un medio físico duradero habitual para el intercambio de software, a cambio de un precio que no exceda el costo razonable de la acción física de transmitir esta fuente, o (2) acceso para la copia de la Fuente Correspondiente desde un servidor de red sin costo alguno.
- c) Transmisión de copias individuales del código objeto junto con una copia de la oferta escrita para proporcionar la Fuente Correspondiente. Esta opción se permite únicamente en ocasiones y para fines no comerciales, y sólo en la medida en que usted haya recibido el código objeto con una oferta de esta naturaleza, conforme a la subsección 6b.

- d) Transmisión del código objeto ofreciendo acceso desde un lugar determinado (en forma gratuita u onerosa) y ofreciendo un acceso equivalente a la Fuente Correspondiente del mismo modo y desde el mismo lugar sin costo adicional. No es necesario que les exija a los destinatarios que copien la Fuente Correspondiente junto con el código objeto. Si el lugar ofrecido para copiar el código objeto fuera un servidor de red, la Fuente Correspondiente podrá estar en un servidor diferente (operado por usted o un tercero) que ofrezca posibilidades de reproducción equivalentes, siempre y cuando se incluyan, junto al código objeto, instrucciones claras para localizar la Fuente Correspondiente. Independientemente de qué servidor albergue la Fuente Correspondiente, usted mantiene la obligación de asegurarse de que el mismo esté disponible por el tiempo que sea necesario para satisfacer estos requerimientos.
- e) Transmisión del código objeto mediante transferencia entre usuarios (peer to peer), siempre y cuando les informe a los usuarios la ubicación del código objeto y la Fuente Correspondiente de la obra para el público en general sin costo alguno conforme a la subsección 6d.

No se necesita incluir una parte separable del código objeto, cuyo código fuente se excluya de la Fuente Correspondiente como una biblioteca de sistemas, para transmitir el código objeto de la obra.

Por “producto de usuario” se entiende (1) un “producto de consumo”, que es cualquier bien personal tangible que se utilice habitualmente para fines personales, familiares o domésticos, o (2) cualquier cosa que se diseñe o comercialice para su incorporación en una vivienda. Al determinar si un producto es un producto de consumo, los casos dudosos deberán resolverse a favor del amparo. Para un producto específico que recibe un usuario particular, un “uso habitual” es el uso común o típico que se le suele dar a ese tipo de producto, independientemente de la condición del usuario particular o de la forma en que el usuario particular utilice el producto o de las expectativas propias o de terceros con respecto al uso del producto. Un producto se considera un producto de consumo independientemente de que se le pueda dar usos sustanciales de índole comercial, industrial o ajena al consumo, salvo que dichos usos representen el único modo significativo de utilizar el producto.

Por “información de instalación” de un producto de usuario se entiende cualquier método, procedimiento, clave de autorización u otro tipo de información requerida para instalar y ejecutar versiones modificadas de una obra amparada en dicho producto de usuario a partir de una versión modificada de su Fuente Correspondiente. La información debe ser suficiente para garantizar que el funcionamiento continuo del código objeto modificado no se vea afectado o imposibilitado por el solo hecho de haberse realizado la modificación.

En el caso de que usted transmita el código objeto de una obra conforme a esta sección en un producto de usuario, junto con un producto de usuario o específicamente para su uso en un producto de usuario, y la transmisión se produzca como parte de una transacción mediante la cual los derechos de posesión y uso del producto de usuario se transfieran al destinatario por un plazo limitado o ilimitado (independientemente de las particularidades de la transacción), la Fuente Correspondiente transmitida conforme a esta sección deberá ir acompañada de la información de instalación. Sin embargo, este requerimiento no se aplicará en el caso de que ni usted ni un tercero conserven la capacidad para instalar el código objeto modificado en el producto de usuario (por ejemplo, que la obra se haya instalado en memoria ROM).

El requerimiento de proporcionar información de instalación no implica la necesidad de seguir proveyendo soporte técnico, garantías o actualizaciones para una obra que haya sido modificada o instalada por el destinatario o para el producto de usuario en el cual se la haya modificado o instalado. Podrá negarse el acceso a una red cuando la modificación en sí misma pueda afectar de un modo adverso y sustancial el funcionamiento de la red o infrinja las normas y los protocolos de comunicación a través de la red.

La Fuente correspondiente que se transmita y la información de instalación que se proporcione conforme a esta sección deberán presentarse en un formato sobre el cual exista documentación pública (y con una implementación disponible para el público en código fuente) y no deberán requerir ninguna clave o contraseña especial para su desempaquetamiento, lectura o reproducción.

1.2.8. TÉRMINOS ADICIONALES.

Los “permisos adicionales” son términos que complementan los términos de esta Licencia al permitir excepciones a una o más condiciones. Los permisos adicionales que se aplican al Programa en su totalidad deberán tratarse como si formaran parte de esta Licencia, en la medida en que sean válidos conforme a las leyes aplicables. En el caso de que los permisos adicionales se apliquen únicamente a una parte del Programa, esta parte podrá utilizarse por separado conforme a dichos permisos, pero el Programa en su totalidad seguirá rigiéndose de acuerdo a esta Licencia independientemente de los permisos adicionales.

Cuando usted transmita una copia de una obra amparada, podrá optar por eliminar cualquier permiso adicional de dicha copia o de cualquier parte de la misma (en ciertos casos, cuando usted modifique la obra, podrán establecerse permisos adicionales para requerir la eliminación de los mismos). Tiene autorización para incluir permisos adicionales en un material

que usted haya agregado a una obra amparada y sobre el cual usted posea o pueda otorgar permisos de copyright adecuados.

Independientemente de cualquier otra disposición de esta Licencia, con respecto al material que usted agregue a una obra amparada, usted podrá (en la medida en que lo autoricen los titulares de los derechos de copyright de dicho material) complementar los términos de esta Licencia con los siguientes términos:

- a) Ausencia de garantías o limitación de la responsabilidad más allá de los términos de las secciones 15 y 16 de esta Licencia; o
- b) Obligación de conservación de atribuciones de autoría o avisos legales razonables específicos en dicho material o en los avisos legales apropiados que se muestren en las obras que lo contengan; o
- c) Prohibición de tergiversación del origen del material, o requerimiento de que en las versiones modificadas de dicho material se indique de un modo razonable que son diferentes de la versión original; o
- d) Limitación del uso de los nombres de los licenciantes o autores del material para fines publicitarios; o
- e) Negativa con respecto al otorgamiento de derechos conforme a las leyes de marcas para el uso de ciertos nombres comerciales, marcas de productos o marcas de servicios; o
- f) Requerimiento de indemnización de los licenciantes o autores de dicho material por parte de cualquier persona que transmita el material (o versiones modificadas del mismo) bajo presunciones contractuales de responsabilidad del destinatario por cualquier responsabilidad que dichas presunciones contractuales impongan directamente sobre los licenciantes y autores del material.

Cualquier otro término adicional no permisivo se considerará una “restricción adicional” en el contexto de la sección 10. En el caso de que el Programa, tal cual usted lo recibió, o cualquier parte del mismo contengan un aviso que indique que el mismo se rige según esta Licencia junto con un término que constituya una restricción adicional, podrá eliminar dicho término. En el caso de que un documento de licencia contenga una restricción adicional pero permita la extensión de la licencia o la transmisión del programa conforme a esta Licencia, usted podrá agregar a la obra amparada cualquier material conforme a los términos de dicho documento de licencia, siempre y cuando la restricción adicional no se mantenga tras la extensión de la licencia o la transmisión del programa.

En el caso de que usted agregue términos a una obra amparada conforme a esta sección, deberá incluir en los archivos fuente correspondientes una declaración de los términos

adicionales que se aplican a dichos archivos o un aviso que indique la ubicación de los términos aplicables.

Se podrán establecer términos adicionales, sean éstos permisivos o no permisivos, en una licencia escrita independiente, o a modo de excepciones; sea como fuere, se aplicarán los requerimientos mencionados anteriormente.

1.2.9. CANCELACIÓN.

Usted no está autorizado a propagar o modificar una obra amparada de ningún otro modo que no se estipule en esta Licencia. Cualquier intento no autorizado por propagarla o modificarla se considerará nulo y conllevará la cancelación automática de los derechos que le haya otorgado esta Licencia (incluida cualquier licencia de patente otorgada conforme al párrafo tercero de la sección 11).

No obstante, en el caso de que deje de violar las cláusulas de esta Licencia, un titular de derechos de copyright particular podrá restituirle la licencia (a) en forma provisoria, hasta tanto dicho titular dé por finalizada su licencia en forma expresa y definitiva, y (b) en forma permanente, si dicho titular no lo notificara de la infracción por algún medio razonable antes de los 60 días posteriores a la cancelación.

Asimismo, la licencia que le otorgue un titular de derechos de copyright particular se le restituirá en forma permanente si dicho titular lo notificara de la infracción por algún medio razonable, ésta fuera la primera vez que usted hubiese recibido una notificación de violación de esta Licencia (por cualquier obra) emitida por dicho titular, y usted subsanara la infracción en un plazo de 30 días a partir de la recepción de la notificación.

La extinción de sus derechos conforme a esta sección no cancela las licencias de aquellos terceros a los que usted les haya otorgado copias o derechos conforme a esta Licencia. En el caso de que sus derechos se cancelen y no se le restituyan en forma permanente, usted no estará capacitado para recibir nuevas licencias para el mismo material conforme a la sección 10.

1.2.10. ACEPTACIÓN INNECESARIA PARA LA POSESIÓN DE COPIAS.

Usted no está obligado a aceptar esta Licencia para poder recibir o ejecutar una copia del Programa. De modo similar, la propagación secundaria de una obra amparada que se

produzca únicamente como consecuencia de una transferencia entre usuarios (peer to peer) a fin de recibir una copia tampoco requiere aceptación. No obstante, esta Licencia es lo único que lo autoriza a propagar o modificar cualquier obra amparada. En el caso de que usted no acepte esta Licencia, los actos anteriores representarán una violación de las leyes de copyright. Por lo tanto, al modificar o propagar una obra amparada, usted expresa su aceptación de esta Licencia para hacerlo.

1.2.11. TRASPASO AUTOMÁTICO DE LICENCIA A DESTINATARIOS SUBSIGUIENTES.

Cada vez que usted transmite una obra amparada, el destinatario recibe automáticamente de los licenciantes originales una licencia para ejecutar, modificar y propagar la obra conforme a esta Licencia. Usted no es responsable de asegurar el cumplimiento de esta Licencia por parte de terceros.

Una “transacción entre entidades” es una transacción mediante la cual se transfiere el control de una organización o de todo el patrimonio de una organización, se subdivide una organización o se fusionan dos o más organizaciones. En el caso de que la propagación de una obra amparada se deba a una transacción entre entidades, cada parte de la transacción que reciba una copia de la obra también recibirá todas las licencias para la obra que el predecesor de la parte tuviera o pudiera otorgar conforme al párrafo anterior, más el derecho de recibir de su predecesor la Fuente Correspondiente de la obra, si el predecesor la tuviera en su poder o pudiera obtenerla con un esfuerzo razonable.

Usted no puede imponer restricciones adicionales para el ejercicio de los derechos que se otorgan o consolidan conforme a esta Licencia. Por ejemplo, usted no puede imponer tarifas, regalías u otros cargos a cambio del ejercicio de los derechos que se otorgan conforme a esta Licencia, así como tampoco puede iniciar acciones legales (incluidas demandas y contrademandas en un pleito) sobre la base de una infracción de patentes por crear, usar, comercializar, ofrecer para la venta o importar el Programa o cualquier parte del mismo.

1.2.12. PATENTES.

Un “colaborador” es un titular de derechos de copyright que autoriza el uso conforme a esta Licencia del Programa o de una obra sobre la cual se base el Programa. La obra cuya licencia se otorgue de esta manera se denomina “versión en colaboración” del colaborador.

Los “derechos de patente fundamentales” de un colaborador son todos los derechos de patente bajo la titularidad o el control del colaborador, ya sea que se los hubiese adquirido previo al otorgamiento de esta Licencia o a partir del mismo, que puedan infringirse de algún modo permitido por esta Licencia para crear, usar o vender su versión en colaboración, pero no incluyen derechos que se podrían infringir únicamente como consecuencia de modificaciones posteriores a la versión en colaboración. A los efectos de esta definición, el “control” incluye el derecho de otorgar sub-licencias de patente de un modo acorde a los requerimientos de esta Licencia.

Cada colaborador le otorga a usted una licencia de patente internacional no-exclusiva libre de regalías conforme a los derechos de patente fundamentales del colaborador para crear, usar, comercializar, ofrecer para la venta, importar y ejecutar, modificar y propagar de algún otro modo el contenido de su versión en colaboración.

En los tres párrafos que se incluyen a continuación, una “licencia de patente” es cualquier contrato o acuerdo expreso, independientemente de su denominación, mediante el cual se convenga no ejercer derechos de patente (como, por ejemplo, una autorización expresa para hacer uso de una patente o una cláusula que establezca que no se iniciarán acciones legales por infringir los derechos de patente). Por “otorgar” una licencia de patente de esta naturaleza a otra parte se entiende el acto de celebrar un contrato o acuerdo mediante el cual se conviene no ejercer derechos de patente en contra de dicha parte.

En el caso de que usted transmita una obra amparada, a sabiendas de que está sujeta a una licencia de patente, y la Fuente Correspondiente de la obra no estuviera disponible para su reproducción, en forma gratuita y conforme a los términos de esta Licencia, a través de un servidor de red de acceso público u otro medio igualmente accesible, usted deberá (1) poner la Fuente Correspondiente a disposición del destinatario subsiguiente, (2) renunciar al beneficio de la licencia de patente para esta obra en particular, o bien (3) tomar las medidas necesarias para extender la licencia de patente a los destinatarios subsiguientes de un modo acorde a los requerimientos de esta Licencia. La frase “a sabiendas de que está sujeta a una licencia de patente” significa que usted efectivamente sabe que, de no ser por la licencia de patente, su transmisión de la obra amparada en un país o el uso que pudiera darle el destinatario a la obra amparada en un país infringirían una o más patentes identificables en dicho país que usted considera válidas por diversas razones.

En el caso de que, en relación con una transacción o contrato individual, usted transmitiera una obra amparada o la propagara consiguiendo su transmisión y otorgara a algunas de las partes que reciban la obra amparada una licencia de patente que las autorizara a usar, propagar, modificar o trasmitir una copia específica de la obra amparada, la licencia de

patente que usted otorgue se extenderá automáticamente a todos los destinatarios de la obra amparada y a las obras que se basen en ella.

Una licencia de patente se considera “discriminatoria” cuando no incluye dentro de su ámbito de cobertura uno o más de los derechos que se otorgan específicamente conforme a esta Licencia, prohíbe el uso de dichos derechos o se otorga como condición de que no se usen dichos derechos. Usted no debe transmitir una obra amparada si fuese una de las partes de un contrato con un tercero que se dedicara a la distribución de software, conforme al cual usted debiera pagarle al tercero por la actividad que usted realice con respecto a la transmisión de la obra y el tercero le otorgara a cualquiera de las partes que reciban de usted la obra amparada una licencia de patente discriminatoria (a) en relación con las copias de la obra amparada transmitidas por usted (o las copias que se hagan de esas copias), o (b) principalmente para compilaciones o productos específicos que contengan la obra amparada y en relación con éstos, a menos que usted hubiese celebrado dicho contrato o que la patente se hubiese otorgado antes del 28 de marzo de 2007.

Ninguna disposición de esta Licencia deberá interpretarse como excluyente o limitativa de ninguna licencia implícita u otras defensas legales contra infracciones a las que, de otro modo, usted pudiese tener derecho conforme a la ley de propiedad intelectual vigente.

1.2.13. PROTECCIÓN DE LA LIBERTAD DE TERCEROS.

En el caso de que le fueran impuestas condiciones (ya sea por una orden judicial, un contrato o de algún otro modo) que contradijeran las condiciones de esta Licencia, usted no quedará eximido de cumplir las condiciones de esta Licencia. En el caso de que no pueda transmitir una obra amparada de un modo que le permita cumplir simultáneamente con las obligaciones establecidas por esta Licencia y cualquier otra obligación pertinente, no podrá transmitirla de ningún modo. Por ejemplo, en el caso de que usted acepte términos que lo obliguen a cobrar regalías por retransmisión de aquéllos a los que usted transmita el Programa, la única forma de satisfacer tanto dichos requerimientos como esta Licencia será abstenerse de transmitir el Programa.

1.2.14. USO CONJUNTO CON LA LICENCIA PÚBLICA GENERAL AFFERO DE GNU.

Independientemente de cualquier otra disposición de esta Licencia, usted tiene permiso para vincular o combinar cualquier obra amparada con una obra cuya licencia se otorgue conforme a la versión 3 de la Licencia Pública General Affero de GNU en una única obra combinada y transmitir la obra resultante. Los términos de esta Licencia seguirán aplicándose a la parte que corresponda a la obra amparada, pero los requerimientos especiales de la sección 13 de la Licencia Pública General Affero de GNU sobre la interacción a través de una red se aplicarán a la combinación como tal.

1.2.15. REVISIONES DE ESTA LICENCIA.

La Fundación para el Software Libre podrá publicar revisiones y/o versiones nuevas de la Licencia Pública General de GNU de vez en cuando. Tales versiones serán de naturaleza similar a la versión actual, pero podrán diferir en cuanto a los detalles para afrontar nuevos problemas o inquietudes.

Cada versión recibirá un número de versión que la distinga. En el caso de que el Programa especifique que se rige por una versión determinada de la Licencia Pública General de GNU “o cualquier versión posterior”, usted podrá optar por adoptar los términos y condiciones de dicha versión específica o de cualquier versión posterior que publique la Fundación para el Software Libre. En el caso de que el Programa no especifique un número de versión de la Licencia Pública General de GNU, usted podrá regirse por cualquier versión que haya publicado la Fundación para el Software Libre.

Si el Programa especificara que un apoderado puede decidir qué versiones de la Licencia Pública General de GNU pueden aplicarse en el futuro, la declaración pública del apoderado sobre la aceptación de una versión determinada lo autorizará a usted, en forma permanente, a optar por dicha versión para el Programa.

Puede que las versiones posteriores de la licencia le otorguen permisos adicionales o diferentes. No obstante, no se les impondrán obligaciones adicionales a ningún autor o titular de derechos de copyright como resultado de la adopción de la versión posterior que usted elija.

1.2.16. AUSENCIA DE GARANTÍAS.

El programa se ofrece sin ningún tipo de garantías, en la medida en que lo permitan las leyes aplicables, salvo disposición contraria por escrito, los titulares de derechos de copyright y/u otras partes proveen el programa "tal cual" sin garantías de ningún tipo, ya sean expresas o implícitas, incluidas, aunque no en forma taxativa, las garantías implícitas de comerciabilidad y aptitud para un propósito determinado. Usted asume todos los riesgos con respecto a la calidad y el desempeño del programa. En el caso de que el programa tuviera defectos, usted asume el costo de todas las actividades de mantenimiento, reparación o corrección.

1.2.17. LIMITACIÓN DE LA RESPONSABILIDAD.

En ningún caso, salvo que así lo dispongan las leyes aplicables o un contrato por escrito, un titular de derechos de copyright o un tercero que modifique y/o transmita el programa según se autoriza anteriormente será responsable ante usted de cualquier daño, incluidos daños generales, especiales, fortuitos o derivados, que pueda surgir del uso o la incapacidad de uso del programa (incluidos, aunque no taxativamente, la pérdida de información, el suministro de información imprecisa o las pérdidas que puedan sufrir usted o un tercero o la incapacidad del programa para interactuar con otros programas), aun cuando dicho titular o tercero hubiese sido advertido de la posibilidad de tales daños.

1.2.18. INTERPRETACIÓN DE LOS APARTADOS 1.2.16 Y 1.2.17.

En el caso de que las cláusulas de ausencia de garantías y limitación de la responsabilidad anteriores carecieran de validez legal a nivel local conforme a sus términos, los juzgados deberán aplicar las leyes locales que más se asimilen a una exención absoluta de cualquier responsabilidad civil en relación con el Programa, salvo que una copia del Programa estuviera acompañada de una garantía o presunción de responsabilidad a cambio de una tarifa.

1.3. APLICACIÓN

Si usted desarrolla un programa nuevo y desea que el público le encuentre la mayor utilidad posible, la mejor manera de lograrlo es hacer de éste un software libre para que todos lo puedan redistribuir y modificar conforme a estos términos.

Para hacerlo, adjunte los siguientes avisos al programa. Lo más seguro es añadirlos al comienzo de cada archivo fuente a fin de que se establezca de un modo efectivo la exclusión de garantías. Asimismo, cada archivo debería incluir el renglón del “copyright” y un vínculo a la ubicación del aviso completo.

<un renglón para incluir el nombre del programa y una descripción breve de sus funciones.>
Copyright © <año> <nombre del autor>

Este programa es software libre: usted puede redistribuirlo y/o modificarlo conforme a los términos de la Licencia Pública General de GNU publicada por la Fundación para el Software Libre, ya sea la versión 3 de esta Licencia o (a su elección) cualquier versión posterior.

Este programa se distribuye con el deseo de que le resulte útil, pero SIN GARANTÍAS DE NINGÚN TIPO; ni siquiera con las garantías implícitas de COMERCIABILIDAD o APTITUD PARA UN PROPÓSITO DETERMINADO. Para más información, consulte la Licencia Pública General de GNU.

Junto con este programa, se debería incluir una copia de la Licencia Pública General de GNU. De no ser así, ingrese en <http://www.gnu.org/licenses/>.

También incluya información de contacto que les permita a los destinatarios comunicarse con usted, ya sea por correo electrónico o convencional.

Si el programa admite la interacción entre terminales, asegúrese de que muestre un breve aviso como el que se incluye a continuación cuando se inicie en modo interactivo:

<programa> Copyright © <año> <nombre del autor>
Este programa se proporciona SIN GARANTÍAS DE NINGÚN TIPO; para más información escriba 'show w'.
Este programa es software libre y usted puede redistribuirlo conforme a ciertas condiciones; para más información, escriba 'show c'.

Los comandos hipotéticos 'show w' y 'show c' deberían mostrar las partes correspondientes de la Licencia Pública General. De más está decir que los comandos de su programa pueden ser diferentes; para una interfaz gráfica de usuario, debería utilizar un cuadro de diálogo de tipo “acerca de”.

En el caso de que trabaje como programador para un empleador o establecimiento educativo, también asegúrese de que éste firme una “renuncia de copyright” para el programa,

si fuera necesario. Para más información a este respecto y sobre cómo aplicar y cumplir la GNU GPL, ingrese en <http://www.gnu.org/licenses/>.

La Licencia Pública General de GNU no autoriza la inclusión de su programa en programas de propiedad privada. Si su programa fuera una biblioteca de subrutinas, puede que considere más útil permitir la vinculación de aplicaciones de propiedad privada con la biblioteca. Si usted deseara hacer esto, utilice la Licencia Pública General Reducida de GNU en lugar de esta Licencia. Pero primero, por favor, lea la información que se incluye en <http://www.gnu.org/philosophy/why-not-lgpl.html>.

2. LICENCIA PARA LA PÁGINA WEB

2.1. LO QUE HACE LA LICENCIA

Las licencias y herramientas de derechos de autor Creative Commons, genera un equilibrio dentro del escenario tradicional de "todos los derechos reservados" que crean las leyes de propiedad intelectual. Nuestras herramientas entregan a todos, desde creadores individuales a grandes compañías e instituciones, una vía simple y estandarizada de otorgar permisos de derechos de autor con sus trabajos creativos. La combinación de nuestras herramientas y nuestros usuarios es un conjunto de bienes comunes digitales vasto y creciente, una fuente de contenidos que pueden ser copiados, distribuidos, editados, remezclados, y usados como base para crear, todo dentro de los límites del derecho de autor.

2.2. DISEÑO Y JUSTIFICACIÓN DE LA LICENCIA

Todas las licencias Creative Commons tienen importantes características en común. Cada licencia ayuda a los creadores -los llamamos licenciantes si usan nuestras herramientas- a mantener sus derechos de autor al mismo tiempo que permiten a otros copiar, distribuir, y hacer algunos usos de su obra, al menos de forma no comercial. Todas las licencias Creative Commons permiten también que los licenciantes obtengan el crédito que merecen por sus obras. Las licencias Creative Commons funcionan alrededor del mundo y duran tanto tiempo como sea aplicable el derecho de autor (pues se basan en el derecho de autor). Estas

características en común sirven como la base a partir de la cual los licenciantes pueden optar por entregar más permisos cuando decidan cómo quieren que su obra sea utilizada.

Un licenciante Creative Commons debe responder un par de simples preguntas en la vía a escoger una licencia: primero, ¿quiero permitir uso comercial o no?, y segundo, ¿quiero permitir obras derivadas o no? Si un licenciante decide permitir obras derivadas, puede también elegir que cualquiera que use la obra -los llamamos licenciatarios- pueda hacer que esa obra esté disponible bajo las mismas condiciones. Llamamos a esta idea "CompartirIgual" y es uno de los mecanismos que (de ser escogido) permite que los procomunes digitales crezcan con el tiempo. CompartirIgual se inspira en la licencia GNU General Public License, utilizada por muchos proyectos de software libre y de código abierto.

Nuestras licencias no afecta a las libertades que la ley otorga a los usuarios de obras creativas protegidas por derechos de autor, como las excepciones y limitaciones a los derechos de autor como los usos justos. Las licencias Creative Commons exigen que los licenciatarios obtengan permiso para hacer cualquier cosa con una obra que la ley reserve exclusivamente al licenciante y que la licencia no permita expresamente. Los licenciatarios deben dar crédito al licenciatario, mantener los avisos de derechos de autor intactos en todas las copias de la obra, y enlazar a la licencia desde las copias de la obra. Los licenciatarios no pueden utilizar medidas técnicas para restringir el acceso a la obra.

2.3. TRES “CAPAS” DE LAS LICENCIAS

Nuestras licencias públicas de derechos de autor incorporan un exclusivo e innovador diseño de "tres capas". Cada licencia comienza como un instrumento legal tradicional, en el tipo

de lenguaje y formato de texto conocidos y armados por los abogados. A esto lo llamamos la capa de Código Legal de cada licencia.

Pero debido a que la mayoría de los creadores, educadores y científicos no son abogados, también hacemos disponibles las licencias en un formato que las personas normales pueden leer: el resumen de la licencia o "Commons Deed" (también conocido como la versión "legible por humanos" de la licencia). Se trata de una referencia práctica para licenciantes y licenciatarios, que resume y expresa algunos de los términos y condiciones más importantes. Piensa en el Commons Deed como una interfaz amistosa para el Código Legal que está debajo, aunque el resumen en sí mismo no es una licencia y su contenido no es parte del Código Legal propiamente tal.

La capa final de la licencia reconoce que el software, desde los motores de búsqueda pasando por la ofimática hasta llegar a la edición de música, juega un papel importante en la creación, copiado, difusión y distribución de obras. A fin de facilitar que la Web sepa dónde hay obras disponibles bajo licencias Creative Commons, entregamos una versión "legible por máquinas" de la licencia: un resumen de los derechos y obligaciones clave escritos en un formato tal que los sistemas informáticos, motores de búsqueda y otras formas de tecnología pueden entender. Para lograr esto, hemos desarrollado un modo estandarizado de describir las licencias que el software puede entender denominado CC Rights Expression Language (CC REL).

Buscar por contenido abierto es una importante función habilitada por nuestra propuesta. Puedes usar Google para buscar contenido con Creative Commons, buscar imágenes en Flickr, discos en Jamendo, y medios en general en spinxpress. Wikimedia Commons, el repositorio multimedia de Wikipedia, es uno de nuestros principales usuarios de licencias.

En conjunto, estas tres capas de licencias aseguran que el espectro de derechos no es solamente un concepto legal. Es algo que los creadores de obras pueden entender, sus usuarios pueden entender, y hasta la propia Web puede entender.

2.4. LA LICENCIA

Reconocimiento-CompartirIgual (CC BY-SA)

Esta licencia permite a otros remezclar, retocar, y crear a partir de tu obra, incluso con fines comerciales, siempre y cuando te den crédito y licencien sus nuevas creaciones bajo condiciones idénticas. Esta licencia suele ser comparada con las licencias "copyleft" de software libre y de código abierto. Todas las nuevas obras basadas en la tuya portarán la misma licencia, así que cualesquiera obras derivadas permitirán también uso comercial. Esa es la licencia que usa Wikipedia, y se recomienda para materiales que se beneficiarían de incorporar contenido de Wikipedia y proyectos con licencias similares.

2.5. RESUMEN DE LA LICENCIA

Usted es libre de:

- Copiar, distribuir y comunicar públicamente la obra.
- **Remezclar** — transformar la obra.
- Hacer un uso comercial de esta obra.

Bajo las condiciones siguientes:

- **Reconocimiento** — Debe reconocer los créditos de la obra de la manera especificada por el autor o el licenciador (pero no de una manera que sugiera que tiene su apoyo o apoyan el uso que hace de su obra).
- **Compartir bajo la misma licencia** — Si altera o transforma esta obra, o genera una obra derivada, sólo puede distribuir la obra generada bajo una licencia idéntica a ésta.

Entendiendo que:

- **Renuncia** — Alguna de estas condiciones puede **no aplicarse** si se obtiene el permiso del titular de los derechos de autor
- **Dominio Público** — Cuando la obra o alguno de sus elementos se halle en el **dominio público** según la ley vigente aplicable, esta situación no quedará afectada por la licencia.
- **Otros derechos** — Los derechos siguientes no quedan afectados por la licencia de ninguna manera:
    - Los derechos derivados de **usos legítimos** u otras limitaciones reconocidas por ley no se ven afectados por lo anterior.

- Los derechos **morales** del autor;
- Derechos que pueden ostentar otras personas sobre la propia obra o su uso, como por ejemplo **derechos de imagen** o de privacidad.

- **Aviso —** Al reutilizar o distribuir la obra, tiene que dejar bien claro los términos de la licencia de esta obra.

2.6. CÓDIGO LEGAL

Reconocimiento “compartirigual 3.0 España.

Creative commons corporation no es un despacho de abogados y no proporciona servicios jurídicos. La distribución de esta licencia no crea una relación abogado-cliente. Creative commons proporciona esta información tal cual (on an "as-is" basis). Creative commons no ofrece garantía alguna respecto de la información proporcionada, ni asume responsabilidad alguna por daños producidos a consecuencia de su uso.

2.7. LICENCIA CCPL

La obra o la prestación (según se definen más adelante) se proporciona bajo los términos de esta licencia pública de creative commons (***ccpl*** o ***licencia***). La obra o la prestación se encuentra protegida por la ley española de propiedad intelectual y/o cualesquiera otras normas que resulten de aplicación. Queda prohibido cualquier uso de la obra o prestación diferente a lo autorizado bajo esta licencia o lo dispuesto en la ley de propiedad intelectual mediante el ejercicio de cualquier derecho sobre la obra o la prestación, usted acepta y consiente las limitaciones y obligaciones de esta licencia, sin perjuicio de la necesidad de consentimiento expreso en caso de violación previa de los términos de la misma. El licenciador le concede los derechos contenidos en esta licencia, siempre que usted acepte los presentes términos y condiciones.

2.7.1. DEFINICIONES.

a. La ***obra*** es la creación literaria, artística o científica ofrecida bajo los términos de esta licencia.
b. En esta licencia se considera una ***prestación*** cualquier interpretación, ejecución, fonograma, grabación audiovisual, emisión o transmisión, mera fotografía u otros objetos protegidos por la legislación de propiedad intelectual vigente aplicable.

c. La aplicación de esta licencia a una ***colección*** (definida más adelante) afectará únicamente a su estructura en cuanto forma de expresión de la selección o disposición de sus contenidos, no siendo extensiva a éstos. En este caso la colección tendrá la consideración de obra a efectos de esta licencia.

d. El ***titular originario*** es:

   i. En el caso de una obra literaria, artística o científica, la persona natural o grupo de personas que creó la obra.

   ii. En el caso de una obra colectiva, la persona que la edite y divulgue bajo su nombre, salvo pacto contrario.

   iii. En el caso de una interpretación o ejecución, el actor, cantante, músico, o cualquier otra persona que represente, cante, lea, recite, interprete o ejecute en cualquier forma una obra.

   iv. En el caso de un fonograma, el productor fonográfico, es decir, la persona natural o jurídica bajo cuya iniciativa y responsabilidad se realiza por primera vez una fijación exclusivamente sonora de la ejecución de una obra o de otros sonidos.

   v. En el caso de una grabación audiovisual, el productor de la grabación, es decir, la persona natural o jurídica que tenga la iniciativa y asuma la responsabilidad de las fijaciones de un plano o secuencia de imágenes, con o sin sonido.

   vi. En el caso de una emisión o una transmisión, la entidad de radiodifusión.

   vii. En el caso de una mera fotografía, aquella persona que la haya realizado.

   viii. En el caso de otros objetos protegidos por la legislación de propiedad intelectual vigente, la persona que ésta señale.

e. Se considerarán ***obras derivadas*** aquellas obras creadas a partir de la licenciada, como por ejemplo: las traducciones y adaptaciones; las revisiones, actualizaciones y anotaciones; los compendios, resúmenes y extractos; los arreglos musicales y, en general, cualesquiera transformaciones de una obra literaria, artística o científica. Para evitar la duda, si la obra consiste en una composición musical o grabación de sonidos, la sincronización temporal de la obra con una imagen en movimiento (*synching*) será considerada como una obra derivada a efectos de esta licencia.

f. Tendrán la consideración de ***colecciones*** la recopilación de obras ajenas, de datos o de otros elementos independientes como las antologías y las bases de datos que por la selección o disposición de sus contenidos constituyan creaciones intelectuales. La

mera incorporación de una obra en una colección no dará lugar a una derivada a efectos de esta licencia.

g. El ***licenciador*** es la persona o la entidad que ofrece la obra o prestación bajo los términos de esta licencia y le concede los derechos de explotación de la misma conforme a lo dispuesto en ella.

h. ***Usted*** es la persona o la entidad que ejercita los derechos concedidos mediante esta licencia y que no ha violado previamente los términos de la misma con respecto a la obra o la prestación, o que ha recibido el permiso expreso del licenciador de ejercitar los derechos concedidos mediante esta licencia a pesar de una violación anterior.

i. La ***transformación*** de una obra comprende su traducción, adaptación y cualquier otra modificación en su forma de la que se derive una obra diferente. La creación resultante de la transformación de una obra tendrá la consideración de obra derivada.

j. Se entiende por ***reproducción*** la fijación directa o indirecta, provisional o permanente, por cualquier medio y en cualquier forma, de toda la obra o la prestación o de parte de ella, que permita su comunicación o la obtención de copias.

k. Se entiende por ***distribución*** la puesta a disposición del público del original o de las copias de la obra o la prestación, en un soporte tangible, mediante su venta, alquiler, préstamo o de cualquier otra forma.

l. Se entiende por ***comunicación pública*** todo acto por el cual una pluralidad de personas, que no pertenezcan al ámbito doméstico de quien la lleva a cabo, pueda tener acceso a la obra o la prestación sin previa distribución de ejemplares a cada una de ellas. Se considera comunicación pública la puesta a disposición del público de obras o prestaciones por procedimientos alámbricos o inalámbricos, de tal forma que cualquier persona pueda acceder a ellas desde el lugar y en el momento que elija.

m. La ***explotación*** de la obra o la prestación comprende la reproducción, la distribución, la comunicación pública y, en su caso, la transformación.

n. Los ***elementos de la licencia*** son las características principales de la licencia según la selección efectuada por el licenciador e indicadas en el título de esta licencia: Reconocimiento, CompartirIgual.

o. Una ***licencia equivalente*** es:

    i. Una versión posterior de esta licencia de Creative Commons con los mismos elementos de licencia.

ii. La misma versión o una versión posterior de esta licencia de cualquier otra jurisdicción reconocida por Creative Commons con los mismos elementos de la licencia (ejemplo: Reconocimiento-CompartirIgual 3.0 Japón).

iii. La misma versión o una versión posterior de la licencia de Creative Commons no adaptada a ninguna jurisdicción (*Unported*) con los mismos elementos de la licencia.

iv. Una de las licencias compatibles que aparece en http://creativecommons.org/compatiblelicenses y que ha sido aprobada por Creative Commons como esencialmente equivalente a esta licencia porque, como mínimo:

   a. Contiene términos con el mismo propósito, el mismo significado y el mismo efecto que los elementos de esta licencia.

   b. Permite explícitamente que las obras derivadas de obras sujetas a ella puedan ser distribuidas mediante esta licencia, la licencia de Creative Commons no adaptada a ninguna jurisdicción (*Unported*) o una licencia de cualquier otra jurisdicción reconocida por Creative Commons, con sus mismos elementos de licencia.

2.7.2. LÍMITES DE LOS DERECHOS.

Nada en esta licencia pretende reducir o restringir cualesquiera límites legales de los derechos exclusivos del titular de los derechos de propiedad intelectual de acuerdo con la Ley de propiedad intelectual o cualesquiera otras leyes aplicables, ya sean derivados de usos legítimos, tales como la copia privada o la cita, u otras limitaciones como la resultante de la primera venta de ejemplares (agotamiento).

2.7.3. CONCESIÓN DE LICENCIA.

Conforme a los términos y a las condiciones de esta licencia, el licenciador concede, por el plazo de protección de los derechos de propiedad intelectual y a título gratuito, una licencia de ámbito mundial no exclusiva que incluye los derechos siguientes:

a. Derecho de reproducción, distribución y comunicación pública de la obra o la prestación.

b. Derecho a incorporar la obra o la prestación en una o más colecciones.

c. Derecho de reproducción, distribución y comunicación pública de la obra o la prestación lícitamente incorporada en una colección.

d. Derecho de transformación de la obra para crear una obra derivada siempre y cuando se incluya en ésta una indicación de la transformación o modificación efectuada.

e. Derecho de reproducción, distribución y comunicación pública de obras derivadas creadas a partir de la obra licenciada.

f. Derecho a extraer y reutilizar la obra o la prestación de una base de datos.

g. Para evitar cualquier duda, el titular originario:

   i. Conserva el derecho a percibir las remuneraciones o compensaciones previstas por actos de explotación de la obra o prestación, calificadas por la ley como irrenunciables e inalienables y sujetas a gestión colectiva obligatoria.

   ii. Renuncia al derecho exclusivo a percibir, tanto individualmente como mediante una entidad de gestión colectiva de derechos, cualquier remuneración derivada de actos de explotación de la obra o prestación que usted realice.

Estos derechos se pueden ejercitar en todos los medios y formatos, tangibles o intangibles, conocidos en el momento de la concesión de esta licencia. Los derechos mencionados incluyen el derecho a efectuar las modificaciones que sean precisas técnicamente para el ejercicio de los derechos en otros medios y formatos. Todos los derechos no concedidos expresamente por el licenciador quedan reservados, incluyendo, a título enunciativo pero no limitativo, los derechos morales irrenunciables reconocidos por la ley aplicable. En la medida en que el licenciador ostente derechos exclusivos previstos por la ley nacional vigente que implementa la directiva europea en materia de derecho sui generis sobre bases de datos, renuncia expresamente a dichos derechos exclusivos.

2.7.4. RESTRICCIONES.

La concesión de derechos que supone esta licencia se encuentra sujeta y limitada a las restricciones siguientes:

a. Usted puede reproducir, distribuir o comunicar públicamente la obra o prestación solamente bajo los términos de esta licencia y debe incluir una copia de la misma, o su Identificador Uniforme de Recurso (URI). Usted no puede ofrecer o imponer ninguna condición sobre la obra o prestación que altere o restrinja los términos de esta licencia

o el ejercicio de sus derechos por parte de los concesionarios de la misma. Usted no puede sub-licenciar la obra o prestación. Usted debe mantener intactos todos los avisos que se refieran a esta licencia y a la ausencia de garantías. Usted no puede reproducir, distribuir o comunicar públicamente la obra o prestación con medidas tecnológicas que controlen el acceso o el uso de una manera contraria a los términos de esta licencia. Esta sección 4.a también afecta a la obra o prestación incorporada en una colección, pero ello no implica que ésta en su conjunto quede automáticamente o deba quedar sujeta a los términos de la misma. En el caso que le sea requerido, previa comunicación del licenciador, si usted incorpora la obra en una colección y/o crea una obra derivada, deberá quitar cualquier crédito requerido en el apartado 4.c, en la medida de lo posible.

b. Usted puede distribuir o comunicar públicamente una obra derivada en el sentido de esta licencia solamente bajo los términos de la misma u otra licencia equivalente. Si usted utiliza esta misma licencia debe incluir una copia o bien su URI, con cada obra derivada que usted distribuya o comunique públicamente. Usted no puede ofrecer o imponer ningún término respecto a la obra derivada que altere o restrinja los términos de esta licencia o el ejercicio de sus derechos por parte de los concesionarios de la misma. Usted debe mantener intactos todos los avisos que se refieran a esta licencia y a la ausencia de garantías cuando distribuya o comunique públicamente la obra derivada. Usted no puede ofrecer o imponer ningún término respecto de las obras derivadas o sus transformaciones que alteren o restrinjan los términos de esta licencia o el ejercicio de sus derechos por parte de los concesionarios de la misma. Usted no puede reproducir, distribuir o comunicar públicamente la obra derivada con medidas tecnológicas que controlen el acceso o uso de la obra de una manera contraria a los términos de esta licencia. Si utiliza una licencia equivalente debe cumplir con los requisitos que ésta establezca cuando distribuya o comunique públicamente la obra derivada. Todas estas condiciones se aplican a una obra derivada en tanto que incorporada a una colección, pero no implica que ésta tenga que estar sujeta a los términos de esta licencia.

c. Si usted reproduce, distribuye o comunica públicamente la obra o la prestación, una colección que la incorpore o cualquier obra derivada, debe mantener intactos todos los avisos sobre la propiedad intelectual e indicar, de manera razonable conforme al medio o a los medios que usted esté utilizando:

    i. El nombre del autor original, o el seudónimo si es el caso, así como el del titular originario, si le es facilitado.

ii. El nombre de aquellas partes (por ejemplo: institución, publicación, revista) que el titular originario y/o el licenciador designen para ser reconocidos en el aviso legal, las condiciones de uso, o de cualquier otra manera razonable.

iii. El título de la obra o la prestación si le es facilitado.

iv. El URI, si existe, que el licenciador especifique para ser vinculado a la obra o la prestación, a menos que tal URI no se refiera al aviso legal o a la información sobre la licencia de la obra o la prestación.

v. En el caso de una obra derivada, un aviso que identifique la transformación de la obra en la obra derivada (p. ej., "traducción castellana de la obra de Autor Original," o "guion basado en obra original de Autor Original").

Este reconocimiento debe hacerse de manera razonable. En el caso de una obra derivada o incorporación en una colección estos créditos deberán aparecer como mínimo en el mismo lugar donde se hallen los correspondientes a otros autores o titulares y de forma comparable a los mismos. Para evitar la duda, los créditos requeridos en esta sección sólo serán utilizados a efectos de atribución de la obra o la prestación en la manera especificada anteriormente. Sin un permiso previo por escrito, usted no puede afirmar ni dar a entender implícitamente ni explícitamente ninguna conexión, patrocinio o aprobación por parte del titular originario, el licenciador y/o las partes reconocidas hacia usted o hacia el uso que hace de la obra o la prestación.

d. Para evitar cualquier duda, debe hacerse notar que las restricciones anteriores (párrafos 4.a, 4.b y 4.c) no son de aplicación a aquellas partes de la obra o la prestación objeto de esta licencia que únicamente puedan ser protegidas mediante el derecho sui generis sobre bases de datos recogido por la ley nacional vigente implementando la directiva europea de bases de datos

2.7.5. EXONERACIÓN DE RESPONSABILIDAD

A menos que se acuerde mutuamente entre las partes, el licenciador ofrece la obra o la prestación tal cual (on an "as-is" basis) y no confiere ninguna garantía de cualquier tipo respecto de la obra o la prestación o de la presencia o ausencia de errores que puedan o no ser descubiertos. Algunas jurisdicciones no permiten la exclusión de tales garantías, por lo que tal exclusión puede no ser de aplicación a usted.

2.7.6. LIMITACIÓN DE RESPONSABILIDAD

Salvo que lo disponga expresa e imperativamente la ley aplicable, en ningún caso el licenciador será responsable ante usted por cualesquiera daños resultantes, generales o especiales (incluido el daño emergente y el lucro cesante), fortuitos o causales, directos o indirectos, producidos en conexión con esta licencia o el uso de la obra o la prestación, incluso si el licenciador hubiera sido informado de la posibilidad de tales daños.

2.7.7. FINALIZACIÓN DE LA LICENCIA

a. Esta licencia y la concesión de los derechos que contiene terminarán automáticamente en caso de cualquier incumplimiento de los términos de la misma. Las personas o entidades que hayan recibido de usted obras derivadas o colecciones bajo esta licencia, sin embargo, no verán sus licencias finalizadas, siempre que tales personas o entidades se mantengan en el cumplimiento íntegro de esta licencia. Las secciones 1, 2, 5, 6, 7 y 8 permanecerán vigentes pese a cualquier finalización de esta licencia.

b. Conforme a las condiciones y términos anteriores, la concesión de derechos de esta licencia es vigente por todo el plazo de protección de los derechos de propiedad intelectual según la ley aplicable. A pesar de lo anterior, el licenciador se reserva el derecho a divulgar o publicar la obra o la prestación en condiciones distintas a las presentes, o de retirar la obra o la prestación en cualquier momento. No obstante, ello no supondrá dar por concluida esta licencia (o cualquier otra licencia que haya sido concedida, o sea necesario ser concedida, bajo los términos de esta licencia), que continuará vigente y con efectos completos a no ser que haya finalizado conforme a lo establecido anteriormente, sin perjuicio del derecho moral de arrepentimiento en los términos reconocidos por la ley de propiedad intelectual aplicable.

2.7.8. MISCELÁNEA

a. Cada vez que usted realice cualquier tipo de explotación de la obra o la prestación, o de una colección que la incorpore, el licenciador ofrece a los terceros y sucesivos licenciatarios la concesión de derechos sobre la obra o la prestación en las mismas condiciones y términos que la licencia concedida a usted.

b. Cada vez que usted realice cualquier tipo de explotación de una obra derivada, el licenciador ofrece a los terceros y sucesivos licenciatarios la concesión de derechos sobre la obra objeto de esta licencia en las mismas condiciones y términos que la licencia concedida a usted.

c. Si alguna disposición de esta licencia resulta inválida o inaplicable según la Ley vigente, ello no afectará la validez o aplicabilidad del resto de los términos de esta licencia y, sin ninguna acción adicional por cualquiera las partes de este acuerdo, tal disposición se entenderá reformada en lo estrictamente necesario para hacer que tal disposición sea válida y ejecutiva.

d. No se entenderá que existe renuncia respecto de algún término o disposición de esta licencia, ni que se consiente violación alguna de la misma, a menos que tal renuncia o consentimiento figure por escrito y lleve la firma de la parte que renuncie o consienta.

e. Esta licencia constituye el acuerdo pleno entre las partes con respecto a la obra o la prestación objeto de la licencia. No caben interpretaciones, acuerdos o condiciones con respecto a la obra o la prestación que no se encuentren expresamente especificados en la presente licencia. El licenciador no estará obligado por ninguna disposición complementaria que pueda aparecer en cualquier comunicación que le haga llegar usted. Esta licencia no se puede modificar sin el mutuo acuerdo por escrito entre el licenciador y usted.

2.7.9. AVISO DE CREATIVE COMMONS

Creative Commons no es parte de esta licencia, y no ofrece ninguna garantía en relación con la obra o la prestación. Creative Commons no será responsable frente a usted o a cualquier parte, por cualesquiera daños resultantes, incluyendo, pero no limitado, daños generales o especiales (incluido el daño emergente y el lucro cesante), fortuitos o causales, en conexión con esta licencia. A pesar de las dos (2) oraciones anteriores, si Creative Commons se ha identificado expresamente como el licenciador, tendrá todos los derechos y obligaciones del licenciador.

Salvo para el propósito limitado de indicar al público que la obra o la prestación está licenciada bajo la CCPL, ninguna parte utilizará la marca registrada "Creative Commons" o cualquier marca registrada o insignia relacionada con "Creative Commons" sin su consentimiento por escrito. Cualquier uso permitido se hará de conformidad con las pautas vigentes en cada momento sobre el uso de la marca registrada por "Creative Commons", en tanto que sean publicadas en su sitio web (website) o sean proporcionadas a petición previa.

Para evitar cualquier duda, estas restricciones en el uso de la marca no forman parte de esta licencia.

Puede contactar con Creative Commons en: **http://creativecommons.org/**.

ANEXOS

ÍNDICE:

1. ANEXO I **03**

1.1. CONCLUSIONES DEL PROYECTO 03

2. ANEXO II **04**

2.1. BIBLIOGRAFÍA 04

2.2. WEBGRAFÍA 04

1. ANEXO I

1.1. CONCLUSIONES DEL PROYECTO

Este trabajo nos ha servido para interiorizarnos y aprender de una manera adecuada como se realiza un proyecto de investigación para en un futuro tener una mejor noción a la hora de realizar una investigación, los pasos que se llevan a cabo y la forma correcta de para realizarlo, de esta manera tenemos un idea más formada del trabajo al cual se enfrenta un verdadero investigador.

Al haber expuesto el proyecto de investigación en una WEB nos pusimos en la tarea de aprender la forma en que se creaba una estructura de este tipo, como se manejaba, cuál era la mejor forma de presentar la información, perfeccionando la exposición del trabajo de la mejor manera posible. Esto hizo que adquiriéramos un conocimiento que, si bien no tiene nada que ver con la materia que estamos desarrollando (estudio del microcontrolador avanzado dsPIC), nos enriquece en otro campo que es muy útil y hoy en día se emplea masivamente.

La elección del tema expuesto tuvo origen en el interés que se nos presentó a los dos integrantes de este grupo cuando reflexionábamos sobre los avances tecnológicos y como consiguiente sobre el avance de la electrónica. Hace no mucho tiempo, no se requería tanta precisión en cálculos, ni de tanta calidad de imagen en las televisiones, por poner un ejemplo; antes con la calidad y almacenaje que nos proporcionaba el DVD, no necesitábamos de una TV de alta de definición, después evoluciona y aparece el blue-ray con mucha más capacidad y calidad, por lo que asociado a esto, necesitamos de una TV más potente y que nos rinda esta mejora,… ¿Qué quiero decir con esto?, pues quiero decir que el dsPIC nace como una necesidad de evolución de los microcontroladores anteriores ante el próspero avance tecnológico que estamos viviendo.

El dsPIC aparte de mejorar en tamaño, robustez, capacidad de periféricos, rapidez…etc, lo que hace es procesar las señales digitalmente, y esto es muy importante cuando el volumen de datos a manejar (cada vez mayor por el avance tecnológico) es grande, y es lo que le diferencia sobre todo de los antecesores (microcontroladores *PIC*).

A raíz de esto nos empezamos a informar sobre el tema en libros sobre estos dispositivos, además de realizar un amplio escaneo por la WEB.

A partir de ahí empezamos a trabajar en la investigación desarrollando un planteamiento concreto del estudio de interés basándonos principalmente en la arquitectura interna y periféricos de estos dispositivos.

Luego comenzamos a indagar sobre como plasmar la información obtenida de forma concreta y estructurada en una página WEB.

Por último realizamos un extenso estudio, para averiguar cómo poder simular el funcionamiento de estos dispositivos mediante diversos programas software, llegando a la conclusión de que una vez comprendida la información de la WEB (cómo funciona el dispositivo) y eligiendo correctamente el software, es relativamente fácil realizar la simulación del funcionamiento de los dsPIC.

2. ANEXO II

2.1. BIBLIOGRAFÍA

ANGULO USATEGUI, José María; ANGULO MARTÍNEZ, Ignacio; GARCÍA ZAPIRAIN, Begoña ... [et al.], (aut.). *MICROCONTROLADORES AVANZADOS dsPIC. Controladores digitales de señales. Arquitectura, programación y aplicaciones.* Ediciones Paraninfo. S.A, 2005. ISBN: 8497323858.

ANGULO USATEGUI, José María; ETXEBARRÍA RUIZ, Aritza; ANGULO MARTÍNEZ, Ignacio... [et al.], (aut.). *Microcontroladores dsPIC. Diseño práctico de aplicaciones.* MC Graw Hill, 2006. ISBN: 84-481-5159-9.

2.1.1.1. WEBGRAFÍA

[Página oficial de Microchip]
http://www.microchip.com

[Licencia de la WEB]
http://creativecommons.org

[Licencia GNU para el software]
http://www.gnu.org/licenses/licenses.es.html

[Curso de Dreamweaver]
http://www.aulaclic.es/dreamweaver-cs5/index.htm

[Programación compilador C30]
http://picmania.garcia-cuervo.net/c30.php

[Programación compilador C CCS]
http://www.micropic.es/mpblog

[Foro sobre los dsPIC]
http://todopic.mforos.com/781922-dspic

[Aplicaciones dsPIC33F]
http://nexp.tistory.com/715

Resumen del Proyecto Fin de Carrera:

"Estudio del microcontrolador avanzado dsPIC"

Realizado por: Juan José Rodríguez Vega

Mariano del Campo García

Dirigido por: Francisco Plaza Pérez

Dto. Tecnología Electrónica – ESCUELA DE INGENIERÍAS INDUSTRIALES

Valladolid, 13 de Junio de 2013

1. INTRODUCCIÓN

La empresa Microchip Technology Inc. ® ocupa el primer puesto en el ranking mundial de microcontroladores de 8 bits desde el año 2003; sus modelos son conocidos popularmente con el nombre genérico de PIC®. Tras el exitoso lanzamiento de las familias de microcontroladores de 16 bits PIC24F y PIC24H, los usuarios necesitan nuevos dispositivos que soporten funciones de procesamiento digital de señales para atender las nuevas tendencias del mercado orientadas al aumento de la conectividad por Internet, las mejoras relacionadas con la imagen y el sonido, el tratamiento matemático de las señales, los sistemas de alimentación, el control de motores y sensores, etc.

Las aplicaciones modernas mezclan las funciones típicas *MCU (microcontrolador)* con las funciones de un *DSP (Procesador digital de señales).* Esta situación ha impulsado a Microchip a fabricar un circuito hibrido MCU/DSP cuyo manejo es similar a los clásicos microcontroladores pero que incluye las principales prestaciones de los DSP. Así ha nacido el Controlador Digital de Señales, abreviadamente *DSC*®, que reúne las características de un microcontrolador PIC de I6 bits (MCU) y las de un DSP de gama baja.

dsPIC es un nombre genérico que se utiliza para referirse a los controladores digitales de señales (DSC) que ha diseñado “Microchip” para facilitar a los usuarios, la transición al campo de las aplicaciones de los procesos digitales de señales. Son los primeros PIC con bus de datos inherente de 16 bits. Incorporan todas las posibilidades de los anteriores PIC y añaden varias operaciones DSP implementadas en hardware, como multiplicación con suma de acumulador (MAC), inversión de acarreo o multiplicación 16x16 bits con signo. Cabe destacar las siguientes características: Set de instrucciones de un solo ciclo (Incluida la multiplicación); trabajan con un BUS de datos de 16 bits (coma fija o flotante); incorporan una memoria RAM grande; gran velocidad y algoritmos complejos (gran rendimiento); modos de direccionamiento pensados para el procesamiento; disponen de conversores A/D rápidos y precisos; están preparados para ser programados en lenguajes de alto nivel.

2. FAMILIAS dsPIC

Comienzan a producirse a gran escala a finales de 2004, pero en la actualidad ya se comercializan más de 100 modelos de este tipo de dispositivo, utilizados para aplicaciones avanzadas MCU y de audio (propósito general), y para el control de sensores, motores y sistemas de alimentación, reunidos en las dos familias siguientes:

- *1ª FAMILIA: dsPIC30F* → Arquitectura y juego de instrucciones de los MCU tradicionales, añadiendo la funcionalidad y principales requisitos de los DSP.
- *2ª FAMILIA: dsPIC33F/E* → Se potencian las capacidades, el número de periféricos y el rendimiento, permitiendo acceder a campos de aplicación más complejos.

Las diferencias más significativas entre las dos familias son las siguientes:

dsPIC30F	*dsPIC33F/E*
21 modelos disponibles	103 modelos disponibles
-	Controlador de DMA (8 canales)
5 temporizadores	9 temporizadores
Modos de trabajo con baja energía: 1. IDLE 2. SLEEP	Modos de trabajo con baja energía: 1. IDLE 2. SLEEP 3. DOZE (frecuencia más baja)
Alimentación de 2 a 5,5 V	Alimentación de 2 a 3,6 V
Rendimiento: 30 MIPS a 4,5 o 5,5 V	Rendimiento: Hasta 70 MIPS a 3,3 V
Pocos vectores de Interrupción (62)	Muchos vectores de Interrupción (118)
Memoria EEPROM	-
Memoria FLASH de hasta 144 KB	Memoria FLASH de hasta 536 KB
Memoria SRAM de hasta 8 KB	Memoria SRAM de hasta 48 KB
Abundantes periféricos	Más periféricos

3. ESTUDIO DEL dsPIC30F6010

Para realizar el estudio del dsPIC hemos seleccionado el dsPIC30F6010 puesto que es considerado uno de los más representativos de la familia 30F. Se trata de un microcontrolador avanzado utilizado en aplicaciones de control de motores de inducción de fase simple o trifásica y los de corriente continua. También son muy apropiados para la gestión de los sistemas de alimentación ininterrumpibles, conversores, módulos para la corrección del factor de potencia, etc. Además se utilizan en sistemas destinados a la calefacción, ventilación, lavadoras, apertura industrial de puertas, control de estabilidad, medición del caudal del agua y consumo de electricidad, sistemas de seguridad, etc. Cabe destacar las siguientes características de este dispositivo:

- Arquitectura de 16 bits.
- 30 MIPS de velocidad máxima de la CPU.
- Memorias:
  - Memoria de datos tipo SRAM de 8KB.
  - Memoria de programa tipo FLASH de 144KB.
  - Memoria no volátil EEPROM de 4KB.
- Rango de temperatura de trabajo: de - 40ºC a 125ºC.
- Rango de voltajes de alimentación: de 2,5V a 5,5V.
- Encapsulado con 80 terminales en formato TQFP.

- Periféricos integrados:
  - 5 temporizadores de 16 bits y 2 temporizadores de 32 bits.
  - Conversor A/D de 16 canales, 10 bits de resolución y velocidad de 1Msps.
  - Periféricos de Captura/Comparación/PWM.
  - 8 Canales PWM para el control de motores.
  - Periféricos de comunicación digital: UART, SPI e I^2C.
  - Interfaz para el codificador de cuadratura.
  - Controlador de Red de Área (CAN).
  - No posee ni interfaz códec (DCI), ni acceso directo a memoria (DMA).
- Incorpora WDT (Perro guardián) y POR (Reset en el encendido).
- Osciladores internos de 7.37 MHz, y 512 kHz.

4. DESARROLLO DE LA PÁGINA WEB

Para la creación de la página web hemos seguido la pauta del Tutor y la lógica, puesto que al ser una web formativa, se trata más de enseñar la información en ella contenida que el continente. No hemos hecho un diseño espectacular ni complicado, puesto que se trataba de confeccionar algo sencillo fácilmente programable y editable, para darle más versatilidad al proyecto y que pueda servir de plantilla a otros estudios futuros.

Hemos intentado que desde la primera página se acceda rápidamente a simple vista a todos los contenidos fundamentales del proyecto. Al pasar el ratón por encima de los botones del menú, se desplegaran otros submenús importantes, para poder acceder a toda la información incorporada en la WEB consiguiendo que con una simple pasada sepamos donde está todo y se pueda acceder rápida y cómodamente a cualquier tema o bloque.

Nuestra página web ha sido realizada a través del programa Dreamweaver CS5.5, basándonos en el diseño de una plantilla que nosotros hemos creado con las siguientes características:

- Una cabecera (header) en la que incluimos el título “Estudio del dsPIC” y una imagen que describe gráficamente a los dsPIC.
- Dentro del cuerpo de la página (body) incluimos un menú desplegable SPRY (ágil), gobernado por funciones JavaScript en el que incluimos la distribución del “Mapa WEB”, mediante vínculos a los correspondientes HTML.
- Creamos una región editable (dentro del body), que será la parte de la plantilla que podrá ser editada cuando creemos las diferentes páginas (HTML) de la WEB basándonos en dicha plantilla, y además, también creamos unas regiones opcionales en las que incluimos los botones de descarga (de aplicaciones), ir arriba y cuestionario, los cuales han sido dotados de imágenes decorativas e integrados en las páginas en las que son necesarios.

- Por último, y no menos importante, hemos creado un pie de página (footer) en el que incluimos las imágenes corporativas de la Universidad de Valladolid (UVA) y de la Escuela de ingenierías Industriales (EII), en las cuales insertamos un "Link" para acceder de manera rápida a las paginas oficiales de ambos estamentos. También incluimos los nombres de los dos diseñadores de la WEB, un validador CSS y la licencia de la WEB (Creative Commons).

Una vez descrito el contenido pasamos a comentar el desarrollo de las múltiples páginas incluidas en la WEB, para ello nos sobra con hacer una descripción general de ellas, pues funcionan todas de la misma manera, excepto que poseen diferentes contenidos de texto, imágenes, tablas, enlaces y en algún caso videos en formato FLASH.

- Todas las páginas están basadas en HTML 1.0 transicional, creadas a través de la plantilla que hemos descrito anteriormente, en las que incluimos básicamente texto e imágenes para describir cada apartado correctamente.
- Cada página está dotada de su correspondiente título, tanto en el encabezado como en la barra del navegador.
- Nos hemos limitado a trabajar con tablas para centrar las imágenes y los videos con mayor sencillez y comodidad (no incluimos etiquetas DIV para el posicionamiento).
- Todas las imágenes están optimizadas con Photoshop CS5.1, la mayoría de ellas han sido obtenidas de la propia WEB del fabricante (www.microchip.com) y posteriormente traducidas a nuestro idioma (Español) y retocadas para obtener la mayor calidad posible con el menor peso (que se abran rápidamente). Otra parte de las imágenes ha sido creada por nosotros, para ampliar la parte visual de la WEB, basándonos en la teoría a la que acompañan.
- Como pilar central del diseño, hemos trabajado con estilos CSS específicos, creados para cada tipo de objeto insertado en la WEB (texto, imágenes, videos, links…etc.).
- Para comprobar el correcto funcionamiento de la WEB en los diferentes navegadores (Internet Explorer, Google Chrome y Firefox), hemos trabajado tanto a nivel local, mediante un servidor "Apache" gratuito y de código abierto, como a nivel remoto mediante un cliente FTP llamado FileZilla para subir los archivos de la WEB al hosting gratuito (servidor remoto) que tenemos creado en *www.hostinger.es.*
- URL de la WEB: ***http://www.estudiodspic.16mb.com.***

Como último punto de este apartado Cabe destacar que los formularios que hemos integrado en la WEB, han sido creados mediante la aplicación "HotPotatoes 6", y cuentan con estilos CSS y funciones JavaScript independientes de la propia programación de la WEB, para que puedan ser ejecutados sin alterar el funcionamiento de la WEB (están basados en HTML 1.1).

5. DESARROLLO DE LAS APLICACIONES

En la creación de aplicaciones hemos utilizado muchos programas, unos con acierto y otros sólo nos han llevado a problemas de incompatibilidades y quebraderos de cabeza. La premisa del estudio es que las aplicaciones fueran realizadas para PROTEUS, a partir de ahí, nosotros hemos tenido que averiguar qué compiladores y que dispositivos eran compatibles con PROTEUS para poder realizar la programación de las aplicaciones, su compilación y depuración bajo el entorno VSM de PROTEUS, llegando a la conclusión de que los mejores compiladores para nuestro fin son el C COMPILER de CSS y el C30 de MPLAB, sobre todo este último puesto que es el compilador creado por el propio fabricante de los dispositivos (Microchip).

Todas las aplicaciones están desarrolladas para el dsPIC33F12MC202, puesto que contiene los mismos periféricos que el dsPIC30F6010 y es posible trabajar con él ya que si está implementado en las librerías de PROTEUS, cosa que no ocurre con los dispositivos de la familia 30F. Se trata de aplicaciones centradas en el funcionamiento de los periféricos más relevantes de los dsPIC como pueden ser los temporizadores, el conversor A/D, el módulo PWM para el control de motores o los módulos de comunicación.

PROTEUS es un software de diseño electrónico desarrollado por “Labcenter Electronics” que consta de dos módulos: ARES para el diseño de circuitos impresos e ISIS para la simulación de circuitos. Nosotros sólo hemos trabajado con el módulo ISIS, mediante el cual podemos diseñar el circuito que deseemos con componentes muy variados, desde una simple resistencia hasta algún que otro microprocesador o microcontrolador, incluyendo fuentes de alimentación, generadores de señales, osciloscopio digital, terminales virtuales y muchos otros recursos. Los diseños realizados en ISIS pueden ser simulados en tiempo real ayudándonos de prácticos controles de depuración paso a paso y visualización del estado de las variables.

El entorno VSM de PROTEUS nos permite enlazar el software compilado con la simulación del código en el microcontrolador y su interacción con cualquier componente electrónico digital o analógico conectado a él, funcionando como un completo simulador para esquemas electrónicos que contienen microprocesador. Se trata de una extensión de PROTEUS con la cual podremos simular, en tiempo real, el comportamiento de la mayoría de los periféricos y recursos de los microcontroladores dsPIC33F, introduciendo nosotros mismos el programa en código C que queremos que lleven a cabo, siendo una herramienta útil para estudiantes y profesionales que desean acelerar y mejorar sus habilidades para el desarrollo de aplicaciones analógicas y digitales. La simulación puede incluir instrumentos de medición y análisis para gráficas que representan las señales obtenidas en la simulación.